Computational Intelligence Techniques for Food and Health Sciences

Alberto Tonda

Habilitation à Diriger des Recherches

Computational Intelligence Techniques for Food and Health Sciences

Discipline : Informatique

Spécialité : Apprentissage Automatique

Alberto Tonda, Ph.D.

Prof. Stefano Di Carlo	Politecnico di Torino, Italy	Rapporteur
Prof. Laetitia Jourdan	Université de Lille 1, France	Rapporteur
Prof. Krzysztof Krawiec	Poznan University of Technology, Poland	Rapporteur
Dr. David Makowski	UMR 518 MIA-Paris, INRAE	Président
Prof. Gabriela Ochoa	University of Stirling, UK	Examinateur

All chapter header images are created by Dr. Evelyne Lutton, INRAE and the professional painter Emmanuel Cayla. The evolved fractal art has been generated with the interactive software ArtiE-Fract[1].

February 8, 2024

[1] `http://evelyne.lutton.free.fr/ArtiE-Fract.html`

Contents

1. Introduction

The *Habilitation à Diriger des Recherches* (HDR) is a French diploma, acknowledging that a researcher is considered experienced enough to be the main supervisor of Ph.D. students[1]. The following text, created as part of my application for the HDR, summarizes my career path and most relevant research lines, highlighting the works that I mostly contributed to, especially regarding the supervision of master and Ph.D. students, with the aim of showing the current degree of maturity in my research path.

1.1 My Career Path

My main research topics during my Ph.D. concerned the real-world applications of modeling and evolutionary computation (EC) techniques, mostly to hardware and software testing. Progressing to my post-doctoral positions, I gained an interest in machine learning (ML), that I see as optimization applied to models, and applications related to the food science and health sectors, where data is often scarce and incomplete, and models' predictions can be hard to interpret. Motivated by these difficulties, I developed theoretical approaches to successfully tackle the obstacles, and at the same time progressed with real-world applications.

My Ph.D. (obtained in 2011 by Politecnico di Torino, Italy, under the supervision of Prof. Giovanni Squillero) was focused on optimization, modeling, and reverse-engineering of complex systems. I am particularly interested by the great challenges presented to optimization techniques by real-world problems: from unpredictable interactions among elements of the same system, to non-linear relationships between variables, from superexponential complexity, to sparsity and scarcity of training data. My thesis manuscript was published as a book, edited by me and my supervisors [SST12].

After obtaining my Ph.D., I started a first post-doctoral position at Politecnico di Torino, in

[1] Or, citing the Arrêté du 23 novembre 1988 relatif à l'habilitation à diriger des recherches, *"[...] anctionne la reconnaissance du haut niveau scientifique du candidat, du caractère original de sa démarche dans un domaine de la science, de son aptitude à maîtriser une stratégie de recherche dans un domaine scientifique ou technologique suffisamment large et de sa capacité à encadrer de jeunes chercheurs."*

the same research team, working on the follow-up of the same topics. Between 2011 and 2012 I worked on my second post-doctoral position, at Institut des Systèmes Complexes, Paris, in the scope of the European FP7 project DREAM. My work in this period was focused on the development of new approaches for optimizing the structure of Bayesian networks models, applied to milk gel processing [Lut+14; Ton+13; Ton+12]. It's during this experience that I first came in contact with scientists from the French National Institute for Research in Agriculture, Food, and the Environment (INRAE), started discovering the intricacies of agri-food systems, and gained a raising interest in ML.

I have been recruited by INRAE as permanent researcher (*Chargé des recherches, classe normale*) in 2012, where I joined team MALICES, focused on modelling complex agri-food and biological systems (*Modélisation des Systèmes Alimentaires et Biologiques Complexes*). As my experience in research grew, I had the opportunity of leading an international networking project on modelling in food science and industry, COST Action FoodMC, between 2016 and 2020. In January 2020 my team merged with team LINK (*Learning and Integration of Knowledge*) to create team EKINOCS (*Expert Knowledge, INteractive modeling and learnINg for understandINg and decisiOn makINg in dYNamic Complex Systems*). Between my recruitment and the time of writing, my research topics slowly expanded from optimization to ML, and the applications went from being more strictly related to food processing to addressing broader questions in life sciences and health. Working on real-world applications informed my methodological developments, raising new research questions related to the interpretability of models and sparsity of data.

1.2 My Research

Since the beginning of my Ph.D., my research has been focused on real-world applications. Real-world optimization problems very often present features that make it impossible or impractical to apply traditional, exact techniques: for example, vast non-convex spaces of candidate solutions that cannot be exhaustively explored, or computationally expensive cost functions (also called *fitness functions*), or even a difficulty of translating the structure of a candidate solution into a simple array of real values. Evolutionary algorithms (EAs) [De 16], ranging from evolution strategies [Sch65] to genetic programming [Koz92], are uniquely suited to tackle these kind of issues, and consequently they have always been a central part of my research topics.

EAs are stochastic optimization techniques that explore the search space of possible solutions attempting to move towards areas with better values of a given fitness function. In order to achieve this objective, EAs keep an archive of candidate solutions called *population*, that can be seen as their current sampling of the fitness landscape. Starting from candidate solutions with the known best fitness values, new solutions are created as random mutations or recombination of existing ones, furthering the exploration of the solution search space. EAs are applied wherever classical optimization techniques fail, and even though there is no theoretical guarantee that they will be able to find the global optimum, corresponding to the absolute best candidate solution, they usually deliver solutions of high quality in a reasonable amount of computational time.

The first applications I tackled as a Ph.D. student dealt mostly with software and hardware testing [Car+11; Gan+10], and represented problems so complex that even classical EAs would struggle finding good solutions. These obstacles motivated my long-standing research into algorithmic aspects of EC, such as measuring diversity in the population of an EA [ST08] with the objective of preserving and promoting it [ST16], with the final objective of further improving the performance of these already efficient techniques.

When I first started stepping into the field of ML, mostly during my post-doctoral experiences, coming from the domain of optimization informed my view of ML algorithms as optimization techniques applied to models. From my perspective, ML is nothing more that transforming a learning problem into an optimization problem: ML algorithms explore a large search space of possible models, attempting to find the global optimum, defined as the model that not only fits the training data provided, but is able to generalize as much as possible to unseen data points. The ML community has, in my opinion, the great added advantage of having created a rich common vocabulary to describe and compare techniques as different as logistic regression [Pea96] and random forest [Bre01]. With this common language, I can argue that some EA applications, like symbolic regression [Koz92; SL09], are indeed ML: the fact that the optimization algorithm used is an EA, from this point of view, is just a detail.

Indeed, while ML and EC progressed almost independently in the past decades, the two fields are deeply linked. The *obvious connection* between the processes of learning and evolution has been pointed out by Alan Turing back in 1950 [TH50]. Even the term "machine learning" was coined in 1959 by Arthur Lee Samuel, a pioneer in the field of computational intelligence, whose attempts to devise a checkers player are frequently listed in EC history [Sam59]. Seminal works in EC explicitly refer to ML, far pre-dating the beginning of the current ML windfall [FPP86; Gol89; GH88; Gre93].

The connection between ML and EC led me to reframe some of the problems of the domain in a familiar perspective: for example, *overfitting*, the phenomenon by which a ML algorithm produces a model that fits the training data but does not generalize well, can be seen as an optimization algorithm missing the desired global optimum (a model able to generalize) and landing on an unsatisfying local optimum instead. The real-world applications I worked on since my post-doctoral experiences, and later on during my permanent position, mainly in the fields of food science [LPT16] and health [Lop+18; Lop+20], also contributed to complete my mindset on ML. Both domains feature high-dimensional problems for which relatively few data points are available, and having "good predictive models" is not enough; it is also necessary to find an explanation of the models' predictions, that a human expert can make sense of.

For these reasons I got interested in techniques that attempt to select the minimal information necessary for ML algorithms, to both make their conclusions human-readable, and help prevent overfitting [BST20b; BT20; Bar+20]. I also find the exact processes by which ML algorithms construct knowledge extremely intriguing, and I started investigating their mechanisms to understand whether it is possible to predict generalization capabilities of a ML model, starting from high-level dataset characteristics [BST20a].

As I will describe more in detail in Chapter 6 I plan to further the research lines I presented, by tackling more complex case studies in food science and health, and at the same time by addressing the more general algorithmic issues raised by these real-world problems. I am currently continuing my research line on the fundamental aspects of ML by developing new methods for detecting extrapolation in ML models (Section 6.1). The emerging challenge of bacteria resistant to antibiotics has all the features of a complex problem that could be tackled with ML and EAs, and this is what I am currently attempting to push towards to (Section 6.2). Finally, my expertise in EAs, and multi-objective optimization in particular, led to applying these techniques to problems in food science that inherently present conflicting objectives (Section 6.3).

1.3 Manuscript Overview

This section briefly summarizes the remaining chapters of the manuscript, acknowledging my many collaborators that made this work possible. As a computer scientist, I developed methodological improvements in the niches I am part of, machine learning and evolutionary computation. Nevertheless, the research lines I explored are motivated by practical needs that arose in the domains I applied ML and EA techniques to, mainly food science and healthcare. Both domains share some fundamental obstacles to the application of artificial intelligence and optimization techniques, such as relative scarcity of data and large feature spaces. I think that the two parts, one more theoretical and one more applied, inform each other, as I tried to summarize in Figure 1.1. Each chapter is structured to contain all necessary background information, so that it can be read independently from the others.

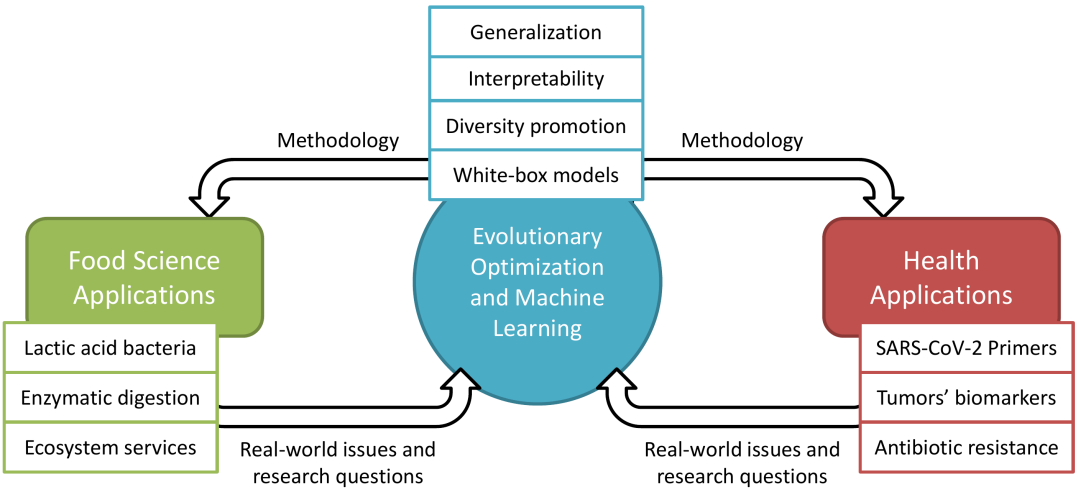


Figure 1.1: Conceptual scheme of the research works described in this manuscript.

Chapter 2 groups research lines on what can be termed the *epistemology* of ML algorithms, dealing with how ML algorithms acquire and use knowledge related to a given task. This work has been carried out in collaboration with my former Ph.D. supervisor, Giovanni Squillero from Politecnico di Torino, Italy, and Evelyne Lutton, INRAE, France. All the ideas have been developed through my co-supervision of a master student of Politecnico di Torino, Pietro Barbiero, who is now pursuing a Ph.D. at the University of Cambridge, UK, in the laboratory of Pietro Liò, Cambridge Center for AI in Medicine, UK. Our collaboration is still ongoing.

Chapter 3 describes my contributions to the algorithmic aspects of EC. Section 3.1 reports a review on the state of the art for diversity preservation and promotion, that I co-wrote with Giovanni Squillero. Section 3.2 summarizes the content of different publications, produced again in collaboration with Giovanni Squillero, through the shared co-supervision of a Ph.D. student at Politecnico di Torino, Marco Gaudesi. Finally, Section 3.3 reports a work that I developed from an initial idea of Maarten Keijzer, Pegasystems, The Netherlands, in collaboration with my colleagues from INRAE, Sébastien Gaucel and Evelyne Lutton.

Chapter 4 summarizes the food science applications I worked on since my recruitment at INRAE. The works have been carried out in collaboration with the colleagues of my institute, mainly Nadia Boukhelifa, Eric Dugat-Bony, Fernanda Fonseca, Sébastien Gaucel, Séverine Layec, Steven Le Feunteun, Evelyne Lutton, and Nathalie Méjean-Perrot, through the shared co-supervision of the Ph.D. students Etienne Deschamps and Thomas Chabin, AgroParisTech, France. For Section 4.3, describing the activities of COST Action FoodMC, a project I led with over 150 participants, there are simply too many people to acknowledge individually, so I will just extend a collective thanks.

Chapter 5 is an overview of my research on healthcare applications. These works have been achieved in collaboration with the Department of Pharmacology of Utrecht University, The Netherlands, and in particular with Alejandro Lopez-Rincon and Aletta Kraneveld. The results in Section 5.1 would have been impossible without the staff of Hospital Civil de Guadalajara "Dr. Juan I. Menchaca", Mexico, in particular Lucero Mendoza-Maldonado; and without the participation of Carmina Perez-Romero, Universidad Central de Queretaro, Mexico.

Chapter 6 concludes the manuscript, outlining the research projects I planned for the next part of my career, most of them derived from ideas contained in the previous chapters.

1.4 Paralipomena[2]

In compiling this manuscript, I had to necessarily perform a selection of my works, focusing not only on what I consider the most relevant, but also on those that are easier to weave together in a coherent narration. As a result, a considerable number of research lines were left out. I would like to briefly mention three of those, as even though they did not fit in this work, I still consider them remarkable professional and personal experiences.

Together with Doina Bucur, University of Twente, The Netherlands, Giovanni Iacca, University of Trento, Italy, and Giovanni Squillero, we tackled topics related to graphs, both for communication and social networks; for some of these works, I co-supervised the Ph.D. student Andrea Marcelli, Politecnico di Torino, Italy. We framed network security issues as optimization problems [Buc+13; Buc+14a; Buc+14b; Buc+15; Buc+16] and posed influence maximization in social networks as a multi-objective problem [Buc+17; Buc+18a; Buc+18b], describing compromises between amount of resources invested and global propagation, and developed an open-source package for influence maximization [Iac+21]. One of our contributions won the Best Paper Award at the EvoApplications conference in 2017.

Another research line I am particularly attached to is related to the development of AI for games. Early works, again stemming from my co-supervision of Ph.D. student Marco Gaudesi, dealt with learning the opponent's behavior for the iterated prisoner's dilemma [Gau+14; Gau+16b], a classic benchmark for game theory. More recent contributions, tackling the real-world games of StarCraft [Gar+15] and HearthStone [Gar+20; Gar+16; Gar+18], are the result of my collaboration with Pablo Garcia-Sanchez, Antonio Mora, and Juan Merelo from University of Granada, Spain.

A last topic I would like to mention is the use of EC and ML techniques to deal with malware. This was the main subject of Andrea Marcelli's thesis, a Ph.D. student I co-supervised with Giovanni Squillero. Together, we published several works on EAs applied to computer virus creation [Gau+15; Gau+16a] and on how to perform clustering on malware applications for mobile phone operative systems [Atz+18].

[2] From the Greek for *what has been left out*.

Bibliography

[Atz+18] Andrea Atzeni et al. “Countering Android Malware: A Scalable Semi-Supervised Approach for Family-Signature Generation”. In: *IEEE Access* 6 (2018), pages 59540–59556 (cited on page 11).

[BST20a] Pietro Barbiero, Giovanni Squillero, and Alberto Tonda. *Modeling Generalization in Machine Learning: A Methodological and Computational Study*. 2020. arXiv: 2006.15680 [cs.LG] (cited on page 9).

[BST20b] Pietro Barbiero, Giovanni Squillero, and Alberto Tonda. “Predictable Features Elimination: An Unsupervised Approach to Feature Selection”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2020, In print (cited on page 9).

[BT20] Pietro Barbiero and Alberto Tonda. “Making Sense of Economics Datasets with Evolutionary Coresets”. In: *Advances in Intelligent Systems and Computing*. Springer International Publishing, 2020, pages 162–170 (cited on page 9).

[Bar+20] Pietro Barbiero et al. “A Novel Outlook on Feature Selection as a Multi-objective Problem”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2020, pages 68–81 (cited on page 9).

[Bre01] Leo Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001), pages 5–32 (cited on page 9).

[Buc+13] Doina Bucur et al. “An Evolutionary Framework for Routing Protocol Analysis in Wireless Sensor Networks”. In: *Applications of Evolutionary Computation*. Springer Berlin Heidelberg, 2013, pages 1–11 (cited on page 11).

[Buc+14a] Doina Bucur et al. “The impact of topology on energy consumption for collection tree protocols: An experimental assessment through evolutionary computation”. In: *Applied Soft Computing* 16 (Mar. 2014), pages 210–222 (cited on page 11).

[Buc+14b] Doina Bucur et al. “The tradeoffs between data delivery ratio and energy costs in wireless sensor networks”. In: *Proceedings of the 2014 conference on Genetic and evolutionary computation - GECCO ’14*. ACM Press, 2014 (cited on page 11).

[Buc+15] Doina Bucur et al. “Black Holes and Revelations: Using Evolutionary Algorithms to Uncover Vulnerabilities in Disruption-Tolerant Networks”. In: *Applications of Evolutionary Computation*. Springer International Publishing, 2015, pages 29–41 (cited on page 11).

[Buc+16] Doina Bucur et al. “Optimizing groups of colluding strong attackers in mobile urban communication networks with evolutionary algorithms”. In: *Applied Soft Computing* 40 (Mar. 2016), pages 416–426 (cited on page 11).

[Buc+17] Doina Bucur et al. “Multi-objective Evolutionary Algorithms for Influence Maximization in Social Networks”. In: *Applications of Evolutionary Computation*. Springer International Publishing, 2017, pages 221–233 (cited on page 11).

[Buc+18a] Doina Bucur et al. “Evaluating surrogate models for multi-objective influence maximization in social networks”. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, July 2018 (cited on page 11).

[Buc+18b] Doina Bucur et al. “Improving Multi-objective Evolutionary Influence Maximization in Social Networks”. In: *Applications of Evolutionary Computation*. Springer International Publishing, 2018, pages 117–124 (cited on page 11).

[Car+11] S. Di Carlo et al. “Increasing pattern recognition accuracy for chemical sensing by evolutionary based drift compensation”. In: *Pattern Recognition Letters* 32.13 (Oct. 2011), pages 1594–1603 (cited on page 8).

[De 16] Kenneth De Jong. *Evolutionary Computation: A Unified Approach*. Cambridge, Massachusetts: Bradford Books, 2016 (cited on page 8).

[FPP86] J Doyne Farmer, Norman H Packard, and Alan S Perelson. “The Immune System, Adaptation, and Machine Learning”. In: *Physica D: Nonlinear Phenomena* 22.1-3 (1986), pages 187–204 (cited on page 9).

[Gan+10] Stefano Gandini et al. “A Framework for Automated Detection of Power-related Software Errors in Industrial Verification Processes”. In: *Journal of Electronic Testing* 26.6 (Nov. 2010), pages 689–697 (cited on page 8).

[Gar+20] Pablo García-ánchez et al. “Optimizing Hearthstone agents using an evolutionary algorithm”. In: *Knowledge-Based Systems* 188 (Jan. 2020), page 105032 (cited on page 11).

[Gar+15] Pablo Garcia-Sanchez et al. “Towards automatic StarCraft strategy generation using genetic programming”. In: *2015 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, Aug. 2015 (cited on page 11).

[Gar+16] Pablo Garcia-Sanchez et al. “Evolutionary deckbuilding in hearthstone”. In: *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, Sept. 2016 (cited on page 11).

[Gar+18] Pablo García-Sánchez et al. “Automated Playtesting in Collectible Card Games using Evolutionary Algorithms: A Case Study in HearthStone”. In: *Knowledge-Based Systems* 153 (Aug. 2018), pages 133–146 (cited on page 11).

[Gau+14] Marco Gaudesi et al. “TURAN: Evolving non-deterministic players for the iterated prisoner’s dilemma”. In: *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2014 (cited on page 11).

[Gau+15] Marco Gaudesi et al. “Malware Obfuscation through Evolutionary Packers”. In: *Proceedings of the Companion Publication of the 2015 on Genetic and Evolutionary Computation Conference - GECCO Companion ’15*. ACM Press, 2015 (cited on page 11).

[Gau+16a] Marco Gaudesi et al. “Challenging Anti-virus Through Evolutionary Malware Obfuscation”. In: *Applications of Evolutionary Computation*. Springer International Publishing, 2016, pages 149–162 (cited on page 11).

[Gau+16b] Marco Gaudesi et al. “Exploiting Evolutionary Modeling to Prevail in Iterated Prisoner’s Dilemma Tournaments”. In: *IEEE Transactions on Computational Intelligence and AI in Games* 8.3 (Sept. 2016), pages 288–300 (cited on page 11).

[Gol89] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Artificial Intelligence. Addison-Wesley Publishing Company, 1989. ISBN: 9780201157673 (cited on page 9).

[GH88] David E Goldberg and John H Holland. “Genetic Algorithms and Machine Learning”. In: *Machine learning* 3.2 (1988), pages 95–99 (cited on page 9).

[Gre93] John J Grefenstette. “Genetic Algorithms and Machine Learning”. In: *Proceedings of the sixth annual conference on Computational learning theory*. ACM. 1993, pages 3–4 (cited on page 9).

[Iac+21] Giovanni Iacca et al. “An evolutionary framework for maximizing influence propagation in social networks”. In: *Software Impacts* (July 2021), page 100107 (cited on page 11).

[Koz92] John R Koza. *Genetic Programming, On the Programming of Computers by Means of Natural Selection. A Bradford Book*. MIT Press, Cambridge, MA, USA, 1992 (cited on pages 8, 9).

[Lop+18] Alejandro Lopez-Rincon et al. “Ensemble Feature Selection and Meta-Analysis of Cancer miRNA Biomarkers”. In: *bioRxiv* (2018). eprint: `https://www.biorxiv.org/content/early/2018/06/21/353201.full.pdf` (cited on page 9).

[Lop+20] Alejandro Lopez-Rincon et al. “Classification and Specific Primer Design for Accurate Detection of SARS-CoV-2 Using Deep Learning”. In: *Scientific Reports* (2020) (cited on page 9).

[LPT16] Evelyne Lutton, Nathalie Perrot, and Alberto Tonda. *Evolutionary Algorithms for Food Science and Technology*. Wiley, Nov. 2016 (cited on page 9).

[Lut+14] Evelyne Lutton et al. “Food model exploration through evolutionary optimisation coupled with visualisation: Application to the prediction of a milk gel structure”. In: *Innovative Food Science & Emerging Technologies* 25 (Oct. 2014), pages 67–77 (cited on page 8).

[Pea96] Karl Pearson. “Mathematical Contributions to the Theory of Evolution. III. Regression, Heredity, and Panmixia”. In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 187 (1896), pages 253–318 (cited on page 9).

[Sam59] Arthur L Samuel. “Some studies in machine learning using the game of checkers”. In: *IBM Journal of research and development* 3.3 (1959), pages 210–229 (cited on page 9).

[SST12] Ernesto Sanchez, Giovanni Squillero, and Alberto Tonda. *Industrial Applications of Evolutionary Algorithms*. Springer Berlin Heidelberg, 2012 (cited on page 7).

[SL09] Michael Schmidt and Hod Lipson. “Distilling free-form natural laws from experimental data”. In: *science* 324.5923 (2009), pages 81–85 (cited on page 9).

[Sch65] Hans-Paul Schwefel. *Cybernetic Evolution as Strategy for Experimental Research in Fluid Mechanics (Diploma Thesis in German)*. Hermann F?ttinger-Institute for Fluid Mechanics, Technical University of Berlin, 1965 (cited on page 8).

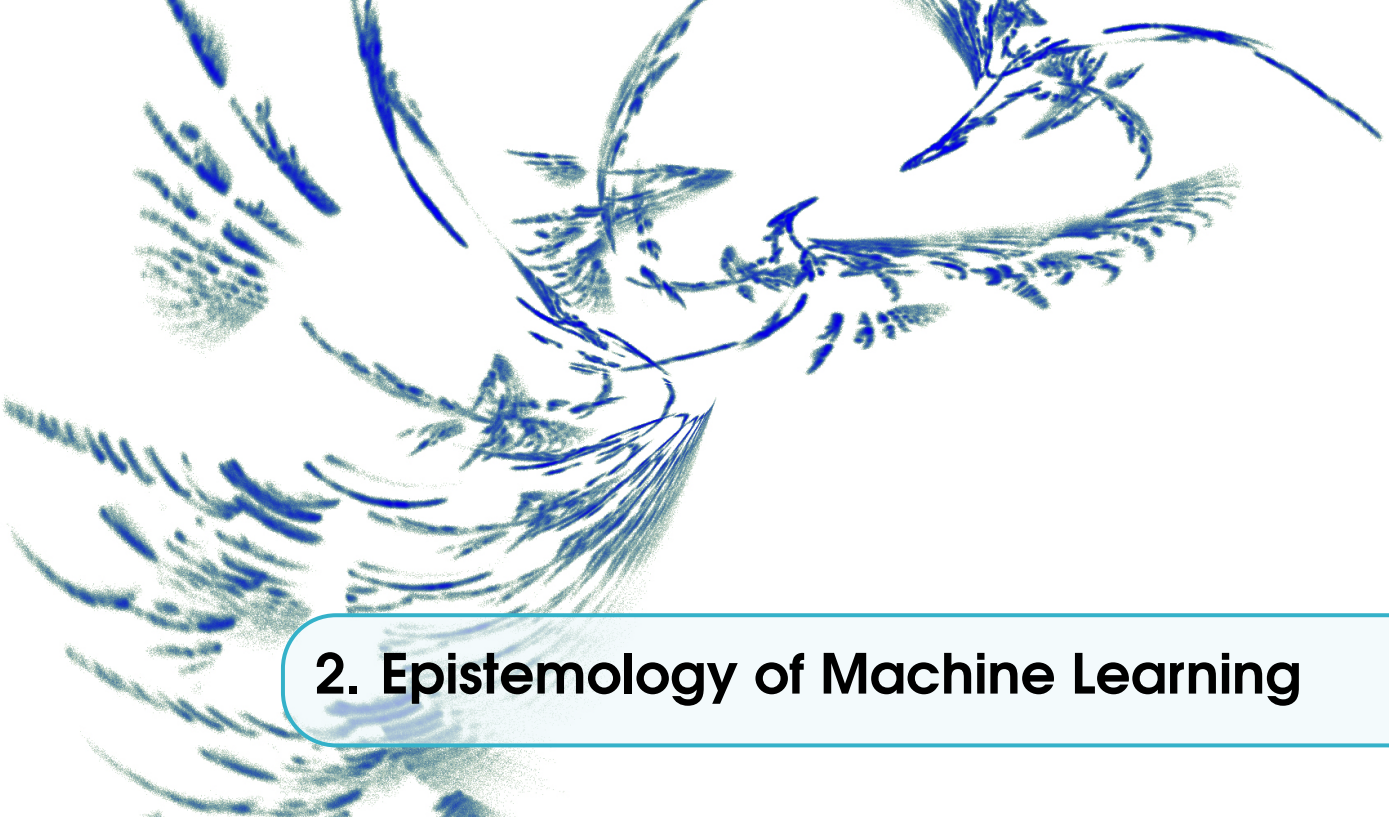
[ST08] Giovanni Squillero and Alberto Tonda. “A novel methodology for diversity preservation in evolutionary algorithms”. In: *Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation - GECCO ’08*. ACM Press, 2008 (cited on page 8).

[ST16] Giovanni Squillero and Alberto Tonda. “Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization”. In: *Information Sciences* 329 (Feb. 2016), pages 782–799 (cited on page 8).

[Ton+13] Alberto Tonda et al. “A Memetic Approach to Bayesian Network Structure Learning”. In: *Applications of Evolutionary Computation*. Springer Berlin Heidelberg, 2013, pages 102–111 (cited on page 8).

[Ton+12] Alberto Paolo Tonda et al. “Bayesian Network Structure Learning from Limited Datasets through Graph Evolution”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pages 254–265 (cited on page 8).

[TH50] Alan M Turing and J Haugeland. *Computing machinery and intelligence*. MIT Press Cambridge, MA, 1950 (cited on page 9).



2. Epistemology of Machine Learning

This chapter groups all activities related to more theoretical aspects of machine learning (ML), that I consider part of its *epistemology*, the theory dealing with how information is transformed into knowledge, and what are the limits of the learned models. Both questions become of utmost practical relevance when machines are tasked with taking critical decisions.

The first research line described in this chapter (Section 2.1) deals with predicting the capability of generalization of ML algorithms, starting from dataset characteristics, such as number of classes, number of features, and statistical metrics describing data distribution, a work currently under review but available on arXiv [BST20a]. The second part (Section 2.2) concerns several techniques for selecting a minimal, essential amount of information from larger datasets, at the level of features [BST20b; Bar+20] or at the level of samples [BST19; BT19; BT20]: information extracted in this way can be used to train ML algorithms and obtain both better generalization and more human-readable decisions. All activities described in the chapter are the result of my co-supervision of Politecnico di Torino's master student Pietro Barbiero (now Ph.D. candidate at University of Cambridge, UK).

2.1 Predicting Generalization

Among all common inquiries regarding ML, perhaps the most basic is: can ML work on a specific problem? Or, in other words: given the characteristics of a target data set, can the effectiveness of a ML approach be predicted? Interestingly, this latter question can be further rephrased as: what are the characteristics of a data set that are well correlated with the possibility, or the impossibility, of obtaining ML models able to effectively extrapolate to unknown instances of the problem? It is well known that ML algorithms are affected by the *curse of dimensionality* [AK18], but ML practitioners also know that it could be possible to obtain reliable models even for high-dimensional data sets, and with a relatively small number of samples [Bar+18]. The common approach among practitioners in the field, when dealing with a new data set, seems to be this one: try as many different ML algorithms as possible in a cross-validation, and evaluate the outcomes; then focus on the techniques that provided the best results, possibly applying them in an *ensemble* [AK17].

Taking inspiration from Oreski et al. [OOK17], where the authors find links between data-set characteristics and efficiency of feature selection techniques, it is here proposed to empirically explore the relation between data-set characteristics and effectiveness of standard ML classification models, to finally obtain a general meta-model able to extrapolate. In order to investigate the relationship between the considered metrics and extrapolation ability, 72 publicly available classification data sets from open-access, curated sources are analyzed. The focus of the experiments is on classification, as supervised ML represents a quite significant portion of real-world problems; and, differently from regression, several sophisticated quality metrics have already been developed for this task [Osi+17].

During the analysis, characteristics such as number of features, number of classes, number of samples, are taken into account, searching for correlations with quality metrics, such as accuracy of a ML model on training and test points. Extrapolation is assessed not just by alternatively dividing the data into training and test sets, but by analyzing whether data points fall inside or outside of the convex hull of the training data. After collecting the meta-data on the performance of state-of-the-art classification algorithms on the data sets, the statistical analysis presents both predictable and surprising results, hinting at the fact that dimensionality might not be so cursed after all.

2.1.1 The curses of dimensionality

As the concept of *curse of dimensionality* will be extensively referenced in the following, it is worth it to briefly summarize it in this Section. The *curse of dimensionality* denotes a variety of different phenomena that impair data analysis if a large number of variables need to be considered at the same time [Bel66; Bel15]. While in most cases it has no closed form nor a unique solution, it has distinctive deleterious effects. Problems like *data sparsity*, *collinearity*, and *overfitting* seem to confirm the platitude that in ML dimensionality is cursed [AK18].

As an example, let consider a collection of n data points generated by sampling two random variables X_1 and X_2 originated from two standard normal distributions ($\mu = 0, \sigma = 1.0$). The amount of samples falling within the interval $x_i \pm \sigma$ with $x_i = 1.5$ is around the 30% both for X_1 and X_2, individually (grey histograms in Figure 2.1). However, when the joint distribution of X_1 and X_2 is considered, the amount of samples falling within the joint interval $(x_1 \pm \sigma, x_2 \pm \sigma)$ (with $x_{1,2} = 1.5$) drops substantially. The worst-case scenario arises when the two random variables are uncorrelated (Figure 2.1, left). As the number of random variables (p) increases, the fraction of points within the p-dimensional interval of radius σ decreases rapidly: $\sim 5\%$ for $p = 2$, $\sim 1\%$ for $p = 3$, and $\sim 0\%$ for $p = 4$ [AK18]. In practice, the sparser the samples the harder will be collecting data that are representative of the population. The best-case scenario occurs when the random variables are perfectly correlated (Figure 2.1, right). In this case, the decrease is still significant, but much slower (from 30% to 20% rather than 5% for $p = 2$).

In most real-world scenarios, a considerable number of variables are correlated. As supervised ML algorithms are likely to favor variables correlated with the target variable [Hal00; YL03], the data-sparsity problem may be usually mitigated by considering together highly correlated sets of variables. However, exploiting correlated variables may also have a catastrophic impact when variables are used for prediction: as there could be more than one subset of variables yielding approximately the same result, considering them together can make it impossible to understand the individual impact of each variable. For example, suppose that two variables X_1 and X_2 can be used to predict a target variable Y by means of a predictive model f:

$$Y = f(X_1, X_2) \tag{2.1}$$

Now suppose that there exists another variable X_3 that can be expressed as a function of both X_1 and X_2, e.g., $X_3 = X_1 + 2X_2$. Such a system of equations can be solved by using an arbitrary pair of variables $\{(X_1, X_2), (X_1, X_3), (X_2, X_3)\}$, as they are all perfectly correlated. Even though the predictive problem appears to be solved, the causative source of variability of Y is now uncertain, and the relative importance of the variables cannot be estimated from data. The situation gets critical when the number of variables exceeds the number of samples ($p > n$): at least one of the variables can always be expressed as a linear combination of the others, thus yielding multiple perfect correlations [ZSK12].

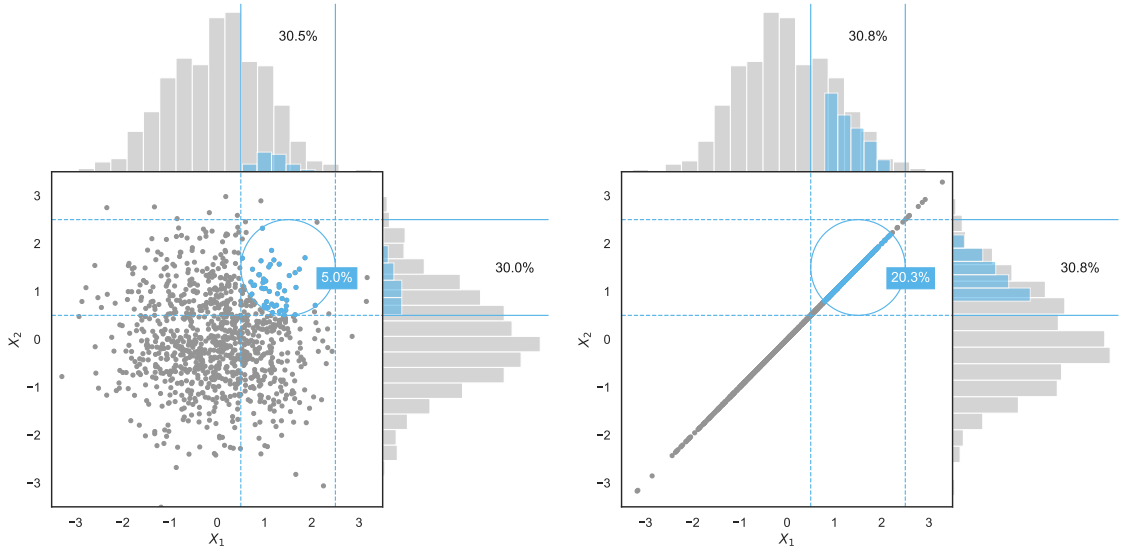


Figure 2.1: Visual representation of the data sparsity phenomenon in case of correlated random variables (left) and collinear random variables (right).

In the case of multicollinearity, one of the possible solutions exploited by supervised ML algorithms (e.g., logistic regression [MZ75]) is to associate a weight to each variable, corresponding to its relevance. Instead of discarding variables which may be the true causative source of variability, these approaches make it possible to take into account all the observed variables at the same time, by increasing the number of model's parameters. Such increase in model complexity may in turn be a possible cause of overfitting, another phenomenon related to the curse of dimensionality. In fact, the increased flexibility makes the model not only able to fit the underlying relationship between variables, but also the random idiosyncrasies of the observations [LKA16b].

2.1.2 Extrapolation and interpolation

The objective of supervised ML can be roughly summarized as automatically obtaining a *predictive model* for a specific phenomenon starting from a set of known cases. Usually such known cases consist in different instances of measurements, or *samples*, each one specifying the values of all different variables, or *features* of the problem. One of these features is the *target variable*, that is, what should be predicted by the final model. ML algorithms try to find a mathematical relationship between the target feature and the others in the set of known data, or *training set* of data. Eventually, such relation can be encoded with a linear model [LKA16a], more complex structures [LBH15], or even ensembles of simpler models [AK17; KA17]. Indeed, it is quite important to assess to what

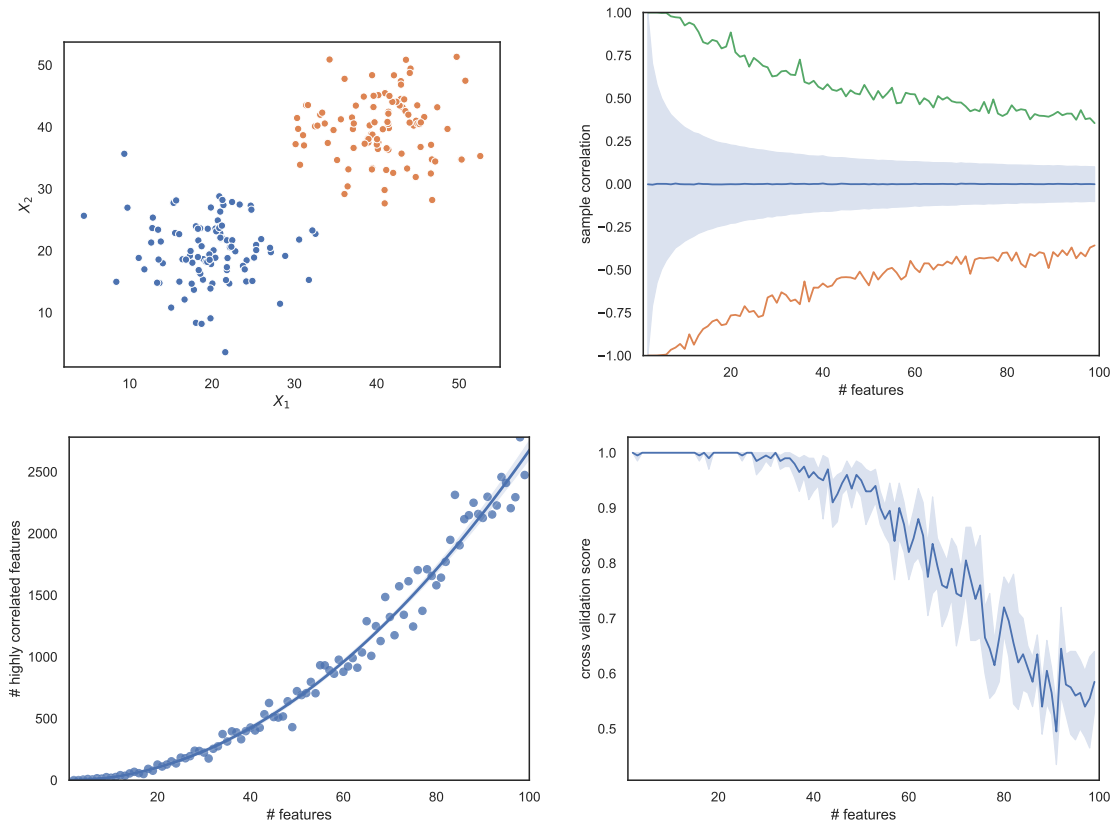


Figure 2.2: Two-class samples drawn from two correlated Gaussian random variables. Notably the classification problem is trivial as the two classes are linearly separable (top-left). By increasing the number of correlated random variables the highest value (in green), the lowest value (in orange), and the variance of samples' correlation shrink towards zero (top-right). The number of highly correlated variables ($\rho > 0.8$) increases polynomially with the overall number of features (bottom-left). On the other hand, the generalization accuracy of logistic regression in predicting class labels decreases, down to the point of providing almost random estimates (bottom-right).

extent the model obtained is able to *generalize*, that is, to provide meaningful predictions for new instances of the problem [Ney+17; Raj+20; RR16]. A model that is only able to deal with the data it was trained on is said to be *overfitted* [LKA16b], and it is generally considered useless regardless its performance on the train set.

In ML the term *capacity* may describe the ability of a model to represent complex relationships — the term *complexity*, when referred to a model, can also be used with a similar meaning. While it sounds obvious that a second-degree polynomial has a higher capacity than a linear regression, and can better fit more instances of data, there are relatively few contributions in literature that attempt to provide a more formal framing [BM02; Pog+04; VC15], and these terms are often used in rather intuitive and qualitative statements. As a fast but crude approximation, ML practitioners often assess model capacity by evaluating the number of parameters that can be tuned inside a model. In theory, the best ML model for a task is the one with just enough capacity to properly represent the training data: models with lower capacity would *underfit*, i.e., they deliver unsatisfying results as they are

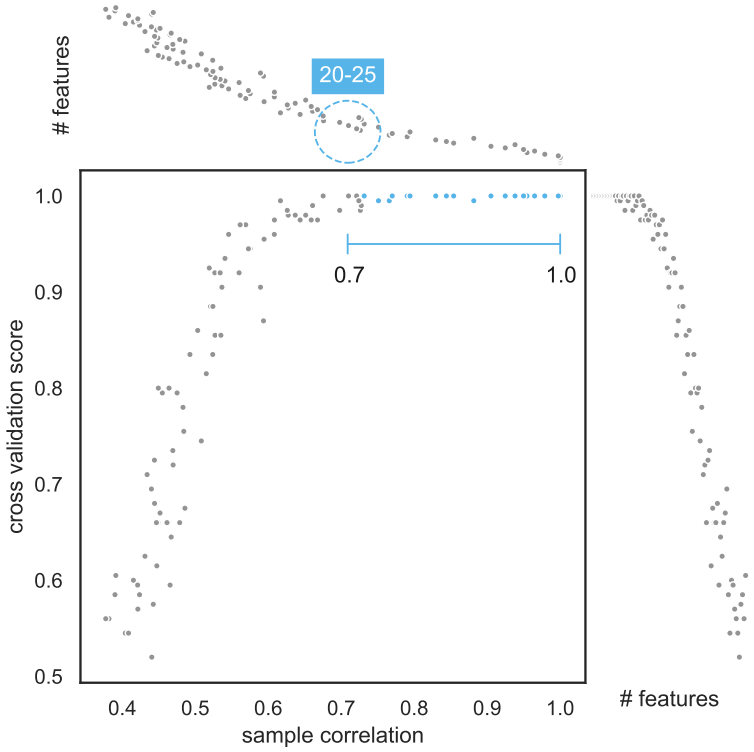


Figure 2.3: Mutual relationships among generalization ability (cross-validation accuracy), sample correlation, and number of features when all variables are correlated and gaussian. When the number of features is higher than 20–25, samples' correlation drops leading to a dramatic decrease in generalization ability. The top part of the figure shows sample correlation on the x axis, and number of features on the y axis; the right-most part describes the relationship between number of features on the x axis and cross-validation accuracy on the y axis.

unable to cope with the complexity of the phenomenon; models with higher capacity would risk to *overfit* and consequently generalize poorly.

A simple depiction of overfitting and underfitting is provided in Figure 2.4. In practice, unfortunately, it's extremely hard to estimate the capacity necessary to correctly represent a data set; and the solution that many ML practitioners use is — yet again — to apply several techniques with increasing capacity to the data set, until either the gain in fitting stops, or the improvements are not considered important enough to justify an increase in capacity. More interestingly, even estimating effective model capacity is not trivial, as there is evidence from works on deep learning that models with enough parameters to theoretically overfit the training data are actually able to generalize well in real-world case studies [Zha+16]. The trade-off between fitting and capacity has been independently explored by different ML communities, with the definition of model-dependent metrics that attempt to take into account both fitting and capacity to assess overall quality, to facilitate model selection: A few examples include the Akaike Information Criterion [Aka74], the Bayesian Information Criterion [Sch+78], and Pareto-based approaches used mainly in symbolic regression [SK05].

While evaluating model capacity can help reducing the chance of overfitting, *measuring* over-

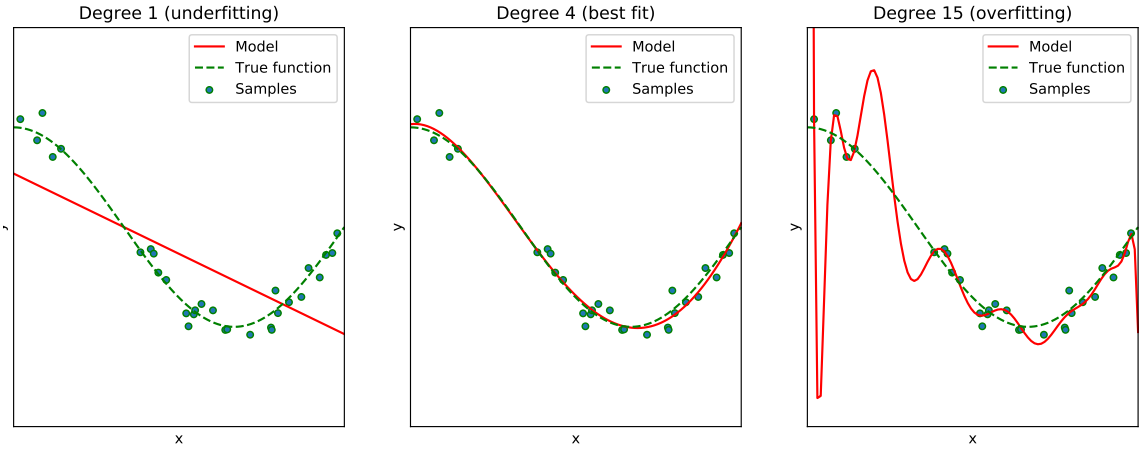


Figure 2.4: Visual representation of the effect of model capacity on fitting. The original data points can be properly represented by a polynomial of degree 4 (**middle**), so a polynomial regression with lower capacity will underfit (**left**), while a polynomial regression with higher capacity will overfit (**right**).

fitting remains far from trivial. Ideal ML models should be able to obtain good predictions even for *unknown* samples of the same problems, but - by definition - the models cannot be tested on unknown data sets. Given this practical need, ML researchers found ways to at least *assess* overfitting, through different techniques. A basic, but extremely efficient technique, is cross-validation (with all its variations, such as leave-one-out cross-validation, stratified cross-validation and the like): the training data is split into k folds of equal size, a ML algorithm is iteratively trained on all folds minus one, and tested on the remaining fold [Sto74]. Analyzing the results of a k-fold cross-validation, for example the average performance, or single instances where the performance on a test fold differs greatly from the others, can provide further insight on the problem characteristics.

As the real objective of evaluating overfitting is to assess a model's capability of extrapolating to unknown instances of the same problem, it is worth it to spend a few words on the meaning of *extrapolation* in supervised ML. As for model capacity, there is an intuitive and imprecise concept of extrapolation, defined as the ability of the model to correctly predict data points that are *considerably different* from the information provided in the training data, but still belong to the same problem. An alternative outlook on extrapolation comes from computational geometry: Interpreting the data points in the training set as points in $\mathbb{R}^f$, where f is the number of features, it is possible to compute its *convex hull*, the smallest polytope that contains all training points. Given the convex hull of the training set, it is then possible to assess whether an unseen test data point will fall inside or outside the convex hull. It is reasonable to assume that, for points inside the convex hull, a ML model will *interpolate* known data to obtain a prediction; while the same model will *extrapolate* for test points placed outside of the convex hull. An example is presented in Figure 2.5. It is important to notice that, depending on the characteristics and the distribution of the training points, this interpretation of interpolation/extrapolation might not correspond to the actual difficulty of predictions for the model. For example, it is possible to imagine a situation where the model will provide better predictions for of a test point outside of the convex hull of the training data, but still relatively close to known

points, than for a test point located inside the convex hull of the training data, but in a part of the space where training points are relatively sparse. Still, in most practical scenarios, it is generally harder for models to reliably predict values for test points outside of the convex hull of the training data. A more in-depth discussion on the convex hull is provided in the following Section.

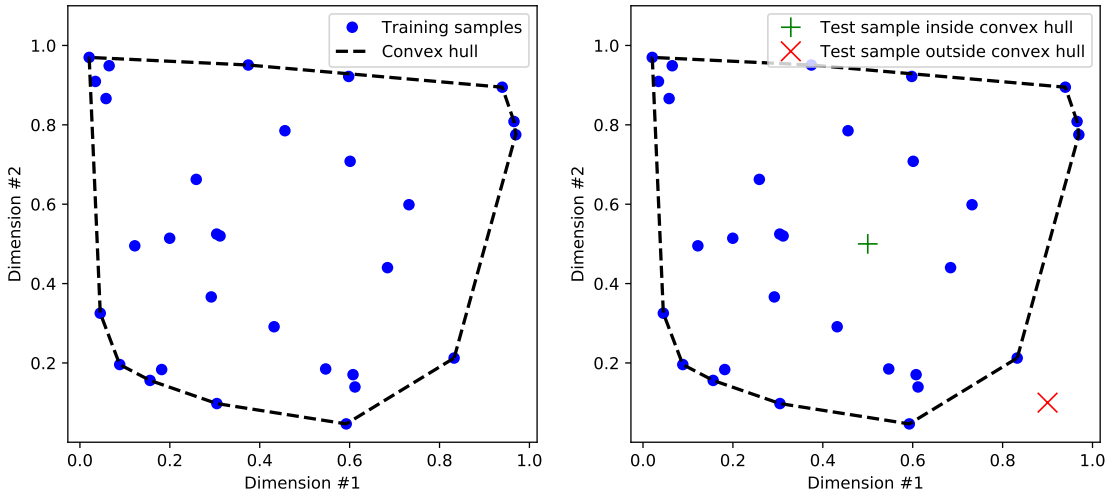


Figure 2.5: An example of convex hull. The convex hull of a set of training samples is the minimal hyper-polygon (in this case, a 2D polygon) that contains all the data inside its edges (left). For a machine learning model trained on the initial training set, predicting a value for an unseen test point inside the convex hull (in green) probably requires interpolation; but predicting for a test point outside the convex hull requires extrapolation (right).

2.1.3 Assessing generalization with the convex hull

In an Euclidean space, the convex hull of a set of points $X = \{x_i \in \mathbb{R}^d\}$ is the smallest convex set containing all the points in X. If the number of points n in X is finite (i.e. if X is a matrix $\mathbb{R}^{n \times d}$), then the convex hull forms a convex polytope in $\mathbb{R}^d$. Finding faces or the set of extreme points of this convex polytope is a NP-hard problem [Tiw08]. However, checking if a point z lies inside or outside the polytope is much easier, and can be performed in polynomial time [ST04]. When dealing with ML models, evaluating the convex hull of a training set can provide extra information on unseen test points: if a test point falls inside the convex hull, it is expected that the ML model will probably *interpolate* known points to find the predicted value; vice versa, if a test point falls outside the convex hull of the training set, the ML model will likely be required to *extrapolate* to obtain a prediction.

The problem of checking whether a point z lies inside the polytope of X has a simple linear programming formulation [PLH95]:

$$\begin{aligned} \min_{y} \quad & c^T y \\ \text{s.t.} \quad & Ay = b \\ & c, y \in \mathbb{R}^n. \end{aligned} \tag{2.2}$$

where:

$$c = 0 \qquad A = \begin{bmatrix} X^T \\ 1^T \end{bmatrix} \qquad b = \begin{bmatrix} z \\ 1 \end{bmatrix} \tag{2.3}$$

Such formulation is known as a *Phase I* method [BV04], as the final goal is not the actual optimization of variable z, but rather checking whether a feasible solution does exist. In such contexts, the cost function can be a constant, as the only objective is satisfying the constraints. The first n constraints impose that the position of z in the feature space must be a combination of the points X:

$$z = \sum_{i=1}^{n} y_i x_i \tag{2.4}$$

The last constraint imposes that such combination must be convex, which implies, by definition, that the coefficients y_i must sum to 1:

$$\sum_{i=1}^{n} y_i = 1 \tag{2.5}$$

If the Phase I problem is feasible, point x can be expressed as a convex combination of the set of points X. By definition, this means that point z lies inside the convex polytope of X. On the contrary, if the Phase I problem does not have any feasible solution, then point z lies outside the convex hull of X.

Even the proposed approach exploiting the convex hull can be affected by the curse of dimensionality. The critical point occurs when the dimension of the Euclidean space exceeds the number of observations ($d > n$). In this case the upper bound of the rank of the matrix X corresponds to the number of samples n [Mac95; Mir12]:

$$rank(X) \leq n \tag{2.6}$$

Therefore, the maximal number of linearly independent columns of X cannot be higher than n. Independently from the number of dimensions d, the points x_i will lie in a subspace $\mathbb{R}^s$ where $s \leq rank(X) \leq n < d$. As a consequence, the convex hull generated by the set of points x_i will belong to the same subspace. By definition, for $s < d$, the subspace $\mathbb{R}^s$ has *measure zero* in $\mathbb{R}^d$ and can be considered as negligible [Fol13]. Hence, when a new point z is added to the space, it is *almost sure* that it will fall outside negligible subspaces as $\mathbb{R}^s$ [JP12]. In summary, when $d > n$, the new point z will *almost never* belong to the convex hull of X, making the computation of the convex hull ultimately useless.

2.1.4 Modeling generalization as a function of data set characteristics

The objective of this work consists in assessing generalization abilities of ML classification models with empirical experiments. To this aim, it is necessary to first select (i) a large set of publicly available data sets for classification, (ii) a representative set of machine learning classifiers, and (iii) a set of data set characteristics. Then, on each data set, a cross-validation using ML models is performed, and the relevant metrics for each fold are computed. Finally, the relationships between data set characteristics and generalization ability of ML models are analyzed, using both classical correlation metrics, and association models derived through symbolic regression.

Data sets

Following the analyses presented in [OOK17] and [Mic+94], this study focuses on classification, as it is easier to characterize classification rather than regression data sets, as several of the characteristics analyzed are based on comparing batches of samples belonging to different classes (see the *Statistical metrics* paragraph in subsection 2.1.4). Additionally, as the following analysis is partly based on evaluating convex hulls, only data sets with real-valued features are considered.

The data sets examined are acquired from the OpenML repository [Van+13], an online, curated collection that, as the time of writing, includes over 3,100 data sets of different kinds. After selecting only data sets related to classification problems, with real-valued features exclusively, and discarding those containing errors, a total of 72 data sets was ultimately retained for the analysis. The number of samples in the selected collection ranges from 47 to $44,819$, while the number of features spans from 2 to $3,758$. All selected data sets have real-valued features and a discrete target (suited for classification). The mean feature correlation of the data sets is 0.619, with a standard error of the mean of 0.007, the average intrinsic dimensionality ratio is 0.629, with a standard error of the mean of 0.047.

Data set characterization

As in previous meta-analyses [Mic+94; OOK17], each data set is characterized using metrics, grouped into four categories: simple, statistical, Euclidean, and generalization metrics. For each data set, such measures are computed over a stratified 10-fold cross validation [Sto74].

Simple metrics

Simple metrics describe general characteristics of data sets, namely number of samples (n), number of features (d, a.k.a. *dimensionality*), and number of classes (c) [DKA06].

Statistical metrics

Statistical metrics assess (i) class differences in feature distributions and shapes, and (ii) relationships between features and classification target.

Levene's test [Lev60] is an inferential statistical test used to assess if a vector of random variables is *homoscedastic*, i.e. if the variance of the random variables is *almost equal*. In the following, Levene's test is used to compare class covariances for each data set feature. The lower the p-value, the higher will be the probability that the class covariances of the feature under study are different [AKR17]. In the following experiments, the score λ collected for each data set is the average of the Levene's p-values of its features.

Pearson's correlation coefficient [Pea20] measures the linear relationship between two variables, providing an indication of the interdependence between pairs of features. The correlations between all pairs of attributes are calculated for each class separately. Since the objective is to evaluate the strength of the relationship and not its sign (positive or negative), the absolute value of the coefficient is used. For each data set, the collected score ρ is the average of the coefficient over all pairs of features and over all classes.

Skewness [Pea05b] corresponds to the third standardized moment of a random variable. It indicates the magnitude of the asymmetry of a feature around its mean, yielding an estimate of the feature's departure from normality. The skewness for a class is computed as a weighted average of the skewness of the feature values of its samples. The final skewness score γ represents the average skewness over all classes.

Kurtosis [Pea05a] corresponds to the fourth standardized moment of a random variable. It indicates the "thickness" of the tails of a density function. Distributions with kurtosis less than 3

Table 2.1: Summary of the metrics used to characterize the data sets analyzed in the study.

Metric type	Symbol	Description
Standard metrics	n	Number of samples
	d	Number of features
	c	Number of classes
Euclidean metrics	$\mathfrak{I}$	Intrinsic dimensionality
	$\mathfrak{I}_r$	Intrinsic dimensionality ratio
	$\mathfrak{N}$	Feature noise $(1 - \mathfrak{I}_r)$
	$\mu_{\mathfrak{D}}$	Average sample distance
	$\sigma_{\mathfrak{D}}$	Standard deviation of sample distance
Statistical metrics	λ	Average of Levene's test p-values
	ρ	Average of class-wise feature correlation
	γ	Average of class-wise feature skewness
	κ	Average of class-wise feature kurtosis
	η	Average of feature-target mutual information
Generalization metrics	CI_{train}	Class-imbalance of training samples
	CI_{test}	Class-imbalance of test samples
	T_{in}	Ratio of test samples inside the convex hull
	T_{out}	Ratio of test samples outside the convex hull $(1 - T_{in})$
	CI_{in}	Class-imbalance of test samples inside the convex hull
	CI_{out}	Class-imbalance of test samples outside the convex hull
	$F1_{train}$	$F1$ for the training set
	$F1_{test}$	$F1$ for the whole test set
	$F1_{in}$	$F1$ for the part of the test set inside the convex hull (interpolation ability)
	$F1_{out}$	$F1$ for the part of test set outside the convex hull (extrapolation ability)

are called *platykurtic*, i.e. they produce fewer and less extreme outliers than the normal distribution. Inversely, distributions with kurtosis higher than 3 are called *leptokurtic* and produce more outliers with respect to the normal distribution. The kurtosis for a class is computed as a weighted average of the kurtosis of the feature values of its samples. The final kurtosis score κ represents the average kurtosis over all classes.

Mutual information [KL87; Ros14] measures the mutual dependence between two variables. In the following experiments, it is used to estimate the amount of information obtained about the classification target by observing a data set feature. The overall mutual information score η is computed as the average over all features.

Euclidean metrics

Euclidean metrics assess the shape of the data manifold.

The *intrinsic dimensionality ratio* provides a normalized estimate of the dimensionality of the data, considering a linear manifold. It is computed counting the number of principal components

needed to explain 90% of the variance in the target [Jol11] ($\mathfrak{I}$). The final score $\mathfrak{I}_r$ is normalized over the number of original features .

Feature noise is an estimate of the amount of information that is useless with respect to the classification task. Following [OOK17] and [Lóp+13], the score is computed through the difference between dimensionality (the original number of features) and intrinsic dimensionality. The final score $\mathfrak{N}$ is normalized over the original dimensionality.

The *average sample distance* $\mu_{\mathfrak{D}}$ and the *standard deviation of sample distances* $\sigma_{\mathfrak{D}}$ [AR10] measure the average pairwise distance between two data set points, and the standard deviation of the resulting distribution, respectively.

Generalization metrics

Generalization metrics estimate the nature and hardness of the classification task. Given the convex hull of a training data set, the ratio of test points inside it T_{in} and outside of it T_{out} assesses the type of generalization task ML models are asked to perform. Indeed, if test points often fall inside the convex hull of the training set, it is expected that the ML model will probably *interpolate* known points to find predicted values, most of the time; vice versa, if test points frequently fall outside the convex hull, the ML model will likely be required to recurrently *extrapolate* to obtain predictions. Class-imbalance may also play a role in impairing classifier performance. It has been computed for training and test samples, both inside and outiside the convex hull.

Classification performance estimates how difficult it is for a given classifier to learn from the training set and generalize to the test set. Since the analyzed data sets usually have more than two classes, the $F1$ score [Van79] is used to measure the classification performance. $F1$ is a measure of classification accuracy, the harmonic mean of the precision and recall. More specifically, the weighted $F1$ score is adopted, to account for label imbalance. Two scores related to the test set are computed, assessing effectiveness in both interpolation ($F1_{in}$) and extrapolation ($F1_{out}$).

2.1.5 Experimental results and discussion

This section describes the experimental results obtained through the analysis of the selected data sets. First, classical linear correlations between the chosen metrics are considered. Then, more complex non-linear models are explored and discussed, outlining the importance of the convex hull. While different algorithms might perform differently on the same data set, testing all possible classifying alternatives is impractical. For this purpose, a *Logistic Regression* (LR) [YHL11] classifier, a *Support Vector Machines* classifier with radial basis function kernel (SVC) [Pla+99], and a *Random Forest* classifier with 100 decision tree estimators (RF) [Bre01a; LW+02] are selected as representative classifiers for the following experiments, taking into account their considerable efficiency and their heterogeneous capacity.

All the code and data necessary to reproduce the experiments is available in a public GitHub repository[1]. The experiments took 5 months of computational time on a server with 64 Intel®Xeon®E7-4830 2.13GHz CPUs, and 128 GB of RAM. The choice of the machine learning classifiers used in this study is also tied to the availability of processing power: while testing other algorithms such as deep neural networks would have been informative, the longer training time would have made the experiments impractical.

[1] `https://github.com/pietrobarbiero/dataset-characteristics`

Correlations between data set characteristics

Once all the considered metrics described in Sec. 2.1.4 have been computed for the 72selected data sets, classical statistical correlations can be evaluated. In particular, computing Pearson's correlation coefficients results in the matrix presented in Fig. 2.6. Analyzing the matrix, several predictable correlations can be found: In the following, a few of the least immediately obvious will analyzed more in depth.

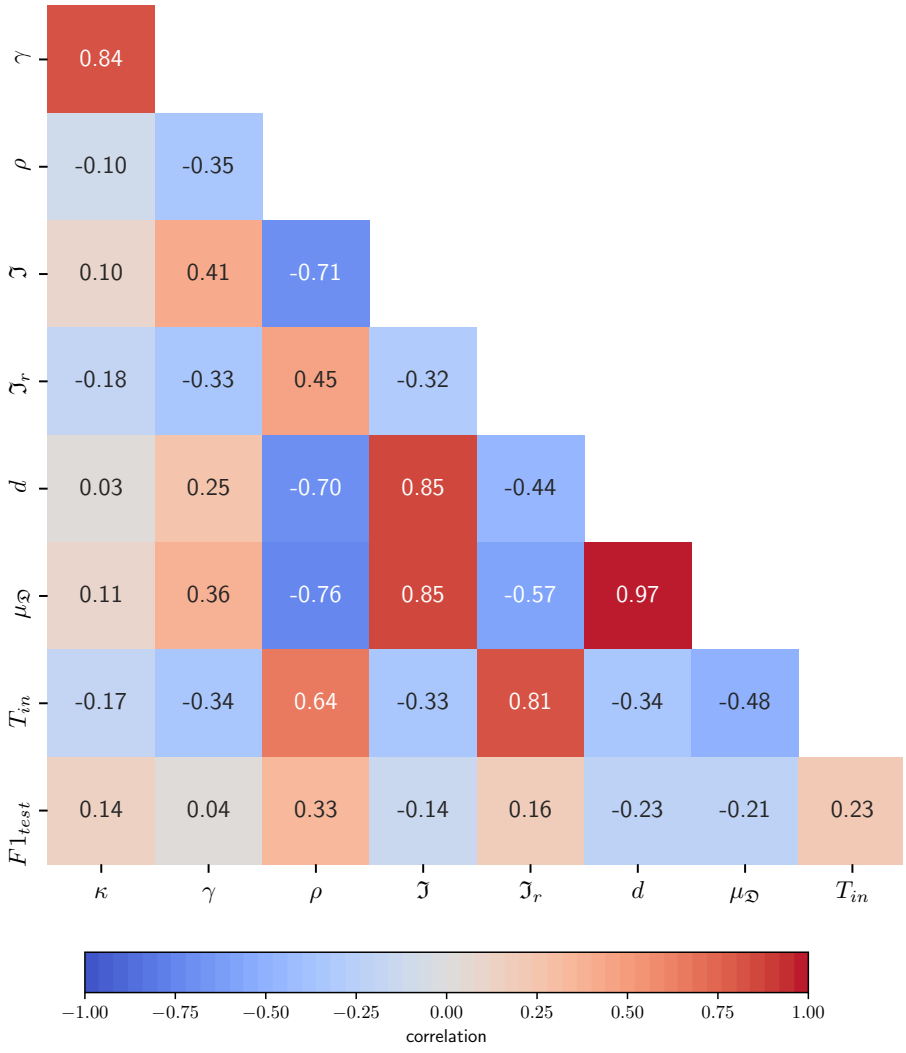


Figure 2.6: Heatmap showing a selection of the most relevant correlations between data set characteristics. $F1_{test}$, the F1 score computed on test samples corresponds to the average score over the three ML models used in the experiments, i.e. LR, SVC, and RF.

$\gamma \diamond \kappa$: the mean skewness and mean kurtosis of a dataset are highly correlated (0.84), as both metrics assess the difference in data distribution with respect to a reference Gaussian distribution.

$\mathfrak{J} \diamond \rho$: very often, the higher the correlation between features, the lower the intrinsic dimensionality of a dataset, so as expected the two metrics are anti-correlated (-0.71).

d: dimensionality is positively correlated with $\mathfrak{I}$ (0.85) and negatively correlated with ρ (-0.70). In other words, as the number of features increases, intrinsic dimensionality tends to rise; and, at the same time, features are less likely to be strongly correlated with each other.

$\mu_{\mathfrak{D}} \diamond \rho$: as the number of features/dimensions increases, the average distance between samples also increases, *unless* the additional features are strongly correlated with existing ones. For this reason, as expected, the average distance between samples is negatively correlated with the average correlation between features (-0.76). For the same reason, a 0.85 positive correlation is found between average sample distance and intrinsic dimensionality ($\mathfrak{I}$), and an equally strong positive correlation between $\mu_{\mathfrak{D}}$ and number of features d.

$T_{in} \diamond \mathfrak{I}_r$: the correlation between the ratio of test samples falling inside the convex hull of the training set and the intrinsic dimensionality ratio (0.81) is maybe the less intuitive of the relationships analyzed so far. When the intrinsic dimensionality $\mathfrak{I}$ is lower than the number of dimensions d, training points used to compute the convex hull might actually all lie in a polytope of dimension $d' \approx \mathfrak{I} < d$; the likelihood of test points laying exactly inside this polytope becomes then low, especially when compared to a situation where $\mathfrak{I} \approx d$, and the convex hull of the training set occupies a much larger portion of the feature space. The very same concept is also expressed by the negative correlation between T_{in} and $\mathfrak{N}$ (-0.81), as a higher feature noise also represents a lower effective dimensionality. The same reasoning holds for the correlations $T_{out} - \mathfrak{I}_r$ and $T_{out} - \mathfrak{N}$.

Regarding classifier-specific metrics, such as $F1$ for training and validation, it is possible to observe how LR presents the highest correlation between the two, suggesting a similar behavior for training and unseen samples. SVC and RF, on the other hand, have poorer correlations, as they tend to overfit the training set more, as expected by classifiers with higher capacity. It is important to notice that the strength of this correlation does not imply a poor performance, as RF, with the lowest correlation, shows the best $F1$ on the test set (see Table 2.2).

Correlations that are unexpectedly weak in the analysis, those between $F1_{test}$ and all metrics related to dimensionality (d, $\mathfrak{I}$, $\mathfrak{I}_r$), hint at a surprising conclusion: the performance of the ML algorithms on an unseen test set is almost independent from the dimensionality of the data set. This is particularly true for LR ($F1_{test} \diamond \mathfrak{I}_r = 0.1$) and RF ($F1_{test} \diamond \mathfrak{I}_r = 0.9$), while corresponding correlations for SVC are higher, but still not very strong ($F1_{test} \diamond d = -0.43$, $F1_{test} \diamond \mathfrak{I} = -0.37$, but $F1_{test} \diamond \mathfrak{I}_r = 0.28$).

For the complete correlation matrices between all considered characteristics, see Figs. 2.7, 2.8, 2.9. In addition to the metrics described in Subsection 2.1.4 Data set characterization, the correlation matrices reported below also include: A_{train} (classification accuracy on training set), A_{test} (classification accuracy on test set), A_{in} (classification accuracy on test samples contained inside the convex hull of the training set), A_{out} (classification accuracy on test samples falling outside the convex hull of the training set), d/n (ratio between number of features and number of samples of the data set), and $\mathfrak{I}/n$ (ratio between intrinsic dimensionality and number of samples of the data set).

The Key Role of the Convex Hull

In Table 2.2 the $F1$-scores of ML models on training and validation sets are reported. A high difference between $F1_{train}$ and $F1_{test}$ corresponds to overfitting. Interestingly, RF exhibits the highest overfitting, while still providing the best generalization performance during validation. On the other hand, the difference between $|F1_{train} - F1_{in}|$ and $|F1_{train} - F1_{out}|$ reveals the discrepancy between interpolation and extrapolation performances, pinpointing the importance of the convex hull in assessing machine learning generalization.

While classical correlation metrics try to optimize the coefficients of models of known structure

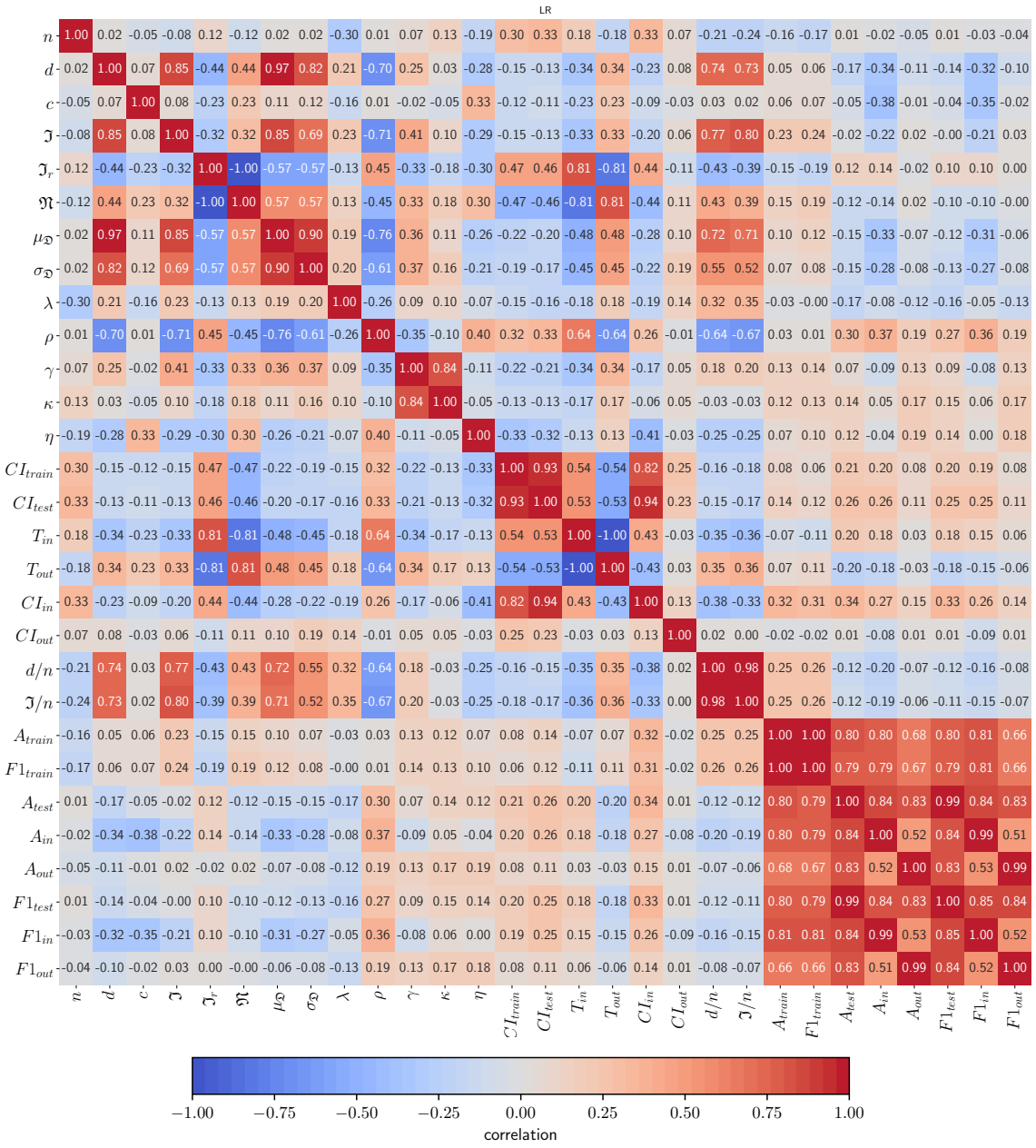


Figure 2.7: Correlations between data-set characteristics using Logistic Regression.

(e.g., often linear), it might be useful to extend such analysis to models of different structure. In statistical terms, this implies assessing *association*, a relationship between variables more general than correlation: two or more variables are *associated* if the values of some provide information on the value of the others [AK15]. To this purpose, the use of *symbolic regression* [Koz92; SL09] is proposed: symbolic regression is a technique that searches the space of mathematical expressions to find the model that best fits a given data set. The irony of using machine learning techniques to analyze the results of a meta-analysis on machine learning is not lost on the authors, but symbolic

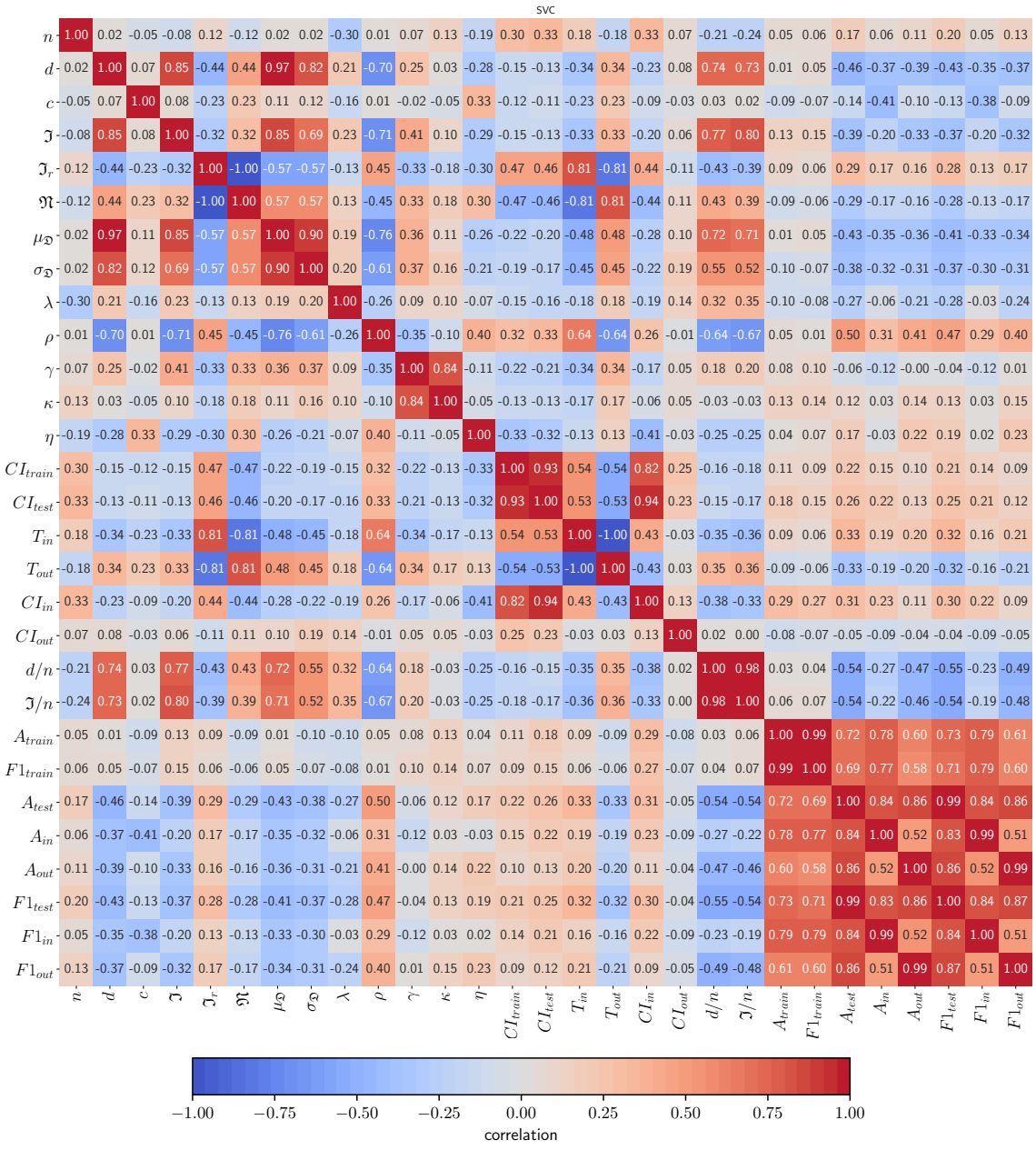


Figure 2.8: Correlations between data-set characteristics using SVC.

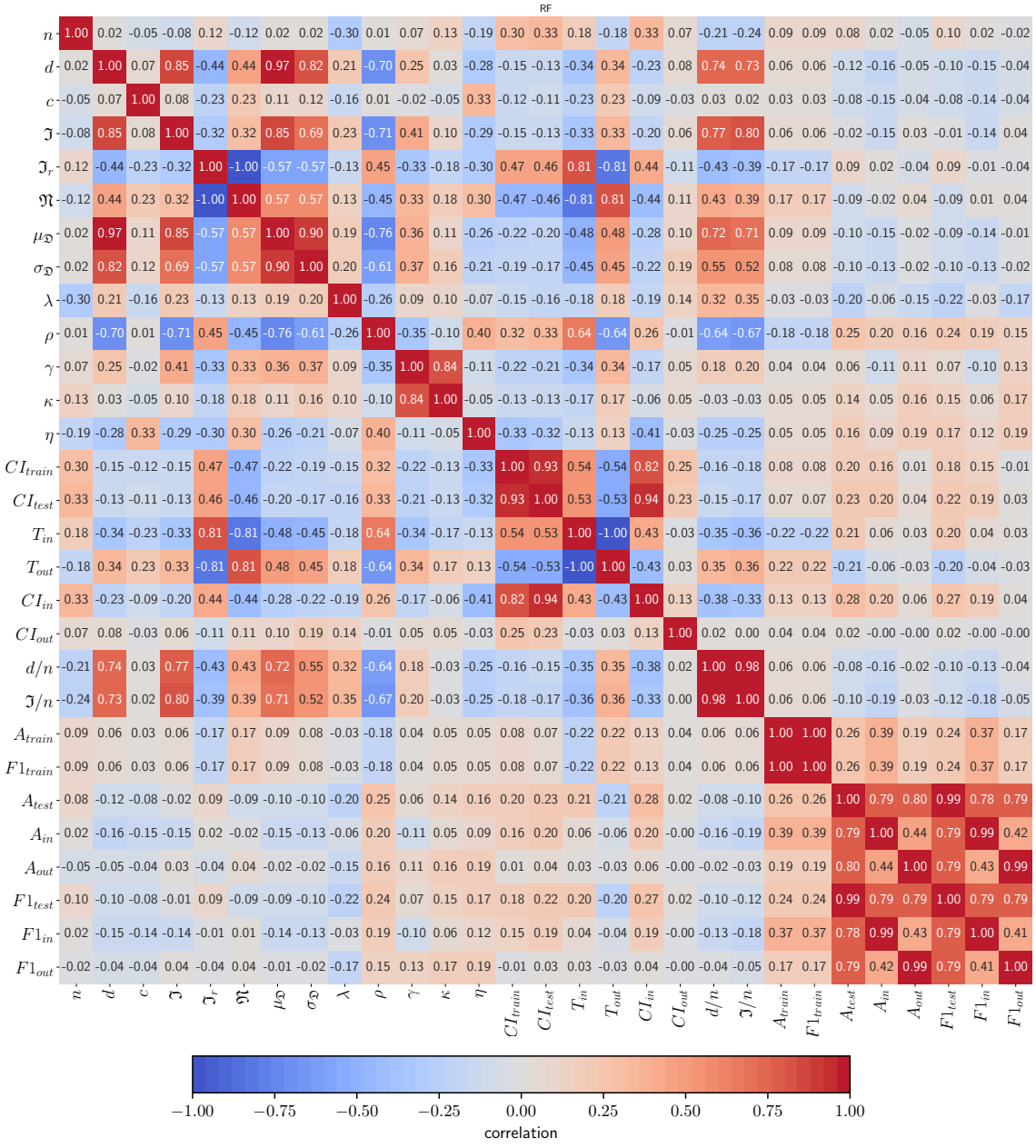


Figure 2.9: Correlations between data-set characteristics using Random Forest.

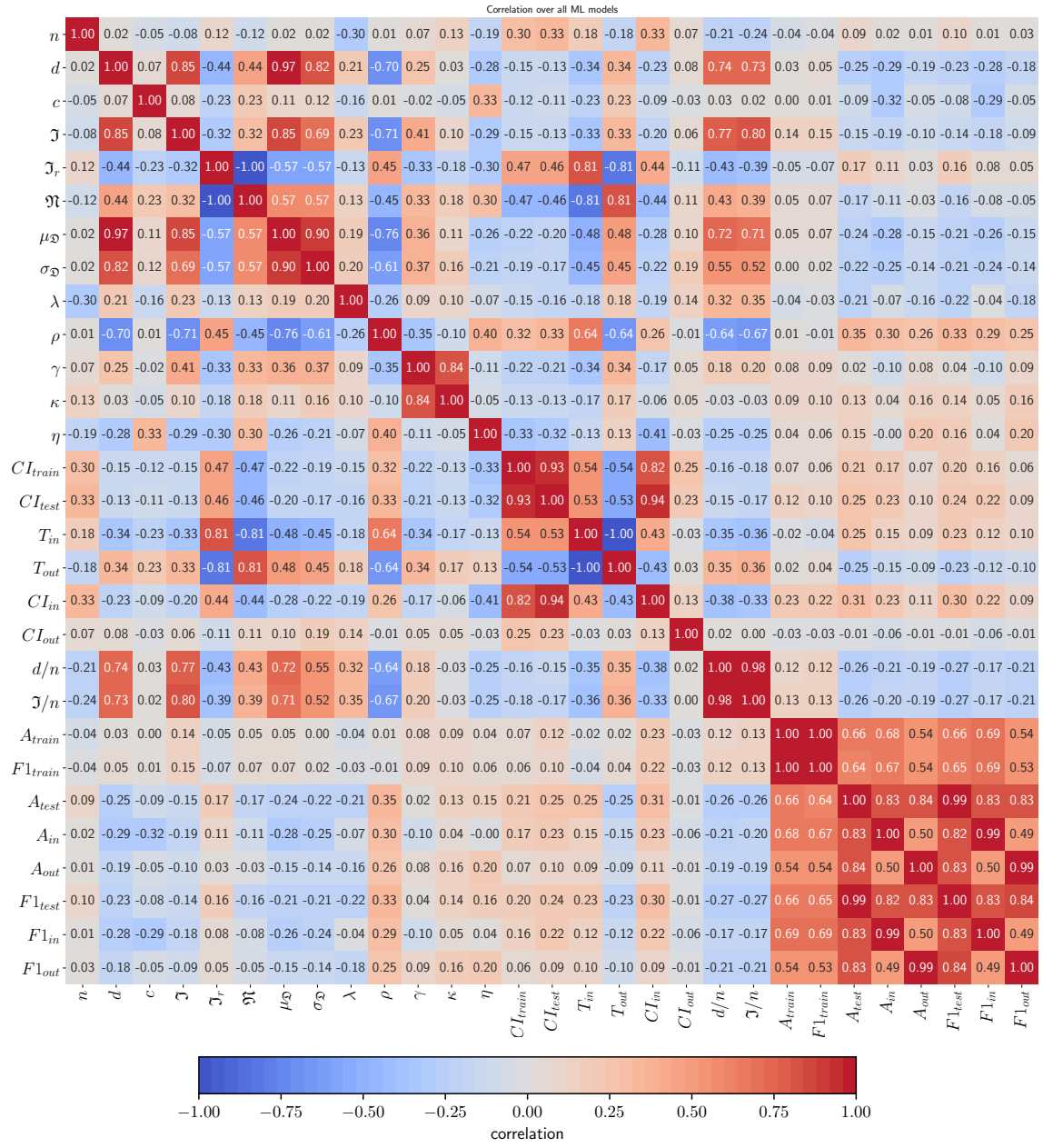


Figure 2.10: Correlations between data-set characteristics over all ML models used in the experiments.

Table 2.2: Average $F1$-score and standard error of the mean of ML models.

ML model	$F1_{train}$	$F1_{test}$	$F1_{in}$	$F1_{out}$
LR	$0.87 \pm 5.16 \cdot 10^{-3}$	$0.79 \pm 5.97 \cdot 10^{-3}$	$0.82 \pm 9.00 \cdot 10^{-3}$	$0.78 \pm 6.44 \cdot 10^{-3}$
SVC	$0.86 \pm 4.19 \cdot 10^{-3}$	$0.78 \pm 6.34 \cdot 10^{-3}$	$0.85 \pm 8.19 \cdot 10^{-3}$	$0.76 \pm 6.81 \cdot 10^{-3}$
RF	$0.99 \pm 1.01 \cdot 10^{-3}$	$0.84 \pm 4.88 \cdot 10^{-3}$	$0.88 \pm 6.76 \cdot 10^{-3}$	$0.82 \pm 5.78 \cdot 10^{-3}$

ML model	$\lvert F1_{train} - F1_{test} \rvert$	$\lvert F1_{train} - F1_{in} \rvert$	$\lvert F1_{train} - F1_{out} \rvert$
LR	$0.08 \pm 3.51 \cdot 10^{-3}$	$0.06 \pm 4.76 \cdot 10^{-3}$	$0.12 \pm 4.22 \cdot 10^{-3}$
SVC	$0.09 \pm 4.32 \cdot 10^{-3}$	$0.06 \pm 4.52 \cdot 10^{-3}$	$0.14 \pm 4.76 \cdot 10^{-3}$
RF	$0.15 \pm 4.74 \cdot 10^{-3}$	$0.11 \pm 6.27 \cdot 10^{-3}$	$0.18 \pm 5.58 \cdot 10^{-3}$

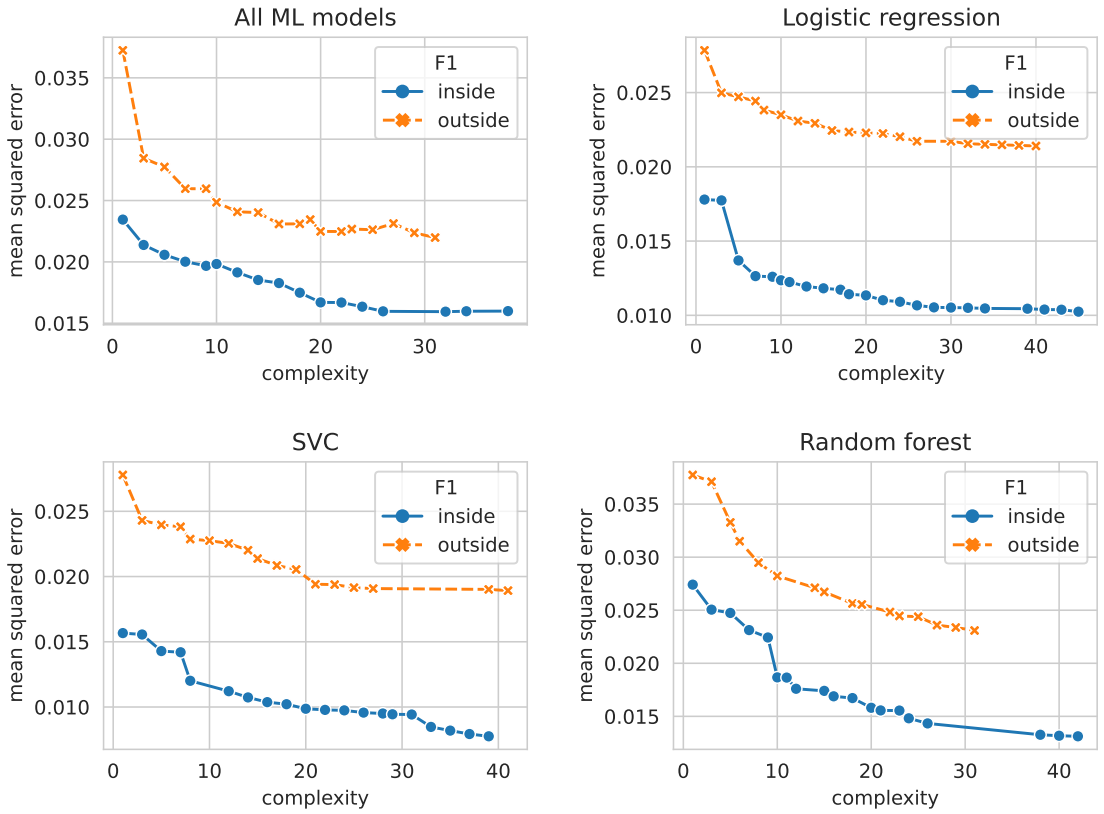


Figure 2.11: Pareto-optimal models predicting $F1_{in}$ and $F1_{out}$ based on data set characteristics taking into account the results on all ML models (**top-left**). Pareto fronts for each ML model: logistic regression (**top-right**), SVC (**bottom-left**), and random forest (**bottom-right**).

regression has the advantage of returning completely human-readable models, that can later be interpreted and explained, all the while considering relationships between data set characteristic more complex than just linear correlations. Furthermore, symbolic regression can deliver multiple candidate solutions, models of increasing complexity and fitting, whose meta-analysis can deliver

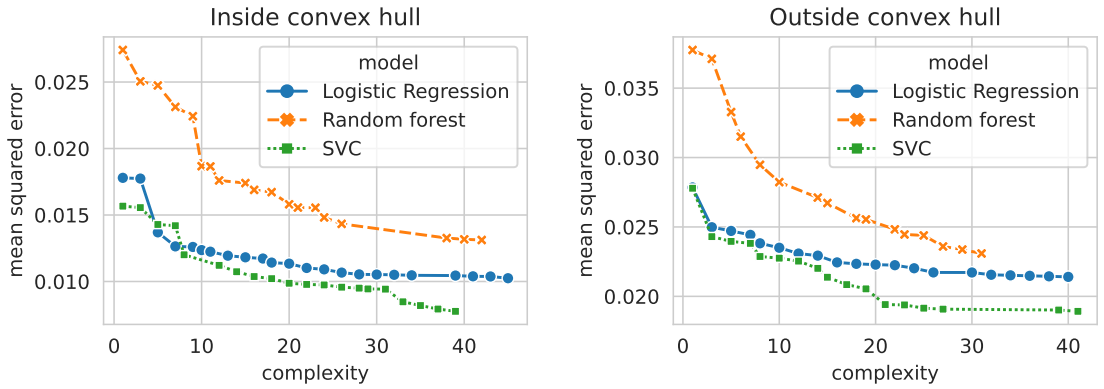


Figure 2.12: Comparison of Pareto fronts predicting $F1_{in}$ and $F1_{out}$ based on data set characteristics.

additional information to the user. For this task, the commercial symbolic regression software Eureqa Formulize[2] is employed. All available building blocks for equations were selected, with the exception of those specifically designed for time series analysis. Each run is stopped when the convergence metric of Eureqa crosses the threshold of 90%.

In order to assess generalization abilities of machine learning models, the results provided by Eureqa in different scenarios are analyzed and compared.For each classifier, symbolic regression generates a set of Pareto-optimal models, predicting the performance of the classifier in terms of $F1$-score. Pareto optimality is considered as a function of both R^2 (i.e. the accuracy of the formula) and complexity (the number of terms and complexity of formula's building blocks). Among all Pareto-optimal equations proposed by symbolic regression, a selection representing reasonable compromises between fitting and complexity is now manually analyzed. Eqs. 2.7-2.13 represent candidate equations of similar complexity ($C = 9$), describing nonlinear associations between data set characteristics and generalization metrics for each machine learning classifier.

$$F1_{in}^{LR} = F1_{train} + 0.020 \cdot \Im \cdot CI_{out}^{2} + \frac{-0.010 \cdot c \cdot F1_{train} \cdot CI_{out}^{5}}{\eta} \quad [R^2 = 0.79, C = 40] \quad (2.7)$$

$$F1_{out}^{LR} = 0.889 \cdot F1_{train}^{LR} + 0.031 \cdot \eta \cdot \gamma + \frac{0.002}{CI_{out} - 0.983} \quad [R^2 = 0.52, C = 16] \quad (2.8)$$

$$F1_{in}^{SVC} = F1_{train}^{LR} + \frac{\Im_r}{n - 201.639} \quad [R^2 = 0.71, C = 8] \quad (2.9)$$

[2]Eureqa Formulize is developed by Nutonian, Inc. `https://www.nutonian.com/products/eureqa/`

$$F1_{out}^{SVC} = 0.883 \cdot F1_{train}^{SVC} + \frac{0.044 \cdot CI_{out}}{CI_{test}} + \frac{0.118}{F1_{train}^{SVC} \cdot CI_{out}^2} - \frac{0.724 \cdot c \cdot \Im}{n}$$
$$[R^2 = 0.54, C = 38] \quad (2.10)$$

$$F1_{in}^{RF} = 192.439 + 0.157 \cdot \mathfrak{N} + 0.027 \cdot \Im + \frac{-15.814}{\rho} + 154.092 \cdot \rho \cdot log(\rho) - 64.840 \cdot e^{\rho} - 0.004 \cdot \gamma \cdot c$$
$$[R^2 = 0.52, C = 38] \quad (2.11)$$

$$F1_{out}^{RF} = 4.367 + 0.148 \cdot CI_{out} + 0.0001 \cdot \kappa + \frac{-1.364}{\rho} + \frac{0.250}{\gamma + CI_{out} \cdot \sigma_{\mathfrak{D}}} - 0.0035 \cdot c - 2.33 \cdot \rho$$
$$[R^2 = 0.39, C = 31] \quad (2.12)$$

$$T_{in} = \frac{3.381 + \gamma \cdot \Im_r}{d + \sigma_{\mathfrak{D}}} - 0.061$$
$$[R^2 = 0.89, C = 12] \quad (2.13)$$

Overall, the interpolation ability ($F1_{in}$) of a ML algorithm on a data set can be predicted in a satisfying way using only the data set characteristics analyzed in this study. On the contrary, predicting extrapolation ability ($F1_{out}$) from data set characteristics seems much harder, as pointed out by the lower R^2 scores of models for corresponding complexity. Besides, the difference between predictors found for LR or SVC, with respect to RF, is noteworthy: in fact, the generalization ability of the models for the first two algorithms seems strongly associated with their training performance ($F1_{train}$) and either the intrinsic dimensionality ratio ($\Im_r$) or the feature noise ($\mathfrak{N} = 1 - \Im_r$). On the other hand, the test accuracy of RF seems much harder to predict, and associated with the average class-wise correlation among features (ρ) *only*. Moreover, is it possible to observe how the ratio of test samples falling inside the convex hull could be easily estimated by considering the class-wise feature correlation (ρ) and either the feature noise ($\mathfrak{N}$) or the intrinsic dimensionality ratio ($\Im_r = 1 - \mathfrak{N}$), confirming the reasoning derived from the previous analysis of the linear correlations: higher feature noise leads to a lower effective dimensionality, thus reducing the likelihood of test samples falling inside the convex hull.

It is interesting to remark how $F1_{train}$ is the variable that explains most of the variance for LR and SVC models; while the same variable does not appear in models for RF, showing how the generalization ability of this classifier is poorly correlated with its performance on the training set, it is generally harder to predict (lower R^2), and seems to solely depend on the class-wise feature correlation ρ. The model for T_{in} also presents interesting insights, displaying rather high $R^2 = 0.88$ and depending on just two variables, ρ and $\mathfrak{N}$. The negative influence of $\mathfrak{N}$ can be explained

intuitively: as the feature noise increases, the intrinsic dimensionality ratio reduces, and thus the points belonging to the convex hull of the training set lie more and more on a polytope of lower dimensionality than the entire feature space, making it more difficult for test points to fall inside its hypervolume. The positive influence of ρ on T_{in} is harder to account for: a speculative explanation could be that a higher class-wise feature correlation would bring points belonging to the same class closer together in the feature space, as depicted in Fig. 2.1. Having the data points gathered in a smaller part of the feature space might imply that more test points will fall inside the convex hull of the training set, without necessarily reducing the true dimensionality of the feature space. This is somehow confirmed by the results reported in Fig. 2.6, where intrinsic dimensionality ratio and average class-wise feature correlation are positively correlated, albeit weakly ($\mathfrak{I}_r \diamond \rho = 0.45$).

The analysis of symbolic regression results is further extended by comparing Pareto fronts of ML models both inside and outside the convex hull to verify whether data set characteristics have a significant impact on models' performances. If Pareto front A dominates Pareto front B it is likely that data set characteristics have a higher impact on ML performances in the first scenario rather than in the second one. The Pareto front analysis is presented in Figs. 2.11 and 2.12. In Fig. 2.11, the link between data set characteristics and generalization ability is analyzed by comparing *interpolation* and *extrapolation* results of ML models. In all scenarios the relationship looks stronger when predictions are made inside the convex hull of training samples. However, while this difference is emphasized for LR and SVC, it is less pronounced for RF. In Fig. 2.12, symbolic regression results are further inspected by comparing Pareto fronts inside and outside the convex hull. Once more, it is possible to observe stronger associations between data set characteristics and LR or SVC compared to RF. This means that the impact of data set-specific properties on RF performances is lower, as if the higher capacity of the model would make it more robust.

2.1.6 Limitations

While the experimental findings of this work could provide novel contributions to the discussion on model generalization in the field of machine learning, it is also important to highlight the limitations of the study.

The first obvious limit lies in the task chosen for the analysis: the results obtained for classification might not be valid for regression tasks, not to mention other types of supervised ML or the whole field of unsupervised ML. Further experiments are needed in order to assess whether the conclusions obtained in this study can hold for other types of tasks. Nevertheless, given the considerable computational effort necessary to run the experiments presented in this section, the authors deem it interesting to present results for classification only, as it is the single task for which the most sophisticated data set metrics are available.

A second point of discussion is the use of the convex hull for estimating whether a ML model is interpolating or extrapolating while predicting a test sample. Evaluating the position of a test sample with respect to the convex hull of the training set is a very attractive metric, as it does not require setting arbitrary thresholds and it can be computed efficiently. There are, however, a few drawbacks: if the number of features is higher than the number of available samples, in other words $d > n$, as previously discussed all samples in the test set will have a high probability of falling outside the convex hull of the training set. So, for corner cases, the value of a quality metric such as $F1_{in}$ might simply not exist, as no samples from the test set would fall into the convex hull of the training set. This might somehow limit the validity of the conclusions drawn for the correlations with $F1_{in}$.

After presenting the analysis of the correlations found on the 72data sets analyzed, there is a

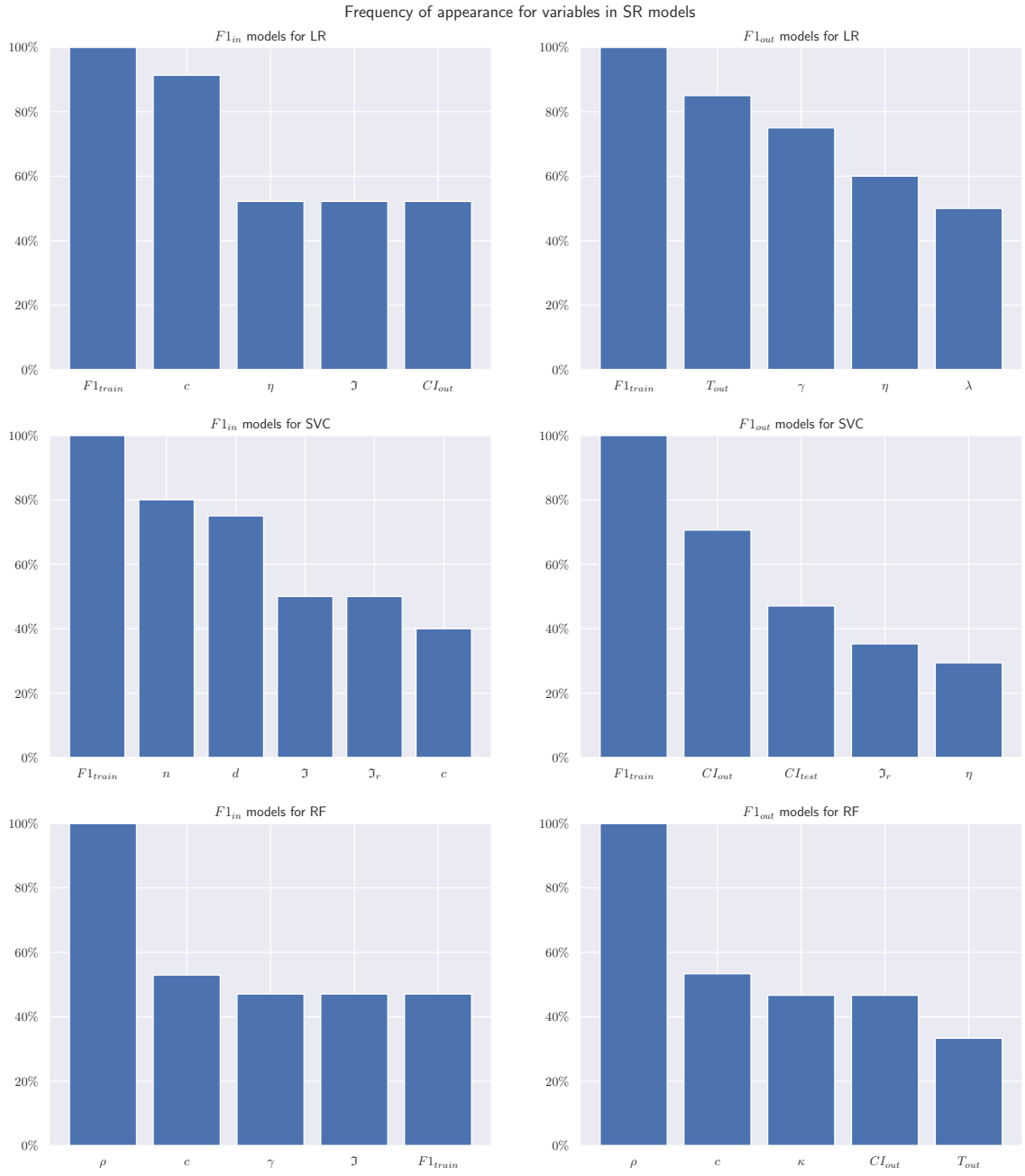


Figure 2.13: Frequency of appearance of the most relevant variables corresponding to data-set characteristics inside SR models for $F1_{in}$ and $F1_{out}$on the Pareto fronts. It is interesting to notice how the performance on the training set $F1_{train}$ is a highly predictive variable for both LR and SVC, while it has lower to no predictive power for RF.

(rather ironic) question to be faced: how *general* are the results found so far? Or, in other words, how well do the *predictions* performed *extrapolate* to unknown data sets? Frankly speaking, it would be unwise to claim that the correlations described in this work hold for all possible data sets, but the sheer number of different data sets analyzed gives some hope of generality.

There is, however, a possible bias in the selection of data sets for this study: the focus on openly accessible, curated data sets, that already had to pass several quality checks in order to be hosted on repositories such as OpenML. This pre-selection process might make the data sets considered in this work not representative of all real-world data sets. In other words, usually a data set is uploaded on OpenML because the authors already know that at least one ML technique is going to work well for that specific data set; thus, what was analyzed might be representative only of data sets for which ML techniques work well. It is also important to remark that, in order to use the convex hull, only data sets with real-valued features were selected, a restriction that might further narrow the scope of the insights presented.

Another possible explanation for the most counter-intuitive correlations uncovered is that real-world data sets are a subset of all possible data sets. While some general mathematical conclusions, such as the curse of dimensionality, might hold for the set of all hypothetical data sets, they might not necessarily be true for the subset of data that is measured from real phenomena. This observation mirrors the remarks by Lin et al. [LTR17]: in an attempt to explain the effectiveness of neural networks and ML at representing physical phenomena, the authors notice that laws of physics can typically be approximated by a tiny subset of all possible polynomials, of order ranging from 2 to 4; this is a consequence of such phenomena usually being symmetrical when it comes to rotation and translation. As the data sets analyzed in this study come from either simulations or real-world experiments, their characteristics might lead ML algorithms to represent them more easily than expected.

2.2 Extracting Meaningful Information

Following the leading question of this chapter, related to how ML algorithms learn and represent knowledge, another related topic I worked on is the extraction of minimal meaningful information, in the form of *features* and *coresets*. When supervised ML algorithms are training on data, they need to select a subset of the information provided as the most important for the task at hand. Not only, but being based on greedy optimization procedures, ML algorithms can sometimes encounter difficulties if the search space presented, be it in number of samples or number of features, is too large: from an optimization point of view, this translates to falling into an unsatisfying local optimum. From the point of view of optimization, it logically follows that by reducing the search space the task becomes less complex, and thus most promising solutions can be found more easily. From a ML perspective, this leads to the somehow counter-intuitive conclusion that providing *less* information to the algorithm actually provides better results.

Selecting the minimal, meaningful information to provide to an algorithm is a problem that the ML approached from two different directions: *feature selection* deals with the features, the variables in the ML problem, while *coreset* and *prototype discovery* deal with the samples.

2.2.1 Feature selection

Feature selection is the process of choosing, or removing, features to obtain the most informative feature subset of minimal size. Such subsets are used to improve performance of machine learning algorithms and enable human understanding of the results. In my research, I tackled this issue

from two different perspectives: (i) a multi-objective approach, aiming at finding compromises between number of features and performance of a target ML algorithm, taking into account the non-separability of the performance of different feature subsets; and (ii) a recursive elimination procedure, where features whose values can be predicted starting from others are identified as redundant and can thus be safely removed.

Background

In ML, feature selection is the process of choosing (or eliminating) features from a dataset, reducing them to the minimal, most informative subset. Removing information might, at first glance, seem detrimental for the performance of ML algorithms: however, certain features might just add noise; or they might be redundant, for example being heavily correlated with others; and finally, eliminating features reduces the search space that ML algorithms have to explore, facilitating the task of finding effective models.

Besides improving the performance of ML algorithms (not only in terms of computation time but also regarding precision of results [FK12]), feature selection can also be used to reduce information and ultimately make it human-readable. For example, while reviewing the contributions of 1,000 different variables in a problem is impossible for human experts, a selection of 10 highly-informative features can usually be analyzed, even if relevant parts of the information are removed. This is particularly useful when dealing with genomic or other high-dimensional data [Ber+15]. More generally, feature selection is one facet of dimensionality reduction, which is an important domain in the field of data visualization [Tsa12].

Feature selection can be performed using various approaches [GE03], simple ones consist in filtering the features according to a criterion (often based on statistical tests), or in using recursive procedures (forward or backwards) to eliminate redundant features [Lew62][CF67]. Subset selection methods are more complex and rely on the definition of a quality measurement of the subset. The problem is thus turned into an optimization one: selecting the best subset of features that maximizes an objective function (usually a "goodness-of-fit" combined with a regularization term, including a penalty for a large number of variables [GE03; Wes+00]). Several single-objective EAs have been proposed, exploiting similar scores for the fitness function [Cil+19; Xue+15]. Finally, feature construction and space dimensionality is another way to reduce information. Subsets made of combinations of features are built for a better representation of the dataset (dimensionality reduction methods, principal component analysis for instance).

Given a candidate subset of features, evaluating its efficacy is not trivial. Ideally, what would need to be measured is the *content of information* of the feature subset, and several metrics have been proposed to assess it in literature: for example *mutual information* [KL87] or *analysis of variance* [Fis19]. In practice, however, even the most popular metrics can only assess part of the information content of a feature subset, as taking into account the contribution of non-linear combinations of features is too computationally expensive.

A different way to assess efficacy for a feature subset is using it as input of a ML algorithm, and evaluate the difference in performance compared with the same algorithm, using all features, or a different feature subset. To avoid issues with overfitting, a K-fold cross-validation can be used, obtaining an average of its performance (for example, classification accuracy) on the test folds. As the cross-validation procedure is stochastic, comparing two feature subsets on just their average performance on test folds is not enough, because the variance of the results is not taken into account. A more robust approach is to consider the K performance results on test folds of the two feature subsets as samples drawn from two probability distributions, and exploit a statistical test to

assess the likelihood that the two sets of samples are drawn from different distributions. If the two sets of samples are separable below an arbitrary confidence threshold, for example $p < 0.05$, the feature subset with the best average performance can be considered better than the other. The main issue of this methodology is that it is sometimes impossible, with the available data, to separate the performance of different feature subsets.

Multi-objective Approaches

Multi-objective optimization algorithms aim at finding the best compromises between conflicting criteria, ultimately delivering a set of non-comparable, non-dominated solutions to the users. Evolutionary Algorithms (EAs) currently represent the state-of-the-art in the field, with the Multi-Objective EA (MOEA), Non-Sorting Genetic Algorithm II (NSGA-II) [Deb+02] being one of the most widely adopted for real-world applications. Given their effectiveness, it is not surprising that MOEAs have been already applied to feature selection problems, where the conflicting objectives are usually: i. minimizing the number of features and ii. maximizing a quality metric for a feature subset. In [Ham+07] the authors apply NSGA-II for feature selection. In [XFZ14], differential evolution is used instead. MOEA approaches to feature selection have been recently applied to facial recognition [VMS13] and medical imaging [Zho+19].

In [Bar+20], I proposed a novel approach to feature selection in ML, framing it as a multi-objective problem with three aims: i. minimizing the number of features; ii. minimizing error on a cross-validation; iii. maximizing mutual information content between each feature and the target. Feature selection can be seen as finding the best compromises between the number of features considered and the final result for a ML algorithm. However, assessing the effectiveness of the selected features for the problem is far from trivial, and only indirect metrics are available.

It must be noted that analyzing all feature subsets for a given dataset is often impossible, as the total number of feature subsets of dimension d for a dataset with F features is:

$$\sum_{d=1}^{F} \binom{F}{d} = 2^F \tag{2.14}$$

Individual representation

Individuals represent feature subsets, and are internally stored as simple bit-strings of size equal to the number of features in the original dataset. A '1' in the i-th position of an individual means that the corresponding i-th feature is included in the subset; a '0' indicates that the i-th feature is not included in the subset.

Fitness function

The first objective in the proposed approach is to minimize the number of features included in a subset:

$$O_1 = \sum_{i=1}^{F} I(i) \tag{2.15}$$

where I is an individual represented as a bit-string, $I(i)$ indicates the bit in i-th position, and F is the number of features in the problem, also corresponding to the size of an individual.

The second objective assesses the effectiveness of a candidate feature subset for a specific problem, through a K-fold cross-validation, a procedure where training data is divided into K parts,

termed *folds*, that are alternatively used for training and test. This objective can be stated as:

$$O_2 = \frac{1}{K}\sum_{i=1}^{K} L_{k(i)} \tag{2.16}$$

where K is the number of folds; $k(i)$ is the i-th fold. $L_{k(i)}$ is defined as:

$$L_{k(i)} = L(y_{k(i)}, \hat{g}^{-k(i)}(x_{k(i)})) \tag{2.17}$$

where L is an error function, evaluating the differences between the values predicted by $\hat{g}^{-k(i)}$ and the known values $y_{k(i)}$; $\hat{g}^{-k(i)}$ is the function learned by a ML algorithm, trained on all data, except fold $k(i)$; in general, $\hat{g}$ is always considered to be an approximation of the real function g that generated the known values of y; $y_{k(i)}$ and $x_{k(i)}$ are the known values of the target and the corresponding features for samples in $k(i)$, respectively. The error measured by L is averaged over the K folds to obtain the final value of O_2.

Finally, the third objective is a proxy for human readability of the candidate feature subset. Using the one-way Analysis of Variance (ANOVA) F-value procedure [Hei01], that captures univariate relationships between a feature and the target. Indeed, the F-value of the i-th feature ϕ_i can be interpreted as the proportion of variance explained by the feature to the total variance in the data. Making the reasonable assumption that a higher amount of explained variance may correspond to a higher discriminating capability, it is then possible to rank features according to their ϕ, where the best feature will have the highest value. Finally, for each subset of features (i.e. the candidate solution), the third fitness objective is function of the worst ϕ in the feature subset:

$$O_3 = \frac{1}{\min(\phi_0, \phi_1, ..., \phi_f)} \tag{2.18}$$

where f is the number of features in the subset. This objective will force the evolutionary process to drop feature sets containing at least one variable whose univariate contribution is negligible. In fact, ML classifiers as well as other automatic FS algorithms (such as RFE) risk to retain a dramatic amount of features which are not a true causative source of the observed phenomenon (a.k.a. false positives). However, they are often selected as they might be slightly correlated with the target, providing a minor contribution to the classification accuracy.

Taking into account Eq. 2.15, 2.16, and 2.18, the multi-objective optimization problem can be described as:

$$argmin(O_1, O_2, O_3) \tag{2.19}$$

Experimental Evaluation of the Multi-Objective Approach

The experiments presented in this work deal with classification only, due to the greater availability of high-dimensional classification datasets in the public domain; but the proposed methodology can also be straightforwardly applied to regression problems. The experimental evaluation of the proposed approach is divided into two parts. Firstly, datasets that have a relatively low dimensionality (9-18 features) are analyzed: as all feature subsets for these datasets can be explored exhaustively, it is possible to assess whether there is actually a single best feature subset, and whether different methodologies are able to find it. In a second batch of experiments, the proposed methodology tackles an artificial dataset with high dimensionality (500 features), that cannot be analyzed exhaustively, but whose characteristics are completely known.

Experimental setup

For the following experiments, the error function L (see Equation 2.17) is classification error, an established quality metric for classifiers, simply defined as the ratio between incorrect predictions and total predictions. The closer classification error is to 0, the higher the quality of the predictive model. The classifier used to learn $\hat{f}$ in O_2 is Logistic Regression [Cox58], a popular algorithm of proved effectiveness.

The MOEA selected for the experiments is NSGA-II [Deb+02], that currently represents the state of the art for multi-objective optimization with up to three objectives. After preliminary evaluations, NSGA-II's parameters are set to: $\mu = 100$, $\lambda = 100$, probability of crossover $p_c = 0.9$, probability of mutation $p_m = 1/l$, where l is the length of an individual, and a stop condition based on the maximum number of generations, set to 200.

The proposed approach, termed *EvoFS* in tables and figures, is compared against three popular state-of-the-art feature selection methods: recursive feature elimination (RFE) [Guy+02], that uses a classifier to score a feature set, then iteratively removes the lowest-performing feature and scores the subset again; greedy forward selection, that greedily adds features to a subset, using either their mutual information (MI) [KL87], or analysis of variance (ANOVA) [Fis19] scores. All these methods need the user to specify the number of features to be selected, so in the experiments they have been called once for every possible size of feature subset in the problem, to have a fair comparison.

As previously stated, comparing the effectiveness of two feature subsets for classification using the error function L is not trivial, due to possible random effects in the classifier's training process, or in the way the training/test split of the data is performed. Randomly dividing the data in K folds and performing a K-fold cross-validation can help obtain a better average for L, but introduces further stochasticity in the process. When comparing results in this work, the outcome of a K-fold cross-validation is considered as K separate samples coming from an unknown statistical distribution. The results of two feature subsets are then compared as if assessing the likelihood that their accuracy scores have equal means. As it is not reasonable to assume that the two distributions have the same standard deviation, a Welch's T-test [Wel47] is applied, with an arbitrary but commonly accepted threshold for the p-value ($p < 0.05$). Such a statistical test assumes that samples are drawn from populations that are normal in shape. As pointed out in [KA14], this assumption is quite easy to meet for a wide range of practical distributions at a significance level $\alpha = 0.05$ and a sample size of $K \geq 5$. In the following, $K = 10$. This procedure will be used to isolate clusters of feature subsets that are non-separable for their classification error, and can thus be considered all equally optimal with regards to this metric. For each considered dataset, running times for all algorithms are reported in Table 2.4.

All the code in the experiments has been implemented in Python v3, using the modules `scikit-learn` [Ped+11b] for all ML and feature selection algorithms, `openml` [Cas+17] for accessing the datasets in the OpenML repository, and `inspyred` [Gar12] for NSGA-II. The scripts are freely available in a Bitbucket repository[3]. Experiments have been run on a consumer-end laptop[4].

Simple datasets

In a first set of experiments, simple datasets with a limited number of features are examined. The advantage of dealing with such datasets is that all their feature subsets can be enumerated and analyzed, a task that becomes impossible if dealing with hundreds or thousands of features. The datasets are freely accessible on the OpenML repository [Van+13], and their characteristics are

[3] `https://bitbucket.org/evomlteam/moea-feature-selection`

[4] Intel® Core™ i7-8750H 2.20 GHz, 8 GB RAM.

summarized in Table 2.3.

Table 2.3: Characteristics of the datasets used in the experiments.

Dataset name	Type	Features	Samples	Classes	Feature subsets
diabetes [DG17]	Medical	9	768	2	512
australian [Qui87]	Credit scores	14	690	2	16,384
vehicle [Sie87]	Vehicle recognition	18	846	4	262,144
Madelon [Guy+05]	Artificial	500	4,400	2	10^{150}

Figures 2.14, 2.15, and 2.16 show how many non-separable feature subset that have size lower or equal to the best performing one each algorithm was able to find: ideally, these are the ones that human users should be interested in. Then, for each algorithm, the position of each non-separable solution found is mapped into the exhaustive exploration of all feature subsets.

Figure 2.14 reports the results for the *diabetes* dataset. While all approaches are able to find feature subsets that are non-separable from the best ones, EvoFS finds the largest number. The same holds for the *australian* dataset, in Figure 2.15, where notably RFE seems unable to find good solutions of small size. For *vehicle*, that features the largest search space so far, results reported in Figure 2.16 show that, this time, RFE performs much better than the other two comparing methods, equalling the performance of EvoFS. Nevertheless, EvoFS is able to find a few non-separable solutions that are of smaller size than those uncovered by RFE.

An interesting general behavior that emerges from the plots, is that EvoFS is able to find non-separable feature subsets of lower size than the other algorithms. Notably, non-separable solutions of size larger than the best performing one are not included in its Pareto fronts.

High-dimensional datasets

The second set of experiments deals with a high-dimensional dataset, for which an exhaustive analysis of all feature sets is impossible. This datasets is artificial, taken from a classification competition focused on feature selection [Guy+05]. The datasets' characteristics are summarized in Table 2.3.

The targed dataset, named *Madelon*, is an artificial dataset that can be procedurally generated, with a few informative features, several features that are linear combinations of the informative features, and a large number of deceiving features called *probes* [Guy03]. For this work, an instance of Madelon is generated with the same parameters as the one featured in the competition [Guy+05]: 5 informative features, 15 linear combination features, 480 deceiving features/probes.

Figure 2.17 illustrates a summary of the results on the Madelon dataset. Remarkably, EvoFS is able to find a higher number of non-separable feature subsets having size lower or equal to the overall best solution. Moreover, EvoFS is also the only algorithm able to identify a non-separable solution of size 3, that includes only informative features (in positions 2, 3, 18).

While the greedy algorithms continue to be extremely fast on the high-dimensional dataset, it is noticeable from the running times reported in Table 2.4, how now RFE, with its iterative process, scales much worse than EvoFS.

Predictable Feature Elimination

Predictable Features Elimination, the approach originally presented in [BST20b], stems from the observation that if the distribution of a feature f_r can be approximated by using the information of

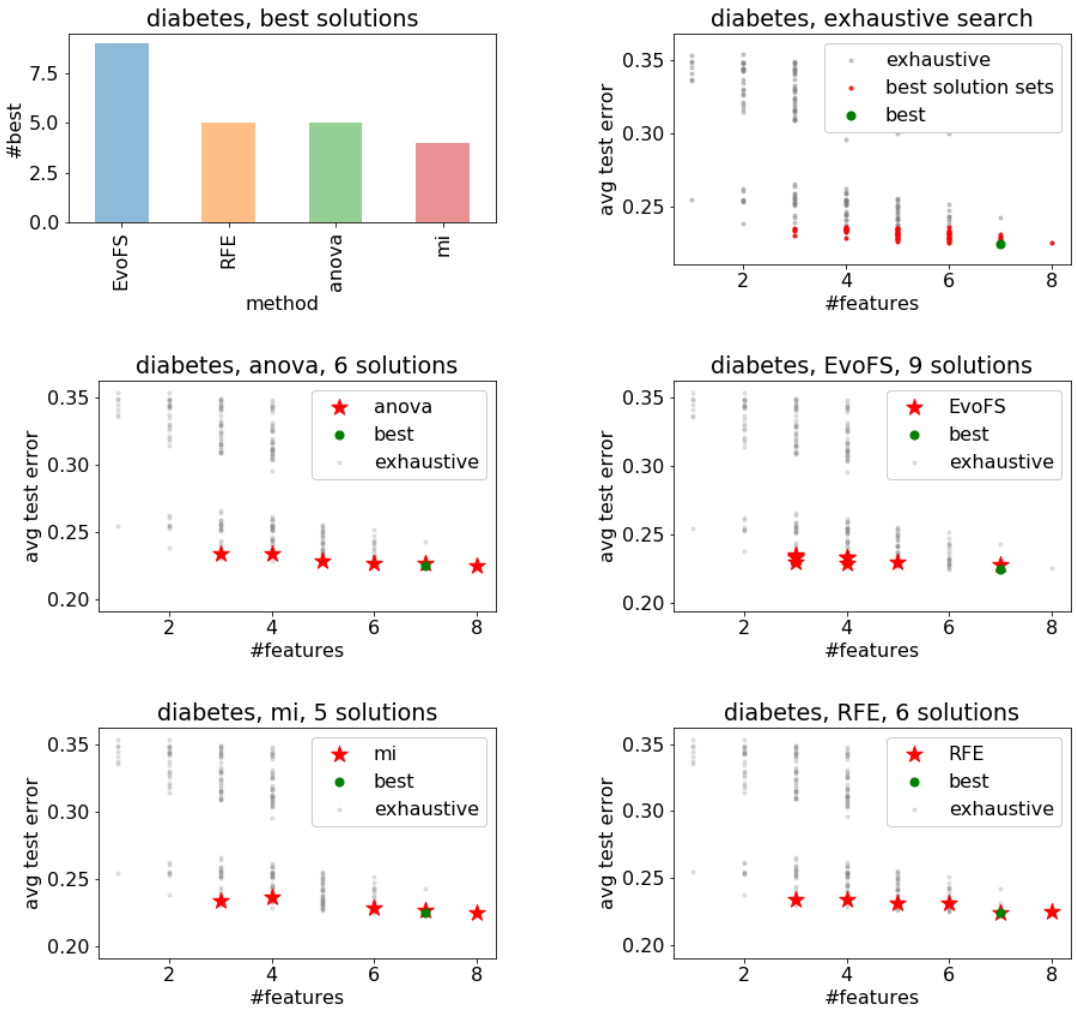


Figure 2.14: **(top left)** Number of non-separable optimal feature subsets, of size less or equal to the one with the lowest error, found by each algorithm. **(top right)** All possible feature subsets for the dataset, identified exhaustively. In red, for each size, the ones that are non-separable. In green, the single feature subset with the lowest average error. **(middle-left to bottom-right)** Features subsets uncovered by the different approaches.

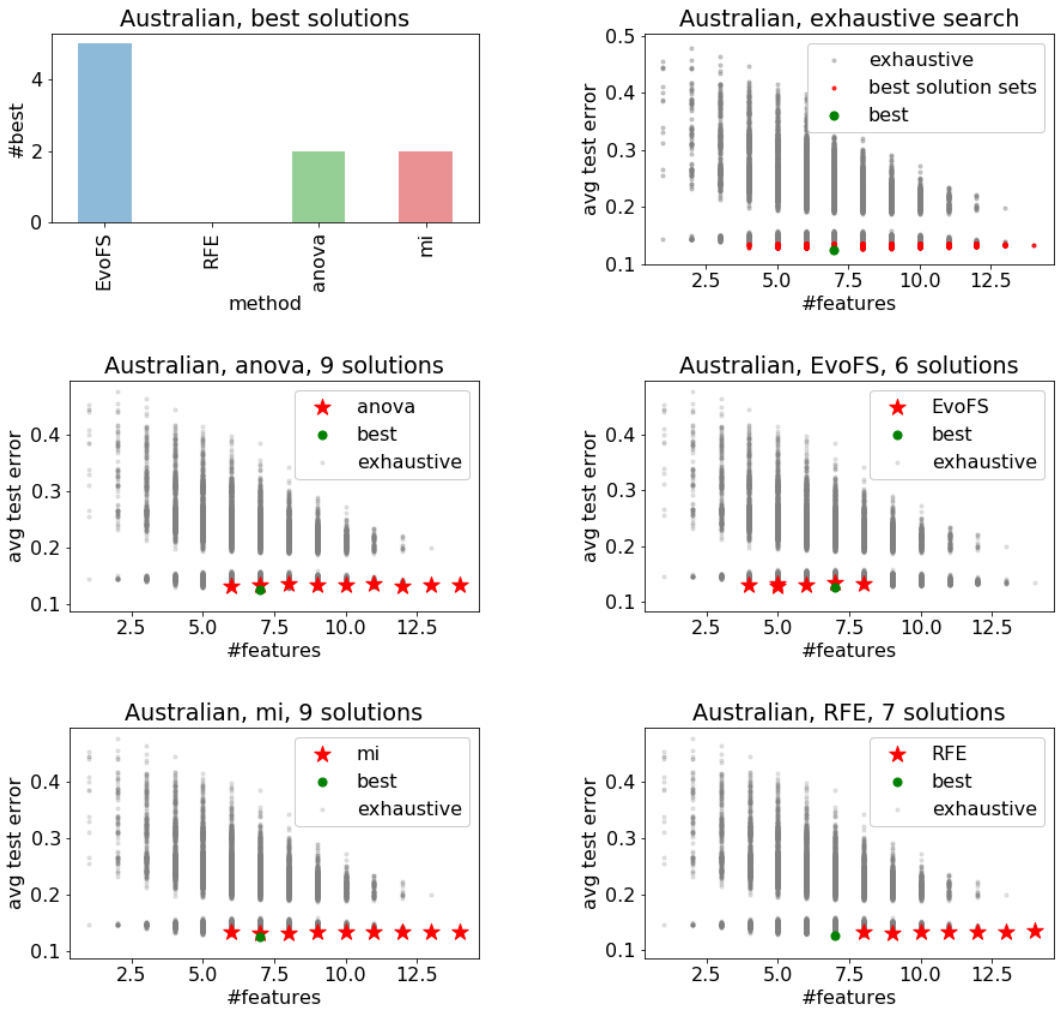


Figure 2.15: **(top left)** Number of non-separable optimal feature subsets, of size less or equal to the one with the lowest error, found by each algorithm. **(top right)** All possible feature subsets for the dataset, identified exhaustively. In red, for each size, the ones that are non-separable. In green, the single feature subset with the lowest average error. **(middle-left to bottom-right)** Features subsets uncovered by the different approaches.

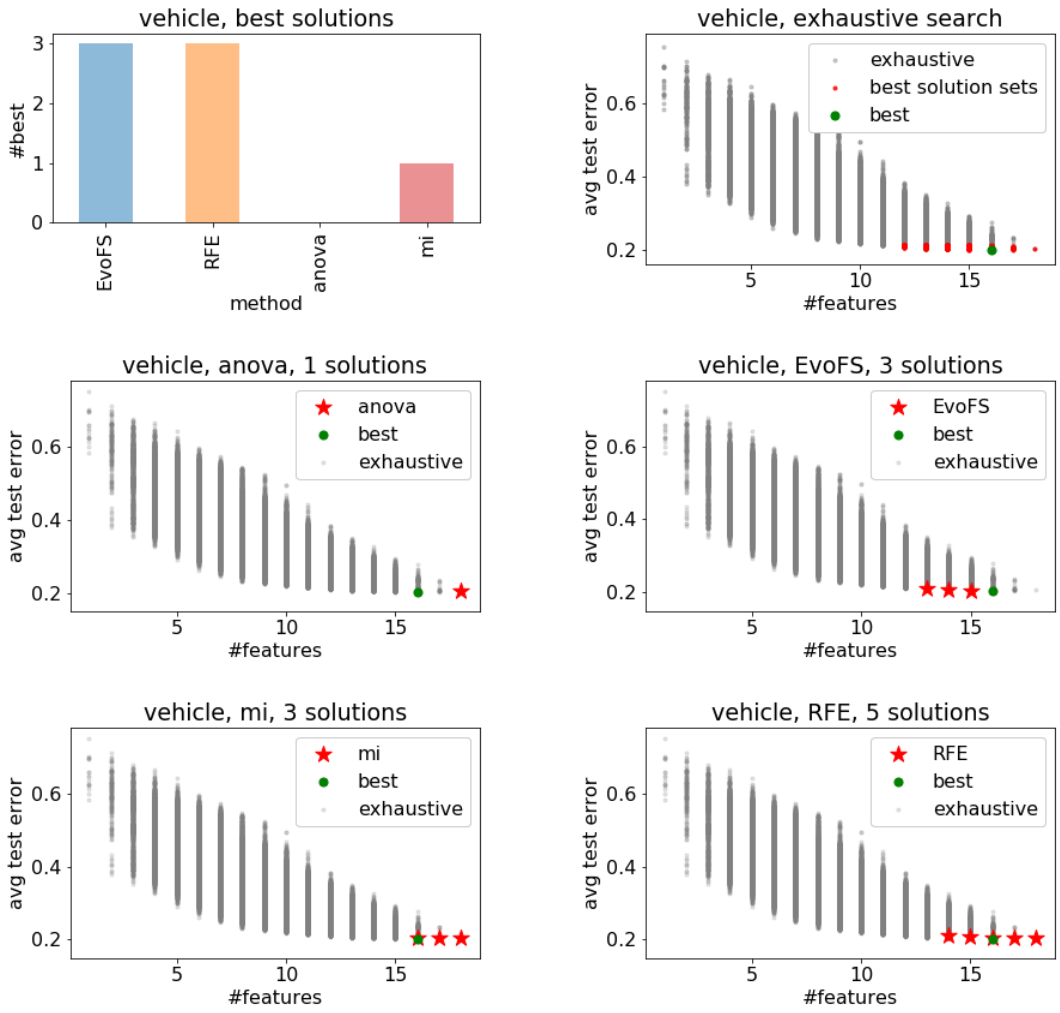


Figure 2.16: **(top left)** Number of non-separable optimal feature subsets, of size less or equal to the one with the lowest error, found by each algorithm. **(top right)** All possible feature subsets for the dataset, identified exhaustively. In red, for each size, the ones that are non-separable. In green, the single feature subset with the lowest average error. **(middle-left to bottom-right)** Features subsets uncovered by the different approaches.

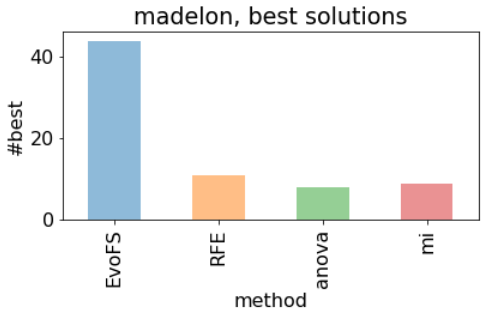


Figure 2.17: Number of non-separable optimal feature subsets, of size less or equal to the one with the lowest error, found by each algorithm, on the *madelon* dataset.

Table 2.4: Running time (seconds) of the feature selection algorithms.

Dataset	**EvoFS**	**anova**	**MI**	**RFE**
diabetes	421.28 s	0.02 s	0.05 s	0.10 s
australian	579.38 s	0.03 s	1.47 s	0.24 s
vehicle	819.43 s	0.03 s	2.17 s	1.88 s
madelon	3,549.57 s	0.05 s	12.97 s	18,925.29 s

other features, then f_r is likely to be almost redundant and of little importance for whatever model will be eventually built. Algorithm 1 summarizes the main steps of the training process.

PFE requires the user to provide two parameters: an auxiliary machine learning model g and a threshold σ. The first is the model used to discriminate non-redundant features, the second, the acceptable loss of information. A sub-optimal choice of g would cause some features to be erroneously marked as non-redundant, increasing the size of the feature set, but probably not affecting the quality of the final model. On the other hand, a low σ is likely to make PFE select a very small set of features, but also to impair the quality of the final model.

The algorithms is composed of two phases: an initialization and the main loop. In the former, features are ranked according to their mutual average linear correlation. First, the feature correlation matrix C is computed:

$$C = \Big(diag(K_{XX})\Big)^{1/2} K_{XX} \Big(diag(K_{XX})\Big)^{1/2} \tag{2.20}$$

where K_{XX} is the auto-covariance matrix of the input matrix X:

$$K_{XX} = E[(X - E[X])(X - E[X])^T)] \tag{2.21}$$

The correlation matrix C is used to estimate the amount of mutual information shared among features. By summing up the rows of C, for each feature an approximation of the amount of information which can be obtained using all the other features is then obtained:

$$\kappa = \sum_i C(i, \cdot) \tag{2.22}$$

The more a feature is correlated with others, the lower the chances that the feature may contain exclusively useful information. Hence, by ranking features according to their mutual average linear correlation, an ordered list of their significance is obtained:

$$\kappa_s = sort(\kappa) \tag{2.23}$$

Starting from the feature with the highest rank f, the ML model g is trained on the remaining features using the f-th feature as a target variable y:

$$y = X(\cdot, \kappa_s(f)) \tag{2.24}$$

The performance of g is assessed on a validation set X_{val} using the coefficient of determination R^2. If the validation score is greater than the user defined threshold σ, then it means that the model g represents an accurate nonlinear association between the f-th feature and the other features. Hence, the chances that the f-th feature may contain exclusively useful information are low. Therefore, the f-th feature should be safely removed from the feature set and the process may continue using the following feature in the ranking. The algorithm stops when more than half of the features have been analyzed.

Algorithm 1: Predictable feature elimination

Input: data $X \in \mathbb{R}^{n,d}$, model g, threshold $\sigma \in [0,1]$
Initialize $C = corr(X^T)$
Initialize $\kappa = \sum_i C(i,\cdot)$
Initialize $\kappa_s = sort(\kappa)$
for $f = 1$ **to** $\lfloor d/2 \rfloor$ **do**
  Initialize $y = X(\cdot, \kappa_s(f))$.
  Split data into train and validation sets
  Train model $g \leftarrow (X_{train}, y_{train})$
  Make validation predictions $\hat{y} = g(X_{val})$
  Evaluate predictions $score = R^2(\hat{y}, y_{val})$
  if $score \geq \sigma$ **then**
    Remove current feature
  end if
end for

I introduce the following theorems to yield a theoretical justification for the proposed approach. Besides, they show how the performance loss can be formally estimated by providing upper bounds in worst case scenarios.

Theorem 2.2.1 — Elimination for linear combinations. Let $\mathscr{F}$ be a set of features $\mathscr{F} = \{f_1, \ldots, f_d\}$ and $\mathscr{F}'$ be a subset of $\mathscr{F}$ such that $f_i \notin \mathscr{F}'$. If the feature f_i is a linear combination of the feature set $\mathscr{F}'$, then fitting a linear model using $\mathscr{F}'$ is equivalent to fitting the same model using $\mathscr{F}' \cup \{f_i\}$.

Proof. By definition f_i is a linear combination of $\mathscr{F}'$, hence:

$$f_i = \sum_{j \neq i} w_j f_j \tag{2.25}$$

which can be written as:

$$f_i = F' w \tag{2.26}$$

where $F' \in \mathbb{R}^{n \times d'}$ is a matrix whose columns are features in $\mathscr{F}'$ and $w \in \mathbb{R}^{d'}$ is a row vector containing the weights of the linear combination.

A linear model g trained using the matrix F' can be written as:

$$\widehat{y} = g(F') = F' w^g = \sum_{j \in d'} f_j w_j^g \tag{2.27}$$

Let F'' be the matrix whose columns correspond to the features in $\mathscr{F}' \cup \{f_i\}$, then the model g trained on F'' can be written as:

$$\begin{aligned}
\widehat{y} = g(F'') = F'' w^g &= \sum_{k \in d''} f_k w_k^g \\
&= \sum_{j \in d'} f_j w_j^g + w_i^g f_i \\
&= \sum_{j \in d'} f_j w_j^g + w_i^g \sum_{j \in d'} f_j w_j \\
&= \sum_{j \in d'} f_j w_j^g + \sum_{j \in d'} f_j w_j w_i^g \\
&= \sum_{j \in d'} f_j w_j^g w_j w_i^g \\
&= \sum_{j \in d'} f_j \omega_j^g
\end{aligned} \tag{2.28}$$

■

Theorem 2.2.2 — Approximate elimination for linear combinations. Let $\mathscr{F}$ be a set of features $\mathscr{F} = \{f_1, \ldots, f_d\}$ and $\mathscr{F}'$ be a subset of $\mathscr{F}$ such that $f_i \notin \mathscr{F}'$. If the feature f_i can be written as a linear combination of the feature set $\mathscr{F}'$ with an additional term ε, then the upper bound of the training error obtained by fitting a linear model on $\mathscr{F}'$ instead of $\mathscr{F}' \cup \{f_i\}$ is at most ε.

Proof. By definition f_i can be written as a linear combination of the feature set $\mathscr{F}'$ with an additional term ε, hence:

$$f_i = \sum_{j \neq i} w_j f_j + \varepsilon \tag{2.29}$$

which can be written as:

$$f_i = F' w + \varepsilon \tag{2.30}$$

where $F' \in \mathbb{R}^{n \times d'}$ is a matrix whose columns are features in $\mathscr{F}'$ and $w \in \mathbb{R}^{d'}$ is a row vector containing the weights of the linear combination.

A linear model g trained using the matrix F' can be written as:

$$\widehat{y} = g(F') = F'w^g = \sum_{j \in d'} f_j w_j^g \tag{2.31}$$

Let F'' be the matrix whose columns correspond to the features in $\mathscr{F}' \cup \{f_i\}$, then the model g trained on F'' can be written as:

$$\begin{aligned}
\widehat{y} = g(F'') = F''w^g &= \sum_{k \in d''} f_k w_k^g \\
&= \sum_{j \in d'} f_j w_j^g + w_i^g f_i \\
&= \sum_{j \in d'} f_j w_j^g + w_i^g \sum_{j \in d'} f_j w_j + \varepsilon \\
&= \sum_{j \in d'} f_j w_j^g + \sum_{j \in d'} f_j w_j w_i^g + \varepsilon \\
&= \sum_{j \in d'} f_j w_j^g w_j w_i^g + \varepsilon \\
&= \sum_{j \in d'} f_j \omega_j^g + \varepsilon
\end{aligned} \tag{2.32}$$

■

Theorem 2.2.3 — Approximate elimination. Let $\mathscr{F}$ be a set of features $\mathscr{F} = \{f_1, \ldots, f_d\}$ and $\mathscr{F}'$ be a subset of $\mathscr{F}$ such that $f_i \notin \mathscr{F}'$. Let g and h be two nonlinear models with equivalent capacity. If the feature f_i can be written as a function of $\mathscr{F}'$ through h with an error term ε, then the upper bound of the training error obtained by fitting g on $\mathscr{F}'$ instead of $\mathscr{F}' \cup \{f_i\}$ is at most $\eta(g, \varepsilon)$.

Proof. By definition f_i can be written as a nonlinear function h of the feature set $\mathscr{F}'$ with an additional term ε, hence:

$$f_i = h(F') + \varepsilon \tag{2.33}$$

where $F' \in \mathbb{R}^{n \times d'}$ is a matrix whose columns are features in $\mathscr{F}'$.

A nonlinear model g trained using the matrix F' can be written as:

$$\widehat{y} = g(F') \tag{2.34}$$

Let F'' be the matrix whose columns correspond to the features in $\mathscr{F}' \cup \{f_i\}$, then the model g trained on F'' can be written as:

$$\widehat{y} = g(F'') = g(F', f_i) = g(F', h(F') + \varepsilon) \tag{2.35}$$

Since the information in $h(F')$ can be obtained from F' by applying the function h, then g can be fitted on F' without information loss by discarding $h(F')$:

$$\widehat{y} = g(F'') = g(F', \varepsilon) = g(F') + \eta(g, \varepsilon) \tag{2.36}$$

where η is a function of g and ε. ■

The following definition can be used to monitor in an unsupervised way the performance loss when multiple features are recursively eliminated. In some applications, this lemma may be used to derive alternative stopping conditions.

Definition 2.2.1 — Validity of feature elimination. Let λ be an upper bound of the performance loss required for a specific application and let $\{\eta_1, \ldots, \eta_k\}$ be a sequence of training errors obtained by performing k steps of feature elimination. The sequence of k feature elimination steps is valid if and only if $\lambda \leq \sum_{i=1}^{k} \eta_i$.

Experimental Evaluation of Predictable Feature Elimination

Experimental setup

The performance of predictable feature elimination is compared over a 10-fold cross-validation against both supervised and unsupervised feature selection algorithms [Li+18; Ped+11b]: Laplacian score for feature selection (`lap_score` [HCN06]), spectral feature selection for unsupervised clustering (`SPEC` [ZL07]), multi-cluster feature selection (`MCFS` [CZH10]), non-negative spectral feature selection (`NDFS` [Li+12]), regularised discriminative feature selection (`UDFS` [Yan+11]), and recursive feature elimination (`RFE` [Guy+02]). For each fold, each algorithm is used to select a subset of the original features.

`Ridge` classifier is used as the internal estimator for RFE. Equivalently, PFE uses an instance of `Ridge` regressor to discard redundant features. This model (i.e. `Ridge`) is chosen both for its high train speed and its generalization ability in a variety of experimental settings. For all the experiments λ has been set to 0.9. PFE is set in order to remove at most half of the original features. For the comparison to be fair, all the other feature selection algorithms are set in order to provide for each fold the same number of features chosen by PFE.

All the necessary code for the experiments has been implemented in Python 3, relying upon open-source libraries [Li+18; Ped+11b]. In order to generate reproducible results, all algorithms that exploit pseudo-random elements in their training process have been set with a fixed seed. Unless differently specified, each algorithm uses its default parameters as defined in [Li+18; Ped+11b]. Each dataset has been standardized removing the mean and scaling to unit variance (`StandardScaler` [ZWC11]). All the experiments have been run on the same machine: an AMD EPYC 7301 16-Core Processor at 2 GHz equipped with 64 GB memory.

Redundancy detection

The Madelon dataset proposed in [Guy03] was specifically designed to challenge feature selection algorithms in detecting redundant features. Features generated by MADELON can be informative (d_i), repeated (d_r), or redundant (d_c). The algorithm generates clusters of points normally distributed about vertices of an hypercube in a subspace of dimension d_i and assigns an equal number of clusters to each class. Then it stacks d_c linear combinations of the informative features followed by d_r duplicates, drawn randomly with replacement from the informative and redundant features. All the remaining features (d_n) are random noise ($d_n = d - d_i - d_c - d_r$). This benchmark dataset is used to assess the ability of PFE in detecting redundant features. For this experiment the number of informative features is set to 150 as well as the number of redundant and repeated features ($d_i = d_c = d_s = 150$). The total number of features is set to $d = 500$, thus $d_n = 50$ features are just random noise. The task is to detect informative, redundant, duplicate, and noisy features in order to correctly classify clusters of samples on hypercube vertices. Experimental results are shown in Figure 2.18. Once feature selection algorithms are fitted on a training fold, an instance of `RidgeClassifier` and of `DecisionTreeClassifier` are used to assess the quality of the selection

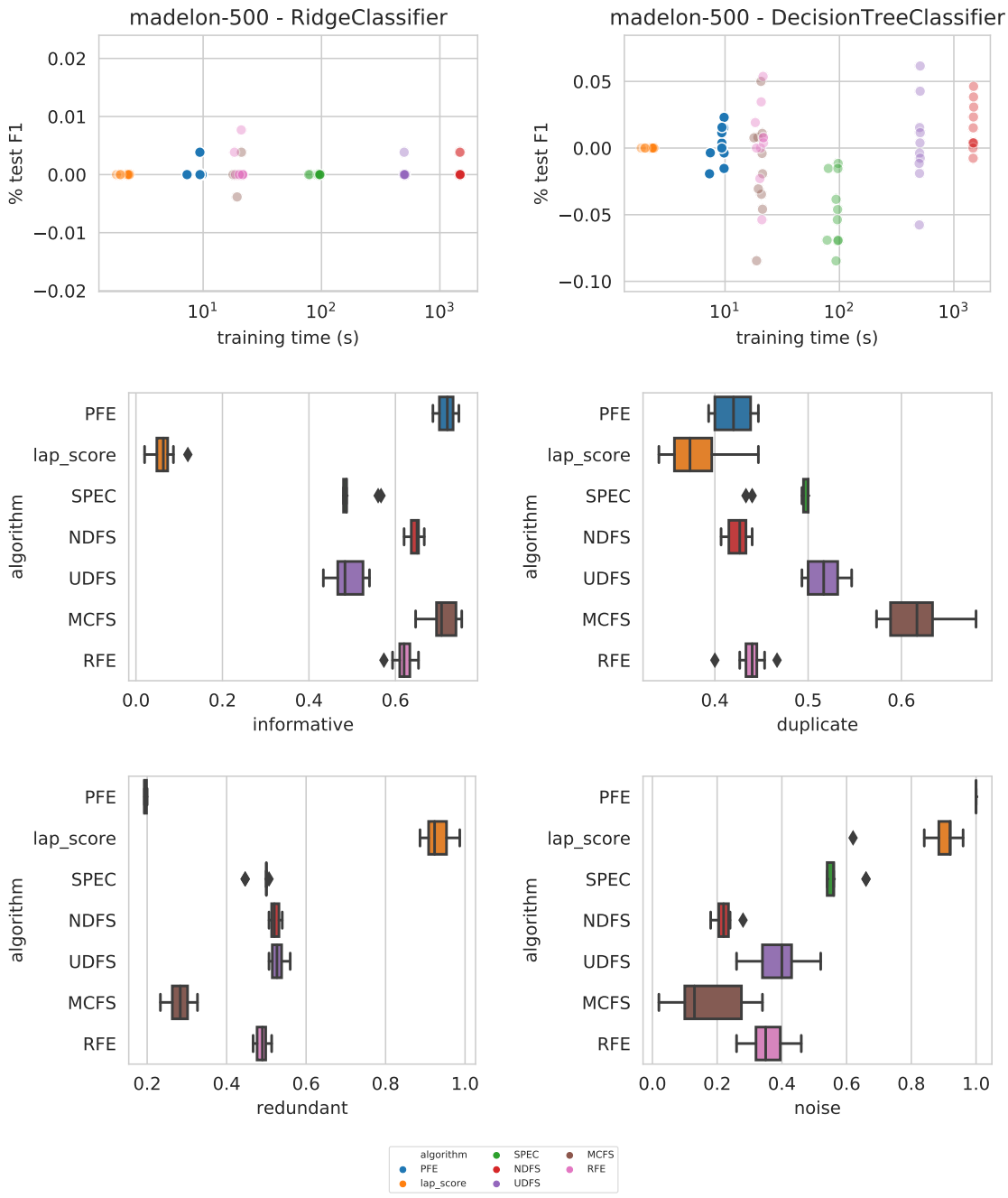


Figure 2.18: Classification performances on the Madelon dataset (top two); features selected for the Madelon dataset (bottom four).

on the test set. The resulting $F1$ score [Van79] is then compared to the one obtained by training on the same fold but using all the original features. In this way, the performance of all techniques can be evaluated with respect to a fair baseline (see Figure 2.18, top). Except for `lap_score`, PFE resulted as the fastest approach. The most interesting result of the simulation is represented by boxplots

in Figure 2.18, bottom. They show for each kind of feature (informative, redundant, duplicate, and noisy) the percentage of features retained by each algorithm. Notably, PFE and `MCFS` preserve most of the informative features while discarding most of the redundancy. However, `MCFS` is the worst technique in terms of duplicate detection, whereas PFE is the best one together with `NDFS` and `lap_score`. It should be remarked, anyway, that PFE retains *all* noisy features. Yet, it is not surprising at all as the approach is not designed to get rid of random noise. In authors view, PFE is not meant to be used alone but combined with other complementary feature selection approaches.

Cross-talk benchmarks

The ability of feature selection algorithms in tackling different kind of machine learning problems is assessed using four benchmark datasets taken from the OpenML[Van+13]. Table 2.5 highlights the main characteristics of the four datasets. The first two (gas-drift and isolet) are used to test

Table 2.5: Benchmark datasets.

dataset	samples	features	classes	reference
gas-drift	13,910	128	6	[Ver+12]
isolet	7,797	617	26	[FC91]
Mercedes	4,209	377	-	[Eri+20]
crime	1,994	127	-	[Tur+11]

classification and clustering performances while the latters are employed for regression. The quality of the selections is assessed over a 10-fold cross-validation. Once fitted on a training fold, each algorithm provides a selection of the original features to an instance of a machine learning model on the filtered training set only. `RidgeClassifier` and `DecisionTreeClassifier` are used to evaluate classification performance in terms of $F1$ score. `AgglomerativeClustering` [War63] and `KMeans` [Ste56] are employed to assess clustering performances through the Silhouette coefficient [Rou87]. The number of centroids for k-means is chosen as twice as much as the number of classes. For regression `Ridge` and `DecisionTreeRegressor` are used to measure the coefficient of determination ($R2$ score, [ST+60]). Once collected, performance scores are compared to the ones obtained by training on the same folds but using all the original features. In this way, the performance of all techniques can be evaluated with respect to a fair baseline. Figures 2.19, 2.20, and 2.21 show the results in terms of performance metrics and training time. As mentioned before, all the other feature selection algorithms are set in order to provide for each fold the same number of features chosen by PFE, thus yielding a fair comparison.

Compared to state-of-the-art techniques, PFE is among the fastest solutions together with `RFE` and `lap_score`. Notably, `RFE` is not used for clustering as it is a supervised algorithm, thus it cannot be employed for unsupervised tasks. Despite its unsupervised nature, PFE often matches `RFE` performances and sometimes provides even better solutions (i.e. Mercedes dataset using `DecisionTreeRegressor`). The efficiency of PFE with respect to other unsupervised approaches is revealed on the largest dataset (gas-drift) where it is faster by a few order of magnitudes. Moreover, even when PFE performances appear to be slightly worse than others (i.e. Mercedes using `Ridge`), it may be sufficient to change the downstream predictor (i.e. the performance looks much better when `DecisionTreeRegressor` is used). Indeed, by construction, PFE performs feature selection such that the information loss is almost negligible.

In summary, experimental results prove PFE to be competitive with state-of-the-art feature

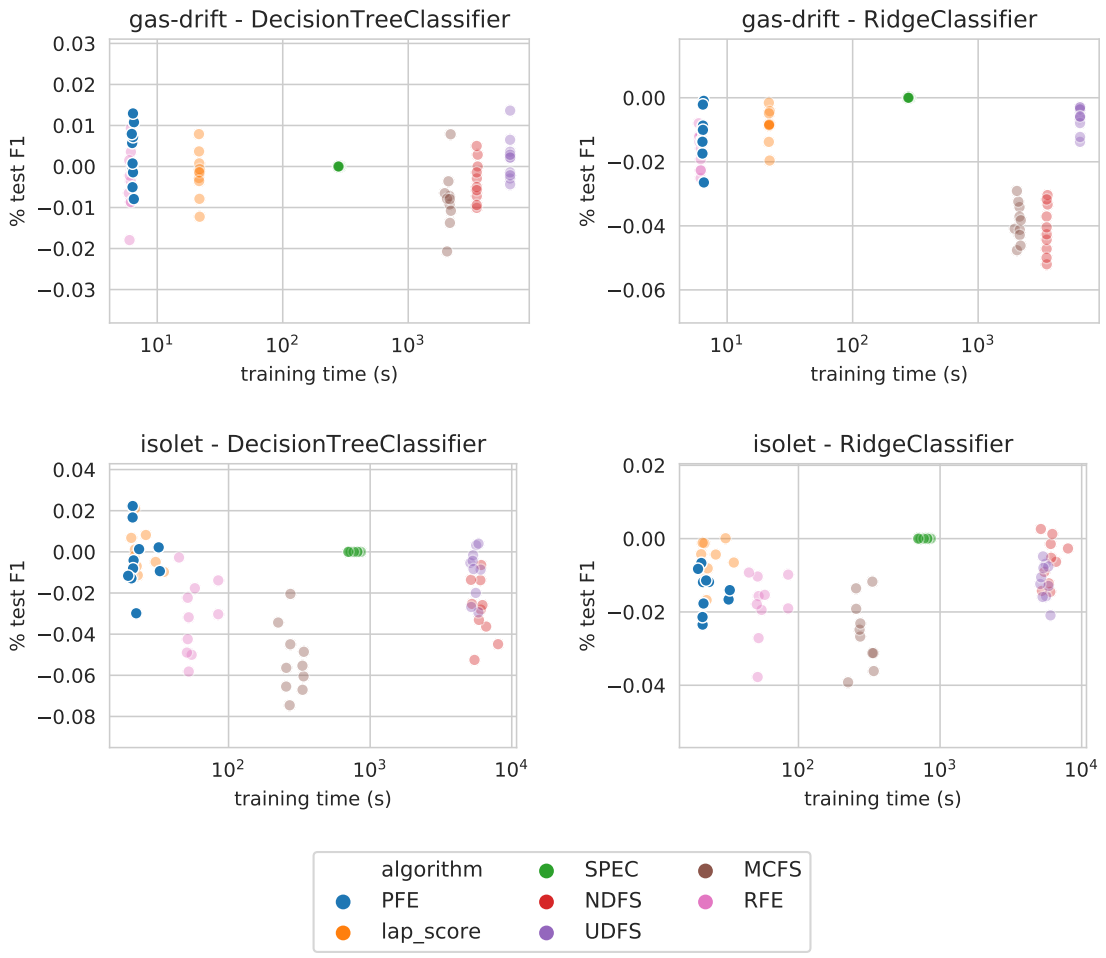


Figure 2.19: Classification results on benchmark datasets.

selection algorithms, on a set of non-trivial classification, regression, and clustering benchmarks. The main drawback of the approach is the impossibility of removing uninformative, but hard-to-predict features, for example those including completely random values: In most cases, however, such features are filtered out by subsequent machine learning algorithms applied to the data, as they have no correlation with the objectives.

2.2.2 Coreset discovery

The concept of *coreset*, originally from computational geometry, has been redefined in the field of machine learning (ML) as the minimal number of input samples from which a ML algorithm can obtain a good approximation of the behavior it would have on the whole original dataset. In other words, a coreset represents a fundamental subset of an available training set, that is sufficient for a given ML algorithm to obtain a good performance, or even the same performance it would have if trained on the whole training set [BLK17]. This definition is intentionally generic, as it encompasses different tasks, ranging from classification, to regression, to clustering, that entail different measures of performance. The practical applications of coresets range from speeding up training time, to

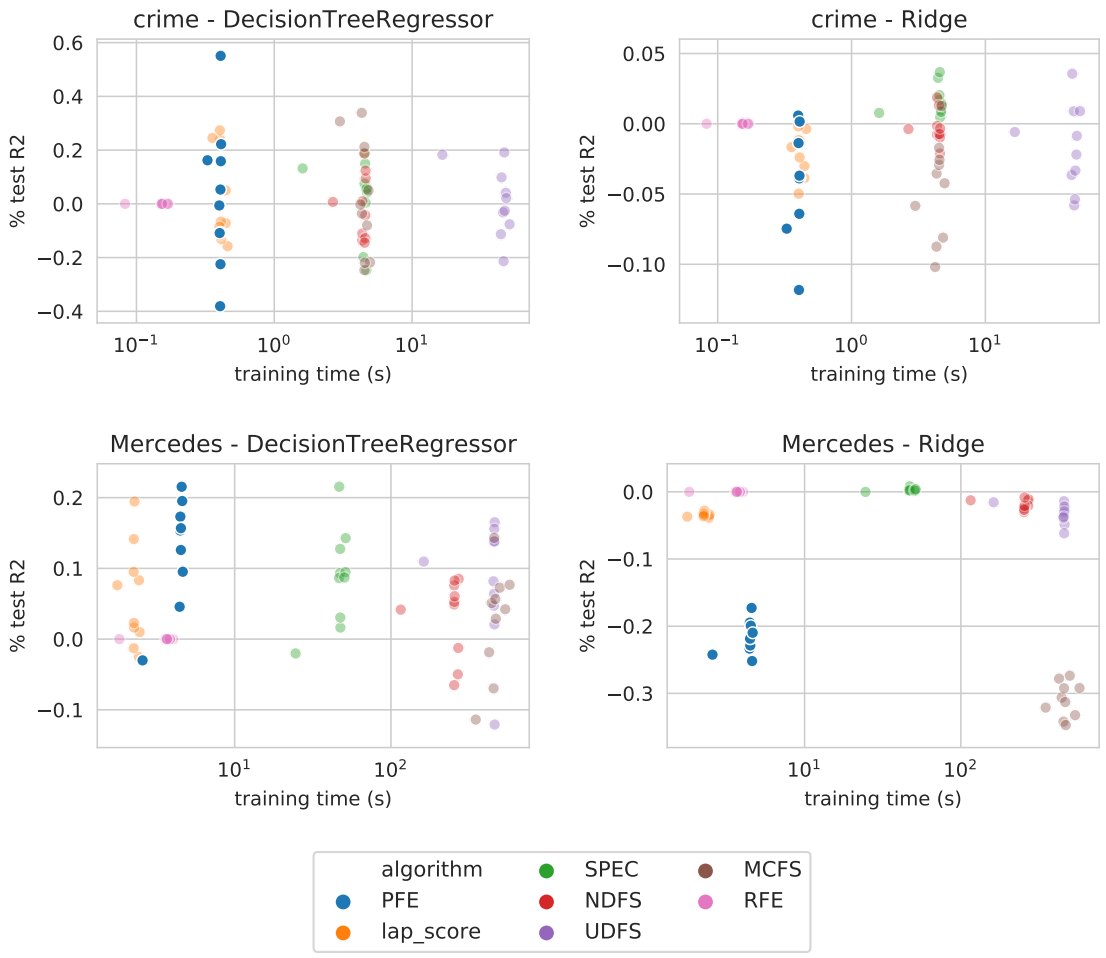


Figure 2.20: Regression results on benchmark datasets.

obtaining a better understanding of the data itself, making it possible for human experts to analyze only a few data points, instead of dealing with prohibitive amounts of samples.

Discovering coresets for a specific task is an open research line, and specialized ML literature reveals a considerable number of approaches, among which the most popular are Bayesian Logistic Regression (BLR, [HCB16]), Greedy Iterative Geodesic Ascent (GIGA, [CB18]), Frank-Wolfe (FW, [Cla10]), Forward Stagewise (FSW, [M A60]), Least-angle regression (LAR, [Efr+04][BDM13]), Matching Pursuit (MP, [MZ93]), and Orthogonal Matching Pursuit (OMP, [PRK93]). Interestingly, very often such algorithms require the user to specify the desired number N of points in the coreset; or assume, for simplicity, that a good coreset is independent from the task and/or the ML algorithm selected for that task. However, the problem of finding a coreset can be intuitively framed as multi-objective, as the quality of the results is probably dependent on the number of points included in the coreset; and minimizing the number of selected points goes against maximizing performance. Moreover, it also seems likely that different ML algorithms would actually need coresets of different size and shape to operate at the best of their possibilities.

Starting from these two intuitions, an evolutionary approach to coreset discovery for classification

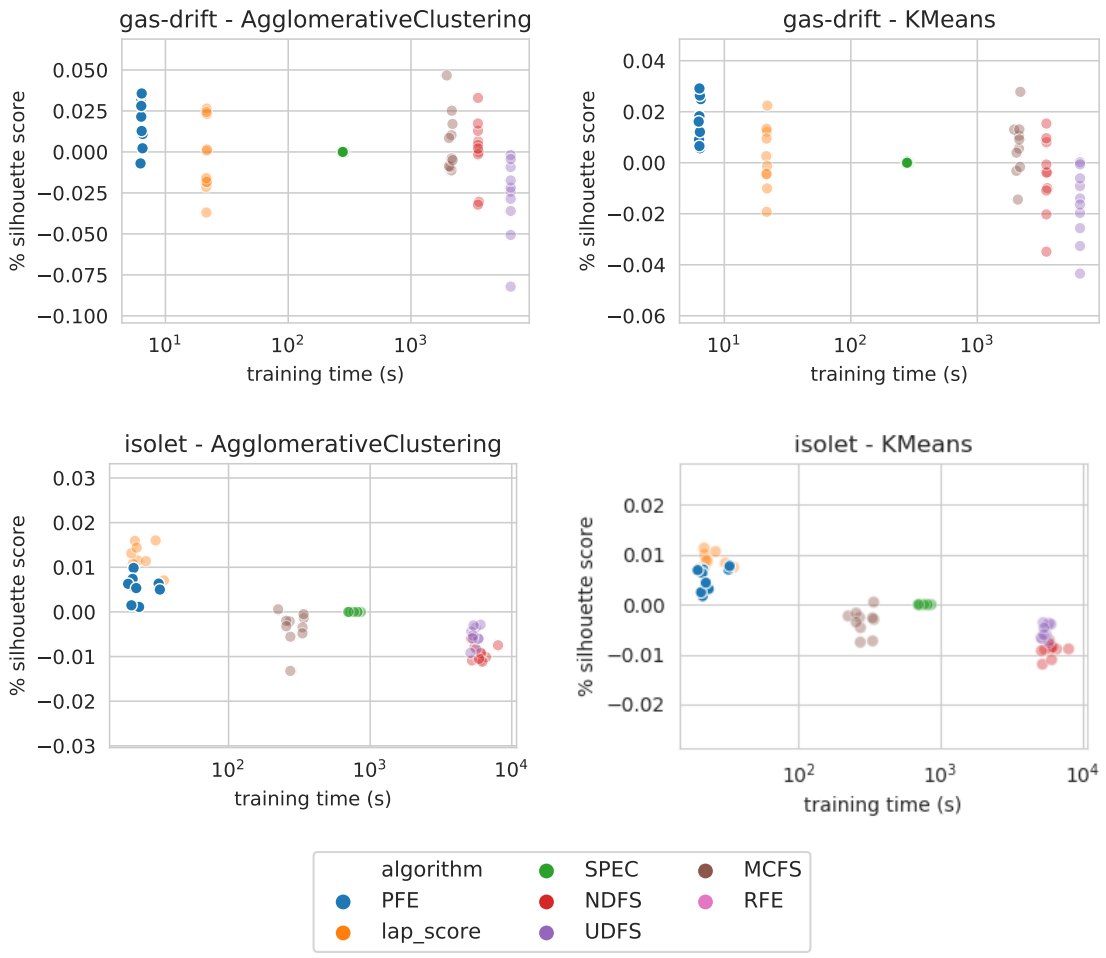


Figure 2.21: Clustering results on benchmark datasets.

tasks is introduced in this section. Starting from a given training set, a state-of-the-art multi-objective evolutionary algorithm, NSGA-II [Deb+02], is set to find the coresets representing the best trade-offs between amount of points (to be minimized) and classifier accuracy (to be maximized), for a specific classification algorithm. The resulting Pareto front will then represent several coresets, each one a different optimal compromise beteween the objectives, and a human expert will then be able to not only select the coreset more suited for their needs, but also obtain extra information on the ML algorithm's behavior from observing its decrease in performance as the number of coreset points decreases.

Experimental results on two iconic classification benchmarks show that the proposed approach is able to best several state-of-the-art coreset discovery algorithms in literature, obtaining results that also allow the classifier to generalize better on an unseen test set, belonging to the same benchmark. Moreover, a meta-analysis of the results on different classifiers shows that indeed, while some of the points selected are common to different algorithms, the choice of points in the coreset is heavily dependent on the classifier selected for the classification task, with similarities between classifiers belonging to the same families.

Coresets

Coresets were originally studied in the context of computational geometry, and defined as a small set of points that approximates the shape of a larger point set. In ML the definition of coreset is extended to intend a subset of the input samples, such that a good approximation to the original input can be obtained by solving the optimization problem directly on the coreset, rather than on the whole original set of input samples [BLK17].

Finding coresets for ML problems is an active line of research, with applications ranging from speeding up training of algorithms on large datasets [TKC05] to gaining a better understanding of the algorithm's behavior. Unsurprisingly, a considerable number of approaches to coreset discovery can be found in the specialized literature. In the following, a few of the main algorithms in the field, that will be used as a reference during the experiments, are briefly summarized: Bayesian Logistic Regression (BLR, [HCB16]), Greedy Iterative Geodesic Ascent (GIGA, [CB18]), Frank-Wolfe (FW, [Cla10]), Forward Stagewise (FSW, [M A60]), Least-angle regression (LAR, [Efr+04][BDM13]), Matching Pursuit (MP, [MZ93]) and Orthogonal Matching Pursuit (OMP, [PRK93]).

BLR is based on the idea that finding the optimal coreset is too expensive. In order to overcome this issue, the authors use a k-clustering algorithm to obtain a compact representation of the data set. In particular, they claim that samples that are bunched together could be represented by a smaller set of points, while samples that are far from other data have a larger effect on inferences. Therefore, the BLR coreset is composed of few samples coming from tight clusters plus the outliers.

The original FW algorithm applies in the context of maximizing a concave function within a feasible region by means of a local linear approximation. In Section 2.2.2, the Bayesian implementation of the FW algorithm designed for core set discovery is considered. This technique, described in [CB17], aims to find a linear combination of approximated likelihoods (which depends on the core set samples) that is similar to the full likelihood as much as possible.

GIGA is a greedy algorithm that further improves FW. In [CB18], the authors show that computing the residual error between the full and the approximated likelihoods by using a geodesic alignment guarantees a lower upper bound to the error at the same computational cost.

FSW [M A60], LAR [Efr+04][BDM13], MP [MZ93] and OMP [PRK93] were all originally devised as greedy algorithms for dimensionality reduction. The simplest is FSW which projects high-dimensional data in a lower dimensional space by selecting one at a time the feature whose inclusion in the model gives the most statistically significant improvement. MP instead includes features having the highest inner product with a target signal, while its improved version OMP at each step carries an orthogonal projection out. Similarly, LAR increases the weight of each feature in the direction equiangular to each one's correlations with the target signal. All these procedures could be applied to the transpose problem of feature selection, that is approximation of core sets.

Very often these algorithms start from the assumption that the coreset for a given dataset will be independent from the ML pipeline used. This premise might not always be correct, as the optimization problem underlying, for example, a classification task, might vary considerably depending on the algorithm used. It is also important to notice that the problem of finding the coreset, given a specific dataset and an application, can be naturally expressed as multi-objective: on the one hand, the user wishes to identify a set of core points as small as possible; but on the other hand, the performance of the algorithm trained on the coreset should not differ from its starting performance, when trained on the original dataset. For this reason, multi-objective evolutionary algorithms could be well-suited to this task.

Multi-objective evolutionary algorithms

When dealing with problems that feature contrasting objectives, there is no single optimal solution to be found. Each candidate solution, in fact, represents a different compromise between conflicting aims. Nevertheless, it is still possible to find *optimal trade-offs*, for which it is not possible to improve an objective without degrading the others. The set of such optimal compromises is called Pareto front. Multi-objective evolutionary algorithms (MOEA) have been particularly successful in tackling problems with contradictory objectives and obtaining good approximations of the Pareto front. One of the most known MOEAs is the Non-Sorting Genetic Algorithm II (NSGA-II) [Deb+02], that exploits a crowding mechanism to evenly spread candidate solutions on the Pareto front, a procedure that is considerably efficient for problems with 2-3 objectives.

Proposed Approach for Coreset Discovery

The proposed approach is to exploit evolutionary computation to explore the space of all subsets of samples in a given training dataset in a classification task, searching for those subset that do not reduce classification accuracy. Such samples would then represent a coreset, a collection of points that is necessary and sufficient to correctly define the decision boundaries for a target classifier, given a target dataset.

A candidate solution in the problem is a set of samples to be kept from the training set. The genome of an individual is thus defined as a binary array of size equal to the training set, with a '1' in a given position meaning that the sample corresponding to that index will become part of the coreset, and a '0' meaning that the sample will be removed.

Intuitively, it is easier to maintain a classifier's accuracy while keeping a large number of samples from its training set, rather than just a small quantity. For this reason, it is more appropriate to frame the optimization problem as multi-objective, with the conflicting aims of *i. minimizing the number of samples in the coreset*, and *ii. maximizing classifier's accuracy on the whole dataset*. The first objective is measured by simply counting the number of '1's, i.e. the number of samples in the coreset, in each candidate solution. For the second objective, the target classifier is trained on the reduced training set, and its accuracy is then tested on the whole set of training points, including all samples removed from the training set by the candidate solution. A scheme of the individual evaluation is presented in Figure 2.22.

Experimental evaluation

In order to empirically validate the proposed approach, the methodology is evaluated with 6 algorithms, chosen to be representative of both hyperplane-based and ensemble, tree-based classifiers: `BaggingClassifier` [Bre99], `GradientBoostingClassifier` [Fri01], `LogisticRegression` [Cox58], `RandomForestClassifier` [Bre01b], `RidgeClassifier` [Tik43], `SVC` (Support Vector Machines) [Hea+98]. All classifiers are implemented in the `scikit-learn`[5] [Ped+11a] Python module and use default parameters. A fixed seed has been set for all those that exploit pseudo-random elements in their training process, such as `RandomForestClassifier`, as for the stated objective it is important that the classifier will follow the same training steps, albeit with a reduced training set. A similar result could have been obtained by repeating multiple times the training process of classifiers containing random elements. In this first batch of experiments, that constitute the proof of concept, the former option is selected to reduce computational effort. The implementation of NSGA-II, used during the evolutionary process, is taken from the `inspyred`[6] [Gar12] Python module. NSGA-II uses a

[5] scikit-learn: Machine Learning in Python, `http://scikit-learn.org/stable/`

[6] inspyred: Bio-inspired Algorithms in Python, `https://pythonhosted.org/inspyred/`

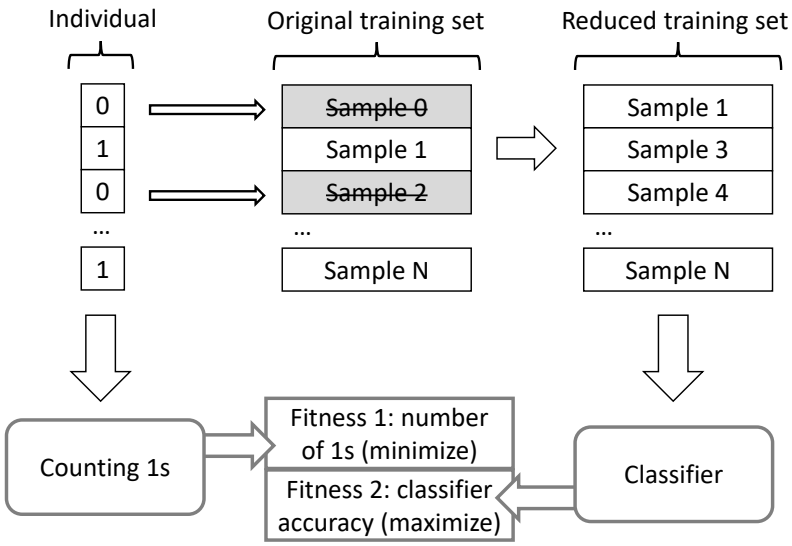


Figure 2.22: Scheme of the proposed fitness evaluation. A given dataset is randomly split between training and test set. Candidate solutions, encoded as binary strings, are used to reduce the training set, by removing samples whose index corresponds to a '0' in the binary string. The target classifier is then trained on the reduced training set, while its accuracy is measured on all the original training samples.

binary tournament selection, $\mu = 100$, $\lambda = 100$, a one-point crossover, a bit-flip mutation, and a stop condition set at 100 generations. Each individual can remove between 1 and 90 samples from the training dataset, so valid candidate genomes can contain between 1 and 90 '0's.

The results obtained by the proposed approach are then compared against the 7 coreset discovery algorithms BLR [HCB16], GIGA [CB18], FW [Cla10], MP [PRK93], OMP [PRK93], LAR [Efr+04][BDM13], and FSW [M A60], described in more detail in subsection 2.2.2. Some of the algorithms require the user to specify the number N of desired points in the coreset: in order to provide a fair comparison, N is set to the size of the highest-accuracy coreset found by the proposed approach in the corresponding experiment.

The first two experimental runs are performed on the well-known *Iris* dataset [Fis36], comprising 150 samples from 3 different classes. The samples are randomly split between a 99-sample training set and a 51-sample test set. As shown in Table 2.6, all considered classifiers perform reasonably well on the dataset, as Iris is a benchmark that presents no particular difficulty for most algorithms. In a first batch of experiments (Iris-2), only two features of the dataset are considered, in order to offer the reader a more intuitive visual assessment of the results. The datasets are shown in Figure 2.23. In further experiments, the proposed approach is tested on *Moons*, a synthetic data set composed of two interleaving distributions having an half circle shape (2 classes, 400 samples, 2 features) and *Credit*, a dataset evaluating credit risk (2 classes, 1000 samples, 20 features). For *Credit*, a human-readable analysis of the results is also presented. The code used in this work is freely available in the BitBucket repository `https://bitbucket.org/evomlteam/evolutionary-core-sets`.

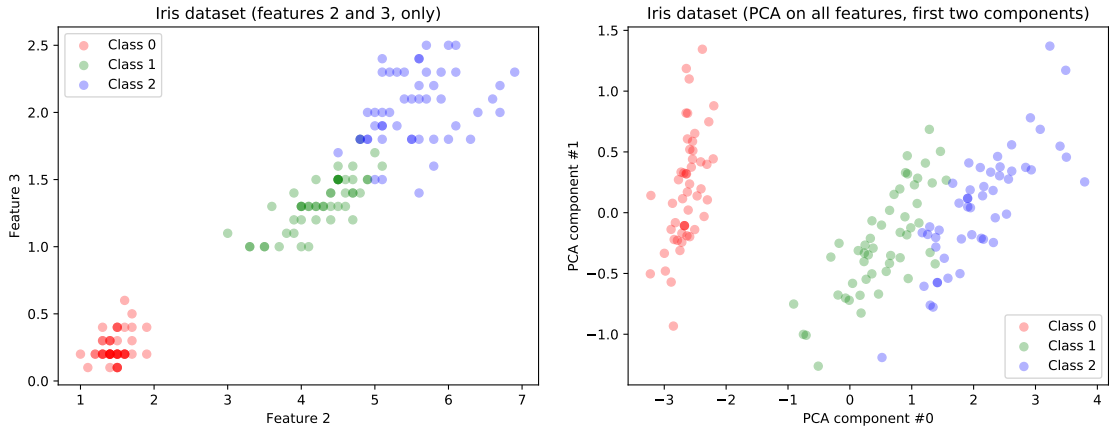


Figure 2.23: Samples from the Iris dataset **(left)** visualized considering only features 2 and 3, **(right)** visualized considering the first two components of a PCA performed on all features.

Experiments on 2-feature Iris

The first column of Figures 2.24 and 2.25 shows the Pareto front obtained by the proposed approach for all the classifiers on Iris-2. It is noticeable how the trade-offs found are different in both size and accuracy, depending on the considered algorithm. The second and third columns of the figures portray the frequency of appearance of samples in the training set inside the candidate solutions on the Pareto fronts of the corresponding experiment. It is easy to observe that several points appear among all candidate coresets found in the same experiment.

Table 2.7 presents the accuracy of the most accurate coreset found by the proposed approach,

Table 2.6: Initial performance (classification accuracy) of the considered classifiers on the Iris dataset (using two features and all, respectively).

	Iris (2 features)			Iris (all features)		
Classifier	**Overall**	**Training**	**Test**	**Overall**	**Training**	**Test**
GradientBoostingClassifier	0.9800	0.9899	0.9608	0.9867	1.0	0.9608
BaggingClassifier	0.9733	0.9899	0.9412	0.9867	1.0	0.9608
LogisticRegression	0.8533	0.8081	0.9412	0.9733	0.9596	1.0
RandomForestClassifier	0.9733	0.9899	0.9412	0.9867	1.0	0.9608
RidgeClassifier	0.8000	0.7677	0.8627	0.8667	0.8586	0.8824
SVC	0.9733	0.9697	0.9804	0.9733	0.9697	0.9804

Table 2.7: Test performance of the considered classifiers and coreset algorithms on the Iris-2 dataset.

	Training	EvoCore	BLR	GIGA	FW	MP	OMP	FSW	LAR
BaggingClassifier	0.9798	1.0000	0.8824	0.8824	0.8039	0.9412	0.9412	0.9412	0.9412
GradientBoostingClassifier	0.9596	0.9804	0.8627	0.8431	0.8824	0.9608	0.9608	0.9608	0.9608
LogisticRegression	0.9697	0.9804	0.6471	0.7059	0.6667	0.9804	0.9804	0.9804	0.9804
RandomForestClassifier	0.9798	0.9804	0.9020	0.8824	0.8039	0.9412	0.9412	0.9412	0.9412
RidgeClassifier	0.9798	0.9608	0.6667	0.6667	0.6667	0.9020	0.9020	0.9020	0.9020
SVC	0.9697	0.9804	0.6667	0.8824	0.8627	0.9412	0.9412	0.9412	0.9412
Average	**0.9731**	**0.9804**	**0.7712**	**0.8105**	**0.7810**	**0.9444**	**0.9444**	**0.9444**	**0.9444**

against coresets discovered by state-of-the-art algorithms in ML literature. Not only the proposed approach outperforms the accuracy of all algorithms on the training set, but it also enables the classifiers to create decision boundaries that generalize better, obtaining improved results on the unseen test set as well.

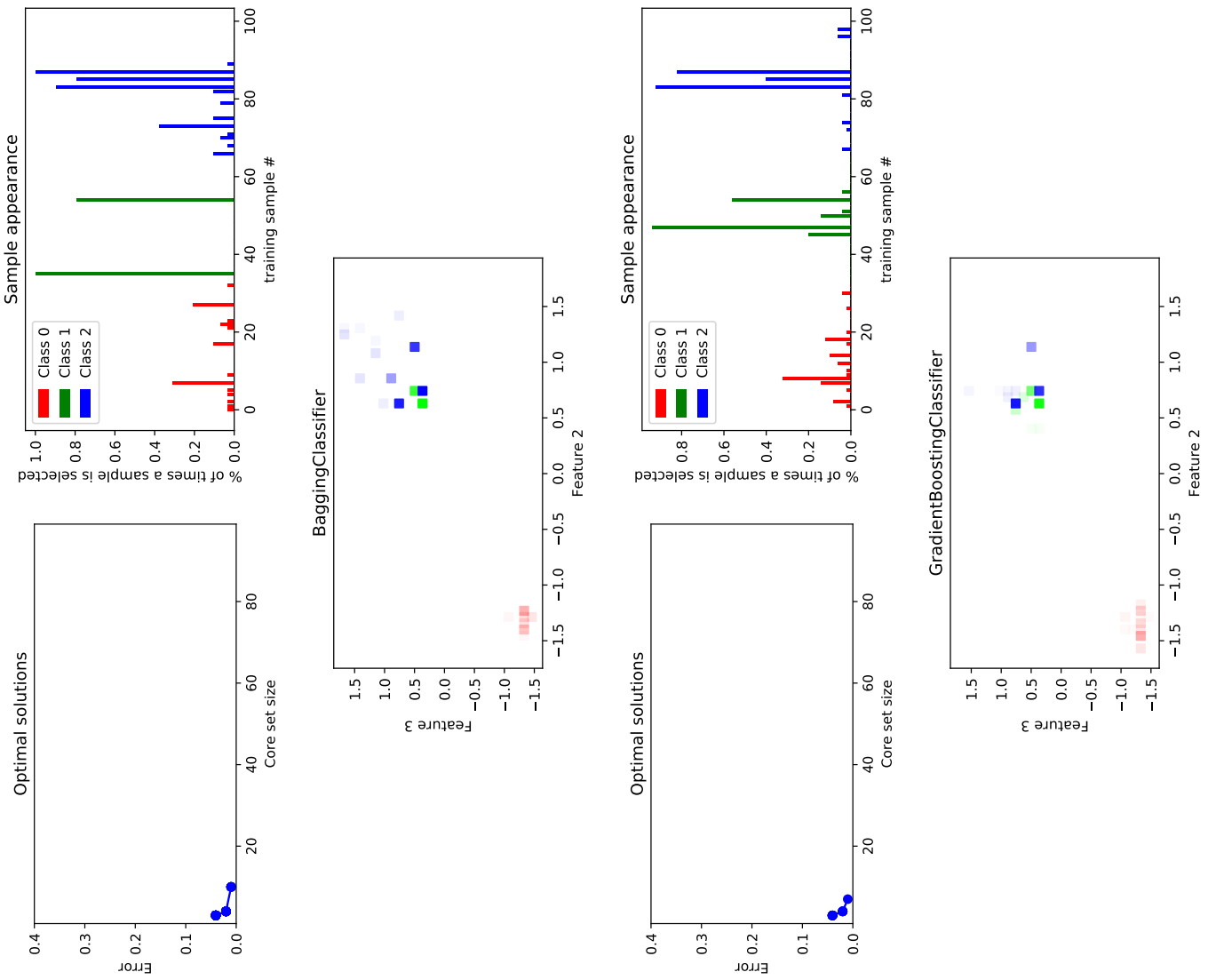
Experiments on 4-feature Iris

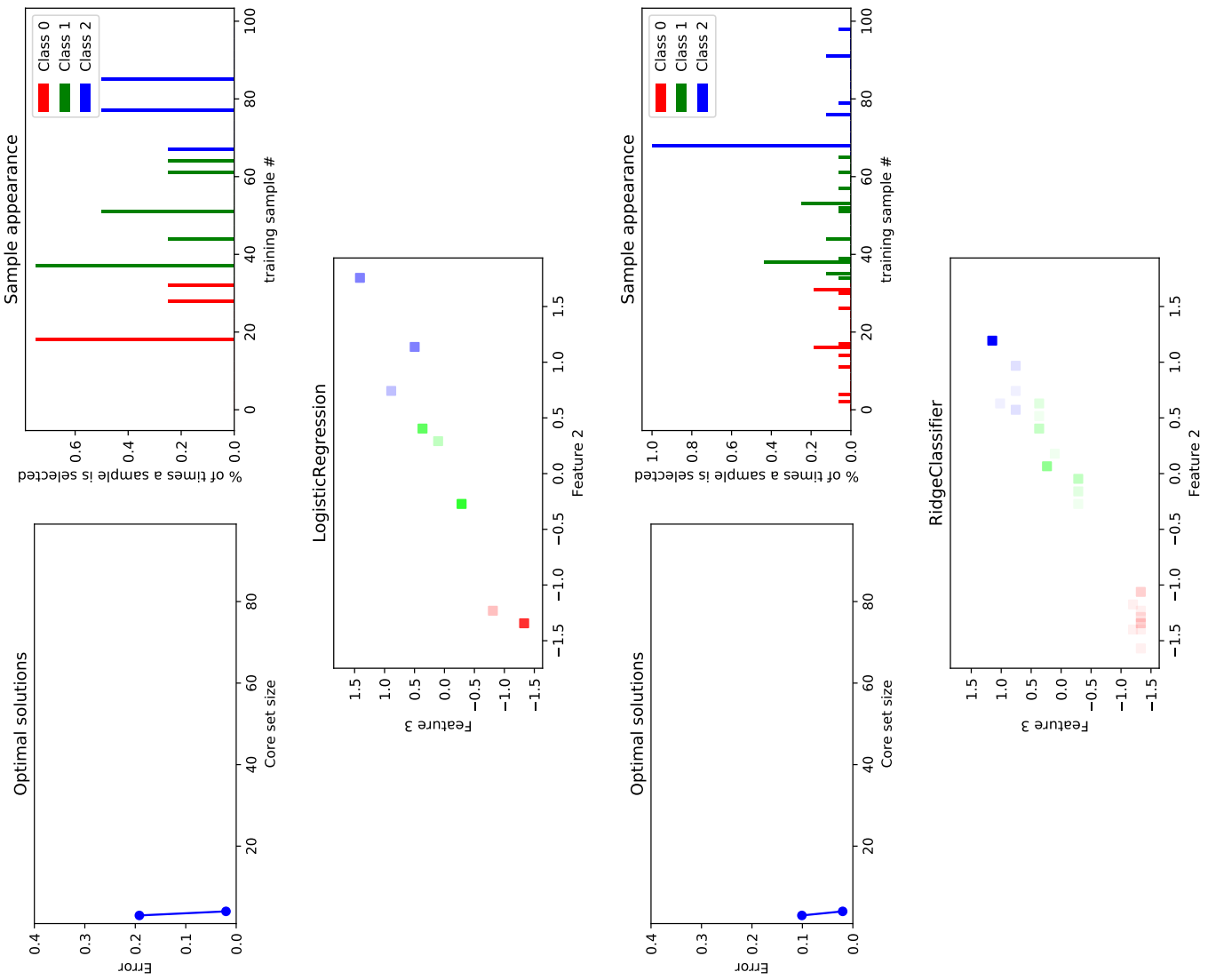
The first column of Figures 2.26 and 2.27 shows the Pareto front obtained by the proposed approach for all the classifiers on Iris-4. Again, the trade-offs found appear different in both size and accuracy, depending on the considered algorithm. As observed for Iris-2, the second and third columns of the figures portraying the frequency of appearance of samples in the training set inside the candidate solutions on the Pareto fronts show several points appearing among all candidate coresets found in the same experiment.

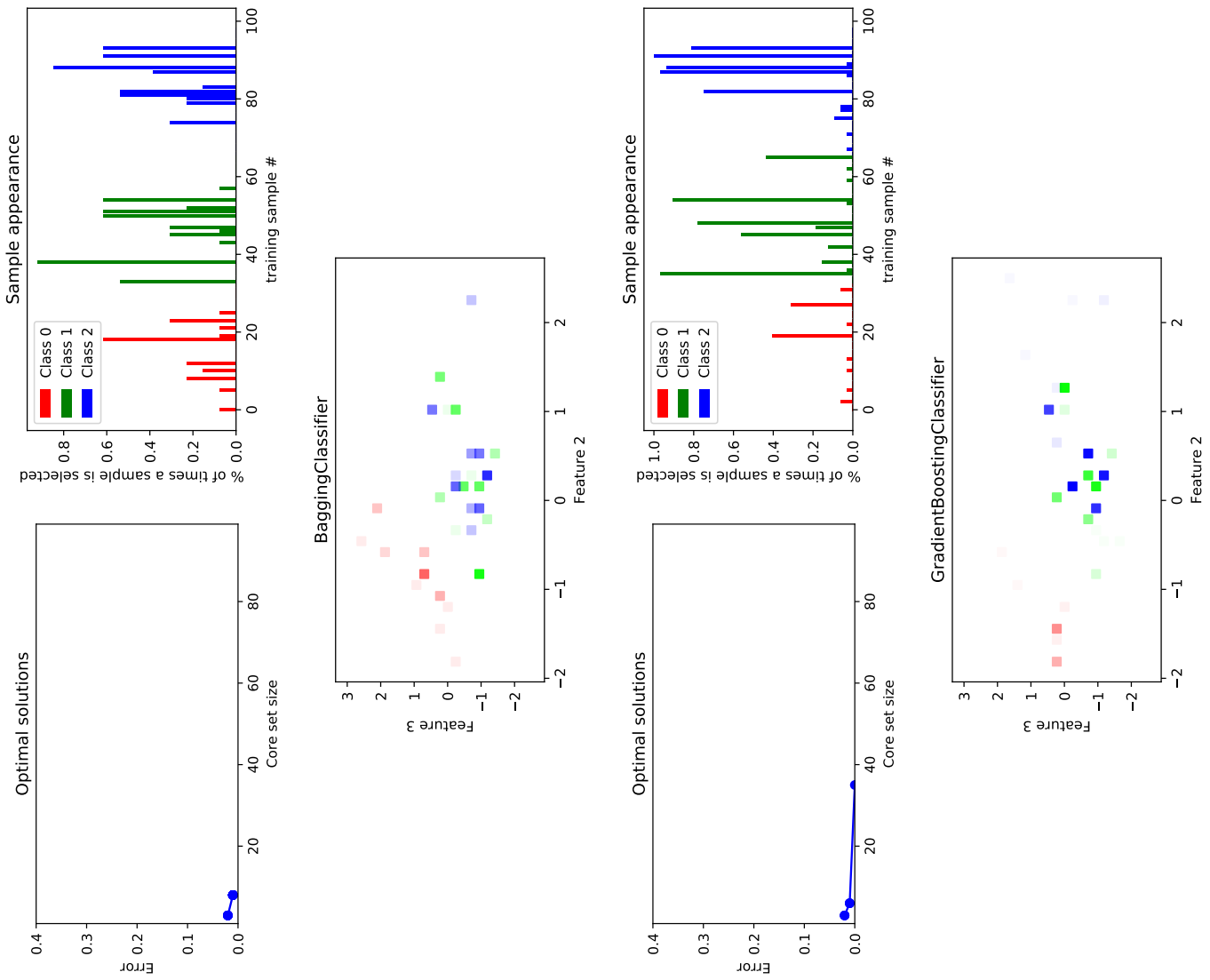
Table 2.8 presents the accuracy of the most accurate coreset found by the proposed approach, against coresets discovered by state-of-the-art algorithms in ML literature. Again, the proposed approach outperforms the accuracy of all algorithms on both the training set and the unseen test set.

Table 2.8: Test performance of the considered classifiers and coreset algorithms on the Iris-4 dataset.

	Training	EvoCore	BLR	GIGA	FW	MP	OMP	FSW	LAR
BaggingClassifier	0.9596	0.9804	0.7059	0.9412	0.8824	0.9608	0.4118	0.8431	0.4118
GradientBoostingClassifier	0.9697	0.9804	0.7647	0.7255	0.8824	0.8824	0.6471	0.8824	0.6471
LogisticRegression	0.9798	1.0000	0.7843	0.8824	0.8039	0.8431	0.8039	0.7059	0.8039
RandomForestClassifier	0.9394	0.9608	0.7255	0.8627	0.9020	0.9804	0.7647	0.7843	0.7647
RidgeClassifier	0.9798	1.0000	0.7255	0.9020	0.8824	0.8824	0.7451	0.7451	0.7451
SVC	0.9697	0.9608	0.7843	0.9608	0.9608	0.9608	0.6275	0.6471	0.6275
Average	**0.9663**	**0.9804**	**0.7484**	**0.8791**	**0.8856**	**0.9183**	**0.6667**	**0.7680**	**0.6667**


Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
Optimal solutions
Error
Core set size
BaggingClassifier
Feature 3
Feature 2
Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
Optimal solutions
Error
Core set size
GradientBoostingClassifier
Feature 3
Feature 2


Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
Optimal solutions
Error
Core set size
LogisticRegression
Feature 3
Feature 2
Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
Optimal solutions
Error
Core set size
RidgeClassifier
Feature 3
Feature 2


Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
Optimal solutions
Error
Core set size
BaggingClassifier
Feature 3
Feature 2
Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
Optimal solutions
Error
Core set size
GradientBoostingClassifier
Feature 3
Feature 2

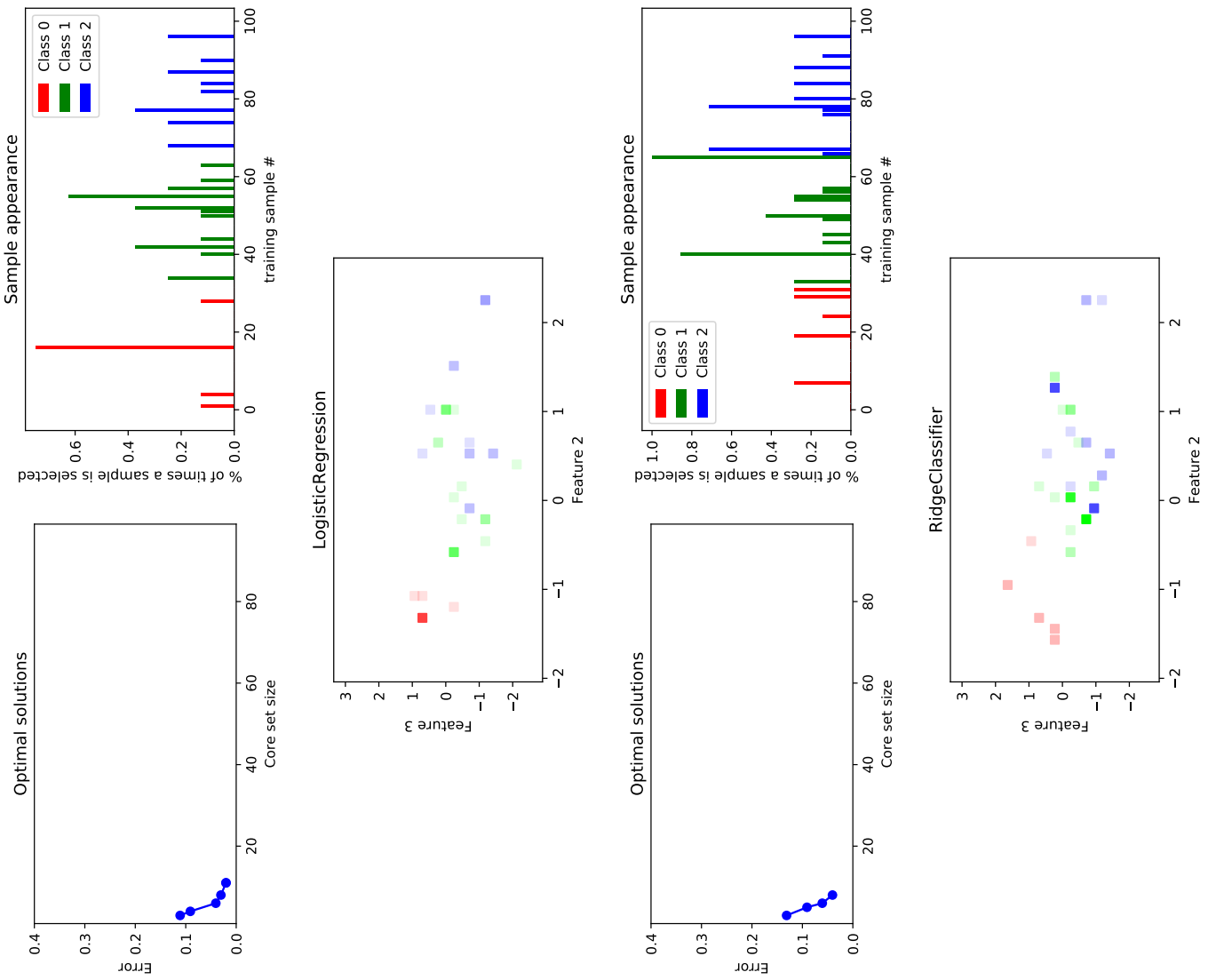

Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
LogisticRegression
Feature 3
Feature 2
Optimal solutions
Error
Core set size
Sample appearance
Class 0
Class 1
Class 2
% of times a sample is selected
training sample #
RidgeClassifier
Feature 3
Feature 2
Optimal solutions
Error
Core set size

Table 2.9: *Moons* data set. Training set size, classification accuracy on an unseen test set and running time (in seconds) for different classifiers exploiting both EvoCore and state-of-the-art algorithms for core set discovery.

	RandomForest			Bagging			SVC			Ridge		
algorithm	size	accuracy	avg time	size	accuracy	avg time	size	accuracy	avg time	size	accuracy	avg time
all samples	266	0.9328	-	2661	0.9254	-	266	0.9179	-	266	0.8134	-
EvoCore	10	**0.9403**	440.2 s	30	**0.9478**	415.9 s	24	**0.9403**	126.6 s	2	**0.8209**	139.8 s
GIGA	2	0.4254	0.01 s	2	0.2463	0.01 s	2	0.4701	0.01 s	2	0.4701	0.01 s
FW	6	0.6493	3.6 s	6	0.6493	3.6 s	6	0.5299	3.6 s	6	0.6866	3.6 s
MP	3	0.5149	4.6 s	3	0.5821	4.6 s	3	0.5896	4.6 s	3	0.6642	4.6 s
FS	2	0.5149	4.3 s	2	0.2313	4.3 s	2	0.6119	4.3 s	2	0.6119	4.3 s
OP	2	0.5149	0.01 s	2	0.2463	0.01 s	2	0.6493	0.01 s	2	0.6493	0.01 s
LAR	3	0.5149	24.2 s	3	0.2388	24.2 s	3	0.5224	24.2 s	3	0.5896	24.2 s

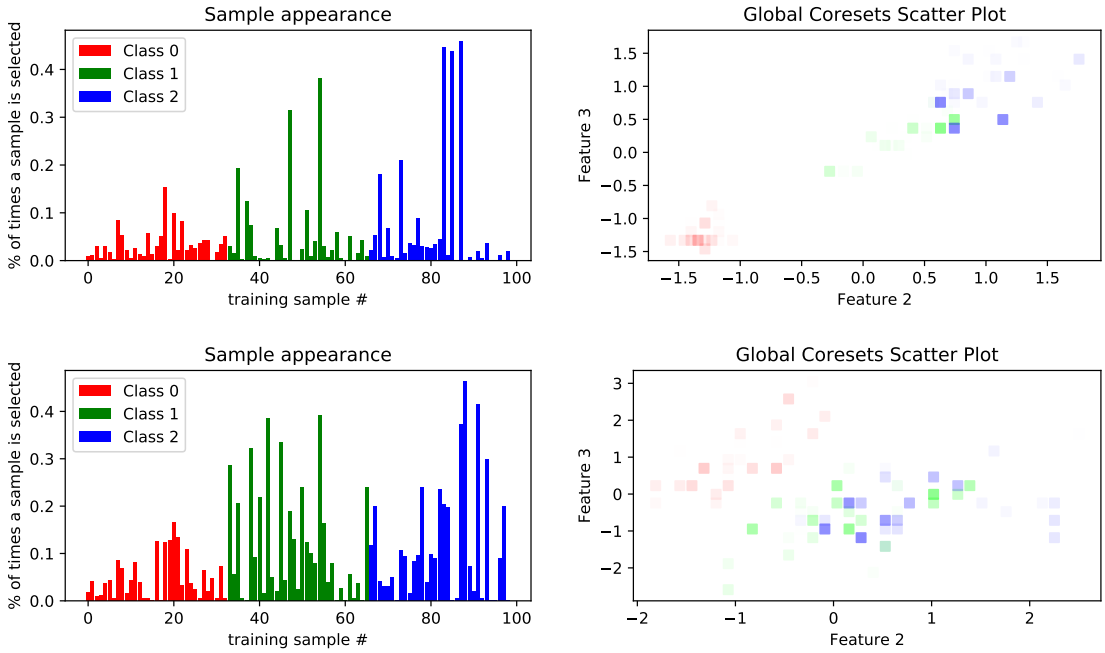


Figure 2.28: Global results, taking into account all candidate solutions found during all experiments on Iris-2 **(top row)** and Iris-4 **(bottom row)**. Bar height in bar plots **(left)** and color saturation in scatter plots **(right)** are directly proportional to the frequency of appearance of a specific sample.

Experiments on Moons

Experimental results on the dataset Moons are summarized in Table 2.9, with decision boundaries for SVC displayed in Figure 2.29. Figure 2.30 reports a meta-analysis of all the Pareto-optimal candidate core sets found by EvoCore, divided by dataset, considering all classifiers. A few samples clearly appear very often among all candidate core sets, while others almost never do, but overall there is a considerable number of samples that are included with low but non-negligible frequency, indicating that different classifiers indeed exploit core sets of different shape.

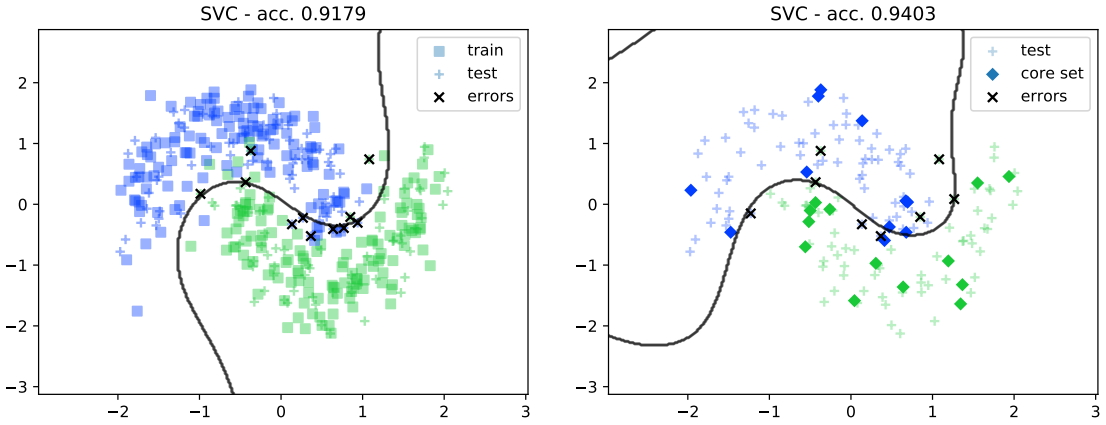


Figure 2.29: Decision boundaries on the Moons data set using all the samples in the training set (Left) and only the core set (Right) for training the classifier. Train samples are represented by squares, test samples by crosses, core samples by black diamonds and test errors by 'x'-shapes.

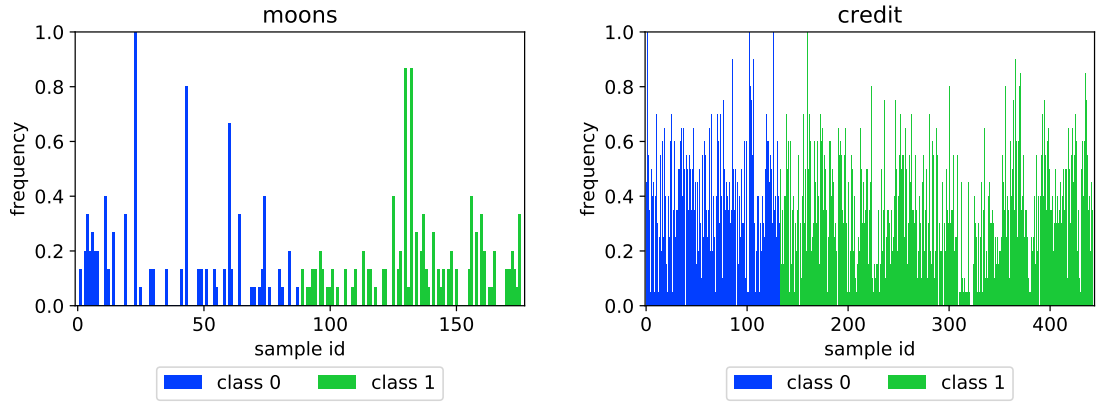


Figure 2.30: Frequency of appearance of samples in the Pareto front solutions of all the classifiers, for the *Moons* and *Credit* datasets.

Experiments on Credit

A summary of the experiments performed on the Credit dataset is reported in Table 2.10. Visualizing the decision boundaries in this case is more difficult, due to the higher dimensionality of the dataset; however, in the following an analysis of the samples most frequently included in the coresets for Credit is presented. Notice from Figure 2.30 that there are four training points that appear in each core set of the Pareto fronts for all the classifiers. Table 2.11 lists the features of the most frequent core samples. Interestingly, such samples represent four different customer profiles. **C1** is a 58-years-old married female. She is asking the bank for relatively small credit for a new radio/tv. Currently, she has 4 existing credits at this bank, but her economical status seems stable (long-term employment, house and real estate of ownership), despite her low disposable income (less than 1,000 DMs). **C2** is an aged single male asking a credit for a new car. He is skilled and well paid (∼2,000 DMs). He has recently changed his job, but he lives in its own house and has a life insurance. Overall, he looks like a responsible customer. **C3** is a young and skilled man. He has rented the house where he lives (with his wife, probably) for the last three years. Despite his young age, he owns a real estate, he already

paid off previous credits with the bank and his disposable income is very high (more than 15,000 DMs). He is currently asking a remarkable credit of more than 3,000 DMs, but he is very rich and the bank will reserve him a kid-glove treatment. **C4** is 32 years old and wants to buy a new radio/tv. He has had the same employment for the past four years and he owns the house where he lives. However, his saving account is nearly empty and he must provide maintenance for two dependants.

Table 2.10: *Credit* data set. Training set size, classification accuracy on an unseen test set and running time (in seconds) for different classifiers exploiting both EvoCore and state-of-the-art algorithms for core set discovery.

	RandomForest			Bagging			SVC			Ridge		
algorithm	size	accuracy	avg time	size	accuracy	avg time	size	accuracy	avg time	size	accuracy	avg time
all samples	666	0.7275		666	0.7066		666	0.7635		666	0.7695	
EvoCore	223	**0.7395**	735	241	0.7006	538	94	0.7335	173	11	0.7485	161
GIGA	137	0.7305	0.11	137	0.7066	0.11	137	0.7096	0.11	137	0.7156	0.11
FW	537	0.6886	0.87	537	0.6856	0.87	537	**0.7635**	0.87	537	**0.7635**	0.87
MP	528	0.6856	1.99	528	0.7036	1.99	528	0.7515	1.99	528	0.7605	1.99
FS	74	0.7006	1.90	74	**0.7186**	1.90	74	0.7305	1.90	74	0.7455	1.90
OP	20	0.7066	0.09	20	0.7036	0.09	20	0.6617	0.09	20	0.6587	0.09
LAR	21	0.6707	0.68	21	0.6886	0.68	21	0.6407	0.68	21	0.6677	0.68

Despite their differences, the low-income aged woman (C1), the middle-class aged man (C2), and the wealthy young man (C3) turn out to be good customers for the bank. They probably represent three "ideal" profiles of good customers, as their characteristics suggest economic stability. On the contrary, the middle-class young (and single) man with two dependants to support and low liquidity seems a representative sample of a risky customer.

Table 2.11: Comparison of the fundamental samples found in the credit risk dataset.

features	C1	C2	C3	C4
checking account (DMs)	<0	no checking	0<x<200	no checking
credit duration (months)	12	24	18	24
credit history	critical/other existing	existing paid	existing paid	existing paid
credit purpose	radio/tv	new car	radio/tv	radio/tv
credit amount	385	2255	3213	1552
savings account (DMs)	<100	unknown	500<x<1000	<100
employment duration (years)	4<x<7	<1	<1	4<x<7
installment rate	4%	4%	1%	3%
personal status	married female	single male	married male	single male
other debtors	none	none	none	none
residence since (years)	3	1	3	1
property magnitude	real estate	life insurance	real estate	car
age	58	54	25	32
other payment plans	none	none	none	bank
housing	own	own	rent	own
existing credits	4	1	1	1
job	unskilled resident	skilled	skilled	skilled
dependants	1	1	1	2
own telephone	yes	no	no	no
foreign worker	yes	yes	yes	yes
target	good	good	good	bad

Discussion

The proposed framework proved to be extremely effective in discovering high-quality coresets for classification. Framing the problem as a function of both application and algorithm seems to be a better method than a more general approach, as highlighted by the differences in the coresets found using different classifiers. Furthermore, obtaining a Pareto front of different compromises, instead of a single solution, might not only provide the user with different viable alternatives, but also provide insight on the behavior of the classifier considered for the task.

Nevertheless, this methodology presents a few drawbacks. The main issue of the approach is computational time. All other coreset discovery algorithms in literature are able to return solutions in a few seconds on regular laptops, while the proposed method typically takes from minutes to tens of minutes. In fact, as the MOEA trains a classifier for each individual evaluation, everything depends on the classifier itself. Among the selected classifiers, a huge variance in training time is noticable between the fastest (`RandomForestClassifier`) and the slowest (`GradientBoostingClassifier`), with corresponding repercussions on the speed of the MOEA. While this problem can be mitigated by parallelizing evaluations, it is arguable that even the current performance is not a great obstacle, as usually coresets are computed once, and then used multiple times. Given that not only the quality of the coresets discovered by the MOEA is higher, but that also several trade-offs are delivered in place of a single solution, it can be claimed that the proposed approach is a preferable alternative to other solutions in literature.

From an overall analysis of the results on the two benchmarks Iris-2 and Iris-4, it is noticeable that, while points frequently appearing in candidate solutions on the Pareto front are generally different between classifiers, algorithms based on decision trees (`RandomForestClassifier`, `GradientBoostingClassifier`, `BaggingClassifier`) and algorithms based on linear hyperplanes (`LogisticRegre-ssion`, `RidgeClassifier`, `SVC`) tend to use similar points, see the second and third column of Figures 2.24, 2.25, 2.26, and 2.27. The two families seem to prefer points on the boundaries between classes (decision trees) or close to the class centroid (linear hyperplanes), respectively. Furthermore, taking into account all coresets found during all the experiments, Figure 2.28 shows how, despite differences between algorithms, a few samples are consistently selected among most of the candidate solutions found. Such samples might have a relevance going beyond the scope of the single technique, being instead excellent representatives of the class they belong to. Not surprisingly, the most common samples for each class are in part close to the class centroid, and in part close to the class boundaries, fully defining the shape of the training samples of the class.

Bibliography

[Aka74] Hirotugu Akaike. “A new look at the statistical model identification”. In: *Selected Papers of Hirotugu Akaike*. Springer, 1974, pages 215–222 (cited on page 21).

[AK15] Naomi Altman and Martin Krzywinski. “Association, correlation and causation”. In: *Nature Methods* 12.10 (Sept. 2015), pages 899–900 (cited on page 30).

[AK17] Naomi Altman and Martin Krzywinski. “Ensemble methods: Bagging and random forests”. In: *Nature Methods* 14.10 (Sept. 2017), pages 933–934 (cited on pages 17, 19).

[AK18] Naomi Altman and Martin Krzywinski. “The curse(s) of dimensionality”. In: *Nature Methods* 15.6 (May 2018), pages 399–400 (cited on pages 17, 18).

[AKR17] Valentin Amrhein, Fränzi Korner-Nievergelt, and Tobias Roth. “The earth is flat ($p > 0.05$): Significance thresholds and the crisis of unreplicable research”. In: *PeerJ* 5 (2017), e3544 (cited on page 25).

[AR10] Howard Anton and Chris Rorres. *Elementary linear algebra: applications version*. John Wiley & Sons, 2010 (cited on page 27).

[BLK17] Olivier Bachem, Mario Lucic, and Andreas Krause. “Practical coreset constructions for machine learning”. In: *arXiv preprint arXiv:1703.06476* (2017) (cited on pages 55, 58).

[BST19] Pietro Barbiero, Giovanni Squillero, and Alberto Tonda. “Evolutionary discovery of coresets for classification”. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, July 2019 (cited on page 17).

[BST20a] Pietro Barbiero, Giovanni Squillero, and Alberto Tonda. *Modeling Generalization in Machine Learning: A Methodological and Computational Study*. 2020. arXiv: 2006.15680 [cs.LG] (cited on page 17).

[BST20b] Pietro Barbiero, Giovanni Squillero, and Alberto Tonda. “Predictable Features Elimination: An Unsupervised Approach to Feature Selection”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2020, In print (cited on pages 17, 44).

[BT19] Pietro Barbiero and Alberto Tonda. “Fundamental Flowers: Evolutionary Discovery of Coresets for Classification”. In: *Applications of Evolutionary Computation*. Springer International Publishing, 2019, pages 550–564 (cited on page 17).

[BT20] Pietro Barbiero and Alberto Tonda. “Making Sense of Economics Datasets with Evolutionary Coresets”. In: *Advances in Intelligent Systems and Computing*. Springer International Publishing, 2020, pages 162–170 (cited on page 17).

[Bar+18] Pietro Barbiero et al. “Understanding Cancer Phenomenon at Gene-Expression Level by using a Shallow Neural Network Chain”. In: *Italian Workshop on Neural Networks (WIRN 2018)*. Volume 6. 2018, pages 1–11 (cited on page 17).

[Bar+20] Pietro Barbiero et al. “A Novel Outlook on Feature Selection as a Multi-objective Problem”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2020, pages 68–81 (cited on pages 17, 41).

[BM02] Peter L Bartlett and Shahar Mendelson. “Rademacher and Gaussian complexities: Risk bounds and structural results”. In: *Journal of Machine Learning Research* 3.Nov (2002), pages 463–482 (cited on page 20).

[Bel66] Richard Bellman. “Dynamic programming”. In: *Science* 153.3731 (1966), pages 34–37 (cited on page 18).

[Bel15] Richard E Bellman. *Adaptive control processes: A guided tour*. Volume 2045. Princeton University Press, 2015 (cited on page 18).

[Ber+15] M. L. Bermingham et al. “Application of high-dimensional feature selection: evaluation for genomic prediction in man”. In: *Scientific Reports* 5.1 (May 2015) (cited on page 40).

[BDM13] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. *Near-optimal Coresets For Least-Squares Regression*. Technical report. 2013. arXiv: 1202.3505v2 (cited on pages 56, 58, 61).

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004 (cited on page 24).

[Bre99] Leo Breiman. “Pasting small votes for classification in large databases and on-line”. In: *Machine Learning* 36.1-2 (1999), pages 85–103 (cited on page 59).

[Bre01a] Leo Breiman. “Random forests”. In: *Machine Learning* 45.1 (2001), pages 5–32 (cited on page 27).

[Bre01b] Leo Breiman. “Random forests”. In: *Machine learning* 45.1 (2001), pages 5–32 (cited on page 59).

[CZH10] Deng Cai, Chiyuan Zhang, and Xiaofei He. “Unsupervised feature selection for multi-cluster data”. In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2010, pages 333–342 (cited on page 52).

[CB17] Trevor Campbell and Tamara Broderick. “Automated Scalable Bayesian Inference via Hilbert Coresets”. In: (2017). arXiv: 1710.05053 (cited on page 58).

[CB18] Trevor Campbell and Tamara Broderick. “Bayesian Coreset Construction via Greedy Iterative Geodesic Ascent”. In: *International Conference on Machine Learning (ICML)*. 2018. arXiv: arXiv:1802.01737v2 (cited on pages 56, 58, 61).

[Cas+17] Giuseppe Casalicchio et al. “OpenML: An R package to connect to the machine learning platform OpenML”. In: *Computational Statistics* 32.3 (2017), pages 1–15 (cited on page 43).

[CF67] Y Chien and King-Sun Fu. “On the generalized karhunen-loève expansion (corresp.)” In: *IEEE Transactions on Information Theory* 13.3 (1967), pages 518–520 (cited on page 40).

[Cil+19] Nicole Dalia Cilia et al. “Variable-Length Representation for EC-Based Feature Selection in High-Dimensional Data”. In: *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*. Springer. 2019, pages 325–340 (cited on page 40).

[Cla10] Kenneth L Clarkson. “Coresets, Sparse Greedy Approximation, and the Frank-Wolfe Algorithm”. In: *ACM Transactions on Algorithms*. 2010 (cited on pages 56, 58, 61).

[Cox58] David R Cox. “The regression analysis of binary sequences”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1958), pages 215–242 (cited on pages 43, 59).

[DKA06] Sophia Daskalaki, Ioannis Kopanas, and Nikolaos Avouris. “Evaluation of classifiers for an uneven class distribution problem”. In: *Applied Artificial Intelligence* 20.5 (2006), pages 381–417 (cited on page 25).

[Deb+02] Kalyanmoy Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pages 182–197 (cited on pages 41, 43, 57, 59).

[DG17] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017 (cited on page 44).

[Efr+04] Bradley Efron et al. “Least Angle Regression”. In: *The Annals of Statistics* 32.2 (2004), pages 407–451 (cited on pages 56, 58, 61).

[Eri+20] Nick Erickson et al. “AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data”. In: *arXiv preprint arXiv:2003.06505* (2020) (cited on page 54).

[FK12] Y Fan and C Kamath. *On the selection of dimension reduction techniques for scientific applications*. Technical report. Feb. 2012 (cited on page 40).

[FC91] Mark Fanty and Ronald Cole. “Spoken letter recognition”. In: *Advances in Neural Information Processing Systems*. 1991, pages 220–226 (cited on page 54).

[Fis19] Ronald A Fisher. “XV.—The correlation between relatives on the supposition of Mendelian inheritance.” In: *Earth and Environmental Science Transactions of the Royal Society of Edinburgh* 52.2 (1919), pages 399–433 (cited on pages 40, 43).

[Fis36] Ronald A Fisher. “The use of multiple measurements in taxonomic problems”. In: *Annals of eugenics* 7.2 (1936), pages 179–188 (cited on page 61).

[Fol13] Gerald B Folland. *Real analysis: modern techniques and their applications*. John Wiley & Sons, 2013 (cited on page 24).

[Fri01] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics* (2001), pages 1189–1232 (cited on page 59).

[Gar12] Aaron Garrett. *inspyred (Version 1.0.1) Inspired Intelligence*. `https://github.com/aarongarrett/inspyred`. 2012 (cited on pages 43, 59).

[Guy03] Isabelle Guyon. "Design of experiments of the NIPS 2003 variable selection benchmark". In: *NIPS 2003 workshop on feature extraction and feature selection*. 2003 (cited on pages 44, 52).

[GE03] Isabelle Guyon and André Elisseeff. "An Introduction to Variable and Feature Selection". In: *J. Mach. Learn. Res.* 3 (Mar. 2003), pages 1157–1182. ISSN: 1532-4435 (cited on page 40).

[Guy+02] Isabelle Guyon et al. "Gene selection for cancer classification using support vector machines". In: *Machine learning* 46.1-3 (2002), pages 389–422 (cited on pages 43, 52).

[Guy+05] Isabelle Guyon et al. "Result analysis of the NIPS 2003 feature selection challenge". In: *Advances in neural information processing systems*. 2005, pages 545–552 (cited on page 44).

[Hal00] Mark A Hall. "Correlation-based feature selection of discrete and numeric class machine learning". In: *Proceedings of the 17th International Conference on Machine Learning (ICML)*. 2000, pages 359–863 (cited on page 18).

[Ham+07] Tarek M Hamdani et al. "Multi-objective feature selection with NSGA II". In: *International conference on adaptive and natural computing algorithms*. Springer. 2007, pages 240–247 (cited on page 41).

[HCN06] Xiaofei He, Deng Cai, and Partha Niyogi. "Laplacian score for feature selection". In: *Advances in neural information processing systems*. 2006, pages 507–514 (cited on page 52).

[Hea+98] Marti A. Hearst et al. "Support vector machines". In: *IEEE Intelligent Systems and their Applications* 13.4 (1998), pages 18–28 (cited on page 59).

[Hei01] Gary W Heiman. *Understanding research methods and statistics: An integrated introduction for psychology*. Houghton, Mifflin and Company, 2001 (cited on page 42).

[HCB16] Jonathan H Huggins, Trevor Campbell, and Tamara Broderick. "Coresets for Scalable Bayesian Logistic Regression". In: *30th Annual Conference on Neural Information Processing Systems (NIPS)*. 2016. arXiv: `arXiv:1605.06423v3` (cited on pages 56, 58, 61).

[JP12] Jean Jacod and Philip Protter. *Probability essentials*. Springer Science & Business Media, 2012 (cited on page 24).

[Jol11] Ian Jolliffe. *Principal component analysis*. Springer, 2011 (cited on page 27).

[Koz92] John R Koza. *Genetic programming: On the programming of computers by means of natural selection*. Volume 1. MIT press, 1992 (cited on page 30).

[KL87] LF Kozachenko and Nikolai N Leonenko. "Sample estimate of the entropy of a random vector". In: *Problemy Peredachi Informatsii* 23.2 (1987), pages 9–16 (cited on pages 26, 40, 43).

[KA14] M Krzywinski and N Altman. “Points of significance: Comparing samples-part I.” In: *Nature methods* 11.3 (2014), page 215 (cited on page 43).

[KA17] Martin Krzywinski and Naomi Altman. “Classification and regression trees”. In: *Nature Methods* 14.8 (Aug. 2017), pages 757–758 (cited on page 19).

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *Nature* 521.7553 (2015), page 436 (cited on page 19).

[Lev60] Howard Levene. “Robust tests for equality of variances”. In: *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*. Edited by Ingram Olkin, Harold Hotelling, et al. Stanford University Press, 1960, pages 278–292 (cited on page 25).

[LKA16a] Jake Lever, Martin Krzywinski, and Naomi Altman. “Logistic regression”. In: *Nature Methods* 13.7 (June 2016), pages 541–542 (cited on page 19).

[LKA16b] Jake Lever, Martin Krzywinski, and Naomi Altman. “Model selection and overfitting”. In: *Nature Methods* 13.9 (Aug. 2016), pages 703–704 (cited on pages 19, 20).

[Lew62] P Lewis. “The characteristic selection problem in recognition systems”. In: *IRE Transactions on information theory* 8.2 (1962), pages 171–178 (cited on page 40).

[Li+18] Jundong Li et al. “Feature selection: A data perspective”. In: *ACM Computing Surveys (CSUR)* 50.6 (2018), page 94 (cited on page 52).

[Li+12] Zechao Li et al. “Unsupervised feature selection using nonnegative spectral analysis”. In: *Twenty-Sixth AAAI Conference on Artificial Intelligence*. 2012 (cited on page 52).

[LW+02] Andy Liaw, Matthew Wiener, et al. “Classification and regression by Random Forest”. In: *R news* 2.3 (2002), pages 18–22 (cited on page 27).

[LTR17] Henry W Lin, Max Tegmark, and David Rolnick. “Why does deep and cheap learning work so well?” In: *Journal of Statistical Physics* 168.6 (2017), pages 1223–1247 (cited on page 39).

[Lóp+13] Victoria López et al. “An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics”. In: *Information Sciences* 250 (2013), pages 113–141 (cited on page 27).

[M A60] Efroymson M. A. “Multiple Regression Analysis”. In: *Mathematical Methods for Digital Computers* (1960) (cited on pages 56, 58, 61).

[Mac95] George Mackiw. “A note on the equality of the column, and row rank of a matrix”. In: *Mathematics Magazine* 68.4 (1995), page 285 (cited on page 24).

[MZ93] Stéphane Mallat and Zhifeng Zhang. “Matching pursuits with time-frequency dictionaries”. In: *IEEE Transactions on Signal Processing* 42.12 (1993), pages 3397–3415 (cited on pages 56, 58).

[MZ75] Richard D McKelvey and William Zavoina. “A statistical model for the analysis of ordinal level dependent variables”. In: *Journal of Mathematical Sociology* 4.1 (1975), pages 103–120 (cited on page 19).

[Mic+94] Donald Michie et al., editors. *Machine Learning, Neural and Statistical Classification*. Upper Saddle River, NJ, USA: Ellis Horwood, 1994. ISBN: 0-13-106360-X (cited on page 25).

[Mir12] Leonid Mirsky. *An introduction to linear algebra*. Courier Corporation, 2012 (cited on page 24).

[Ney+17] Behnam Neyshabur et al. "Exploring Generalization in Deep Learning". In: *arXiv:1706.08947 [cs]* (July 2017). arXiv: 1706.08947 [cs] (cited on page 20).

[OOK17] Dijana Oreski, Stjepan Oreski, and Bozidar Klicek. "Effects of dataset characteristics on the performance of feature selection techniques". In: *Applied Soft Computing* 52 (Mar. 2017), pages 109–119. ISSN: 15684946 (cited on pages 18, 25, 27).

[Osi+17] FY Osisanwo et al. "Supervised machine learning algorithms: classification and comparison". In: *International Journal of Computer Trends and Technology (IJCTT)* 48.3 (2017), pages 128–138 (cited on page 18).

[PLH95] Panos M Pardalos, Y Li, and WW Hager. "Linear programming approaches to the convex hull problem in $\mathbb{R}^m$". In: *Computers & Mathematics with Applications* 29.7 (1995), pages 23–29 (cited on page 23).

[PRK93] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad. "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition". In: *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers* (1993), pages 40–44. ISSN: 1058-6393. arXiv: 1108.3326 (cited on pages 56, 58, 61).

[Pea05a] Karl Pearson. "Das fehlergesetz und seine verallgemeiner-ungen durch Fechner und Pearson. A Rejoinder". In: *Biometrika* 4.1-2 (1905), pages 169–212 (cited on page 25).

[Pea05b] Karl Pearson. "Skew variation, a rejoinder". In: *Biometrika* 4.1-2 (1905), pages 169–212 (cited on page 25).

[Pea20] Karl Pearson. "Notes on the history of correlation". In: *Biometrika* 13.1 (1920), pages 25–45 (cited on page 25).

[Ped+11a] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pages 2825–2830 (cited on page 59).

[Ped+11b] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *Journal of Machine Learning Research* 12.Oct (2011), pages 2825–2830 (cited on pages 43, 52).

[Pla+99] John Platt et al. "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods". In: *Advances in Large Margin Classifiers* 10.3 (1999), pages 61–74 (cited on page 27).

[Pog+04] Tomaso Poggio et al. "General conditions for predictivity in learning theory". In: *Nature* 428.6981 (2004), page 419 (cited on page 20).

[Qui87] J. Ross Quinlan. "Simplifying decision trees". In: *International journal of man-machine studies* 27.3 (1987), pages 221–234 (cited on page 44).

[Raj+20] Abhejit Rajagopal et al. "Predicting Generalization in Deep Learning via Local Measures of Distortion". In: *arXiv:2012.06969 [cs, stat]* (Dec. 2020). arXiv: 2012.06969 [cs, stat] (cited on page 20).

[Ros14] Brian C Ross. "Mutual information between discrete and continuous data sets". In: *PloS ONE* 9.2 (2014), e87357 (cited on page 26).

[Rou87] Peter J Rousseeuw. “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis”. In: *Journal of computational and applied mathematics* 20 (1987), pages 53–65 (cited on page 54).

[RR16] Alessandro Rudi and Lorenzo Rosasco. “Generalization Properties of Learning with Random Features”. In: *arXiv:1602.04474 [cs, stat]* (Feb. 2016). arXiv: `1602.04474 [cs, stat]` (cited on page 20).

[SL09] Michael Schmidt and Hod Lipson. “Distilling free-form natural laws from experimental data”. In: *Science* 324.5923 (2009), pages 81–85 (cited on page 30).

[Sch+78] Gideon Schwarz et al. “Estimating the dimension of a model”. In: *The Annals of Statistics* 6.2 (1978), pages 461–464 (cited on page 21).

[Sie87] J Paul Siebert. “Vehicle recognition using rule based methods”. In: (1987) (cited on page 44).

[SK05] Guido F Smits and Mark Kotanchek. “Pareto-front exploitation in symbolic regression”. In: *Genetic Programming Theory and Practice II*. Springer, 2005, pages 283–299 (cited on page 21).

[ST04] Daniel A Spielman and Shang-Hua Teng. “Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time”. In: *Journal of the ACM (JACM)* 51.3 (2004), pages 385–463 (cited on page 23).

[ST+60] Robert George Douglas Steel, James Hiram Torrie, et al. “Principles and procedures of statistics.” In: *Principles and procedures of statistics.* (1960) (cited on page 54).

[Ste56] Hugo Steinhaus. “Sur la division des corp materiels en parties”. In: *Bull. Acad. Polon. Sci* 1.804 (1956), page 801 (cited on page 54).

[Sto74] Mervyn Stone. “Cross-validatory choice and assessment of statistical predictions”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 36.2 (1974), pages 111–133 (cited on pages 22, 25).

[Tik43] Andrey Nikolayevich Tikhonov. “On the stability of inverse problems”. In: *Dokl. Akad. Nauk SSSR*. Volume 39. 5. 1943, pages 195–198 (cited on page 59).

[Tiw08] Hans Raj Tiwary. “On the hardness of computing intersection, union and Minkowski sum of polytopes”. In: *Discrete & Computational Geometry* 40.3 (2008), pages 469–479 (cited on page 23).

[Tsa12] Flora S. Tsai. “Dimensionality reduction for computer facial animation”. In: *Expert Systems with Applications* 39.5 (Apr. 2012), pages 4965–4971 (cited on page 40).

[TKC05] Ivor W Tsang, James T Kwok, and Pak-Ming Cheung. “Core vector machines: Fast SVM training on very large data sets”. In: *Journal of Machine Learning Research* 6.Apr (2005), pages 363–392 (cited on page 58).

[Tur+11] Michelle C Turner et al. “Long-term ambient fine particulate matter air pollution and lung cancer in a large cohort of never-smokers”. In: *American journal of respiratory and critical care medicine* 184.12 (2011), pages 1374–1381 (cited on page 54).

[Van79] Cornelis Joost Van Rijsbergen. *Information retrieval*. Butterworth-Heinemann, 1979 (cited on pages 27, 53).

[Van+13] Joaquin Vanschoren et al. “OpenML: Networked Science in Machine Learning”. In: *SIGKDD Explorations* 15.2 (2013), pages 49–60 (cited on pages 25, 43, 54).

[VC15] Vladimir N Vapnik and A Ya Chervonenkis. “On the uniform convergence of relative frequencies of events to their probabilities”. In: *Measures of Complexity*. Springer, 2015, pages 11–30 (cited on page 20).

[Ver+12] Alexander Vergara et al. “Chemical gas sensor drift compensation using classifier ensembles”. In: *Sensors and Actuators B: Chemical* 166 (2012), pages 320–329 (cited on page 54).

[VMS13] Leandro D Vignolo, Diego H Milone, and Jacob Scharcanski. “Feature selection for face recognition based on multi-objective evolutionary wrappers”. In: *Expert Systems with Applications* 40.13 (2013), pages 5077–5084 (cited on page 41).

[War63] Joe H Ward Jr. “Hierarchical grouping to optimize an objective function”. In: *Journal of the American statistical association* 58.301 (1963), pages 236–244 (cited on page 54).

[Wel47] Bernard L Welch. “The generalization of Student’s’ problem when several different population variances are involved”. In: *Biometrika* 34.1/2 (1947), pages 28–35 (cited on page 43).

[Wes+00] J. Weston et al. “Feature selection for SVMs”. In: *Advances in Neural Information Processing Systems 13*. MIT Press, 2000, pages 668–674 (cited on page 40).

[XFZ14] Bing Xue, Wenlong Fu, and Mengjie Zhang. “Multi-objective feature selection in classification: a differential evolution approach”. In: *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer. 2014, pages 516–528 (cited on page 41).

[Xue+15] Bing Xue et al. “A survey on evolutionary computation approaches to feature selection”. In: *IEEE Transactions on Evolutionary Computation* 20.4 (2015), pages 606–626 (cited on page 40).

[Yan+11] Yi Yang et al. “L2, 1-norm regularized discriminative feature selection for unsupervised”. In: *Twenty-Second International Joint Conference on Artificial Intelligence*. 2011 (cited on page 52).

[YHL11] Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. “Dual coordinate descent methods for logistic regression and maximum entropy models”. In: *Machine Learning* 85.1-2 (2011), pages 41–75 (cited on page 27).

[YL03] Lei Yu and Huan Liu. “Feature selection for high-dimensional data: A fast correlation-based filter solution”. In: *Proceedings of the 20th international conference on machine learning (ICML)*. 2003, pages 856–863 (cited on page 18).

[Zha+16] Chiyuan Zhang et al. “Understanding deep learning requires rethinking generalization”. In: *arXiv preprint arXiv:1611.03530* (2016) (cited on page 21).

[ZL07] Zheng Zhao and Huan Liu. “Spectral feature selection for supervised and unsupervised learning”. In: *Proceedings of the 24th international conference on Machine learning*. 2007, pages 1151–1157 (cited on page 52).

[Zho+19] Zhiguo Zhou et al. “Multi-objective based radiomic feature selection for lesion malignancy classification”. In: *IEEE journal of biomedical and health informatics* (2019) (cited on page 41).

[ZWC11] Dennis Zill, Warren S Wright, and Michael R Cullen. *Advanced engineering mathematics*. Jones & Bartlett Learning, 2011 (cited on page 52).

[ZSK12] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. "A survey on unsupervised outlier detection in high-dimensional numerical data". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 5.5 (2012), pages 363–387 (cited on page 19).

3. Evolutionary Optimization

Since the start of my Ph.D., evolutionary optimization has been a core part of my scientific activities. While most research lines I worked on dealt with real-world case studies, I also contributed to more fundamental aspects of Evolutionary Computation (EC), often motivated by practical obstacles found by tackling applications.

This chapter presents the outcomes of two research lines I explored: the first, related to diversity preservation and promotion, summarizes a review on the state of the art [ST16] (Section 3.1) and two proposed distance metrics for complex genomes [GST13a; GST14] (Section 3.2). This part has been developed through my shared co-supervision of Ph.D. student Marco Gaudesi, from Politecnico di Torino. The second research line described in this chapter (Section 3.3) deals with the use of standard Genetic Programming (GP) for learning models encoded as systems of Ordinary Differential Equations (ODEs) [Gau+14]. While this last part is more application-oriented, I still consider it part of my contributions to the field of evolutionary algorithms (EAs), and I decided to place it as the last section of this chapter as a bridge towards the real-world applications of Chapter 4 and 5.

3.1 Diversity Preservation and Promotion

In the past decades, different evolutionary optimization methodologies have been proposed by scholars and exploited by practitioners, in a wide range of applications. Each paradigm shows distinctive features, typical advantages, and characteristic disadvantages; however, one single problem is shared by almost all of them: the "lack of speciation". While natural selection favors variations toward greater divergence, in artificial evolution candidate solutions do homologize. Many authors argued that promoting diversity would be beneficial in evolutionary optimization processes, and that it could help avoiding premature convergence on sub-optimal solutions. This section surveys the research in this area up to mid 2010s, it re-orders and re-interprets different methodologies into a single framework, and proposes a novel three-axis taxonomy. Its goal is to provide the reader with a unifying view of the many contributions in this important corpus, allowing comparisons and informed choices. Characteristics of the different techniques are discussed, and similarities are highlighted; practical ways to measure and promote diversity are also suggested.

3.1.1 Introduction

Evolution is the biological theory that animals and plants have their origin in other types, and that the distinguishable differences are due to modifications in successive generations. Natural evolution is not a random process, on the contrary, it is based on random variations, but some are rejected while others preserved according to objective evaluations. Only changes that are beneficial to the individuals are likely to spread into subsequent generations. Darwin called this principle "natural selection" in his *Origin of the Species* [Dar59], a process where random variations simply "afford materials".

When natural selection causes variations to be accumulated in one specific direction the result strikingly resembles a deliberate optimization process. Indeed, such processes only require to assess the effect of random changes, not the ability to design intelligent modifications, inspiring several researchers. *Evolutionary computation* (EC) is the offshoot of computer science focusing on algorithms loosely inspired by the theory of evolution. The definition is deliberately vague since the boundaries of the field are not, and cannot be, sharply defined. EC is a branch of computational intelligence, and it is also included into the broad framework of bio-inspired heuristics.

Divergence of character is a cornerstone of Darwinian theory: "the principle, which I have designated by this term, is of high importance on my theory, and explains, as I believe, several important facts" [Dar59]. The English biologist favored sympatric speciation, and such principle perfectly rationalizes why breeds diverge in character not only from their common parents, but also from each other; and why differences, at first barely appreciable, steadily increase over generations. Indeed, the principle is "simplicity itself" [May92]: the more the co-habitants of an area differ from each other in their ecological requirements, the less they will compete; therefore, in natural evolution, any variation toward greater divergence is likely to be favored.

Differently, artificial evolution in EC is plagued by an endemic lack of diversity: during evolutionary optimization processes, all candidate solutions frequently homologize. This situation has different effects on the different search algorithms, but almost all are quite deleterious. Such a *lack of speciation* has been pointed out by Holland in his seminal works [Hol75], and nowadays is plainly recognized by scholars. The problem is usually labeled with the oxymoron "premature convergence", that is, the tendency of an algorithm to converge towards a point where it was not supposed to converge to in the first place.

EC is based by necessity on oversimplifications of the complex mechanics of nature. Darwinian theory focuses on members of the same species vying for limited resources, and the push for evolutionary diversification is prominent in ecosystems with limited resources [SB07], while it is not clear whether the competition between different species favors or rather impedes diversification [Bai+13]. In artificial evolution, on the other hand, there is no clear distinction between *intraspecific* and *interspecific* competition, because either the struggle is simulated at the level of individuals inside a single species, or at the level of species with no explicitly defined individuals.

Moreover, according to Darwin "the same spot will support more life if occupied by very diverse forms" [Dar59]. But optimization algorithms use a *fitness function* that evaluates the goodness of each candidate solution with respect to a given task – that is, the whole ecosystem is indirectly modeled through its effects, and only few very specific facets are taken into consideration. The general inability to exploit environmental niches noted by Holland could be explained with the absence of such *natural spots* to survive in. Indeed, even the term "environment" is not widely used by Evolutionary Computation (EC) scholars, that favor "fitness landscape".

Over the years, EC has shown the capability to tackle quite difficult problems with very complex

fitness landscapes. Evolutionary optimizers have been successfully exploited both in stationary and dynamic situations, and they were demonstrated able to identify either single optimums or whole Pareto sets in multi-objective problems. In such a wide variety of applications, promoting diversity inside the population has often been seen as the common key factor to improve performances (e.g., [BR07; BGK04; CLY09; Shi01; Urs02]).

Not surprisingly, scientific literature reports several sharp methodologies to promote diversity, that range from general techniques to problem-dependent heuristics. However, the fragmentation of the field and the difference in terminology led to a general dispersion of this important corpus of knowledge in many small, hard-to-track research lines – and consequently to the risk of neglecting effective solutions already known in similar domain, or re-discovering equivalent solutions in different communities.

The goal of this survey is to re-order and re-interpret the different approaches for promoting diversity into a single comprehensive framework, and to propose a taxonomy that enables the comparison of techniques originally presented for different EAs. For the nature of the topic and the vastness of the field, such a classification will be necessarily coarse-grained. Nevertheless, it could help scholars undertaking these issues, and practitioners tackling new problems.

It is important to stress that these considerations concern EAs applied to optimization tasks, only. There is a considerable amount of research lines that employ evolutionary computation as a means to analyze the dynamics of artificial life, with different problems and objectives, for which the premises of this work may not hold true[1].

In the following, subsection 3.1.2 briefly introduces EC and the notion of diversity, and subsection 3.1.3 introduces the proposed taxonomic scheme. The following subsections use the first axis of the taxonomy as main theme to survey and classify the different approaches presented in literature, namely section 3.1.4 for lineage-based ones, 3.1.5 for genotype-based, and 3.1.6 for phenotype-based. Subsection 3.1.7 provides some recommendations for practitioners who approach this topic. Finally, subsection 3.1.8 concludes the section and shows possible future research directions for the subject.

3.1.2 Evolutionary computation and the notion of diversity in evolutionary algorithms

EC does not have a single recognizable origin. Some scholars identify its starting point in 1950, when Alan Turing drew attention to the similarities between learning and evolution [Tur50]. Others pointed out the inspiring ideas that appeared later in the decade, despite the fact that the lack of computational power impaired their diffusion in the broader scientific community [Fog98]. More commonly, the birth of EC is set in the 1960s with the appearance of three independent research lines: John Holland's genetic algorithms (GA) [Hol75]; Lawrence Fogel's evolutionary programming (EP) [Fog62]; Ingo Rechenberg's and Hans-Paul Schwefel's evolution strategies (ES) [BS02b]. The three paradigms monopolized the field until the 1990s, when John Koza entered the arena with genetic programming (GP) [Koz92].

These four main paradigms, together with several variants proposed over the years, have been grouped under the umbrella term of "evolutionary algorithms" (EAs). In the 1990s, Kennedy et al. proposed the new "particle swarm optimization" (PSO), a population-based optimization technique that mimics the principles of social interaction rather than struggling for survival. Despite the apparent differences, PSO and all related *swarm-based* approaches are nowadays usually listed amongst EA techniques [PKB07].

[1] Readers interested in *Artificial Life* as a discipline may find a comprehensive review of its roots, methodological tools, and applications in [BM12].

In such algorithms, a single candidate solution is an *individual*; the set of all candidate solutions that exists at a particular time represents the *population*. Evolution proceeds through discrete steps called *generations*. In each of them, the population is first expanded and then collapsed, mimicking the processes of breeding and struggling for survival. Moreover, most evolutionary optimization algorithms proceed by alternating phases of *exploration*, where distant parts of the search space are sampled, with phases of *exploitation*, where small neighborhoods of the best solutions are thoroughly investigated for improvements. Maintaining a set of solutions, EAs are more resilient than other optimization techniques to the attraction of local optima [ES10]; nevertheless, particularly promising points might induce an algorithm to a premature phase of exploitation, concentrating the whole population around few single points. When all candidate solutions are very close in the solution space, EAs could eventually lose their ability to explore new promising areas.

This phenomenon is a direct consequence of the mechanisms used to generate new individuals, collectively known as *genetic operators*. They can be roughly divided into *recombinations* and *mutations*: the formers mix together the information contained in two or more solutions to create new ones; the latter work by changing the structure of a single solution, and usually perform small adjustments. When all parents are very similar, the potential to *jump* to remote parts of the search space is strongly impaired: thus, "conventional wisdom suggests that increasing diversity should be generally beneficial" [Shi97]. It is, therefore, useful to first discuss the notion of "diversity", the reciprocal concept of "similarity", and how they can be measured.

Diversity is usually quantified in three different ways: as a distance metric between individuals; as a measurable attribute of the individuals (*individual diversity*); as a characteristic of the population as a whole (*population diversity*). Individual diversity may be measured considering *how far* an individual is from the whole population, or from a subset; population diversity, similarly, may be defined as the average individual diversity. However, in many applications, a distance is not required to sensibly define the concept of "individual diversity", nor individual diversity is required to define an acceptable "population diversity".

When analyzing an evolutionary process, at least three different levels can be recognized: *genotype*, *phenotype*, and *fitness* – and promoting diversity might be pursued in different manners at each level. Considering diversity at the level of phenotype is usually the more effective way to enhance performances, but phenotypic diversity is also the hardest to define and the more demanding to calculate. In practical cases, assessing the diversity at the level of genotype requires acceptable effort, and there is more correlation between genotype and phenotype than between phenotype and fitness. This considerations may explain why the majority of the methodologies for promoting diversity take into account the genotypic diversity.

Genotype-phenotype distinction

In biology, the distinction between genotype and phenotype is apparent: the former is the genetic constitution of an organism; the latter is the composite of the organism's observable characteristics or traits. Moreover, albeit a precise definition of the term "fitness" led to some discussions [Daw99], all scholars agree on the relationship between individual's fitness and reproductive success[2].

In EC, the genotype is the internal representation of the individual, or, more operatively, the entity that is directly manipulated by genetic operators; the fitness is the measure of how well the candidate solution is able to solve the target problem. While alternative definitions have been proposed in literature, for the purpose of this discussion the term "phenotype" denotes the candidate solution

[2]The phrase "survival of the fittest" was first used by Herbert Spencer in his *Principles of Biology* [Spe64] to better describe Darwin's idea of "preservation of favoured races in the struggle for life."

that is encoded in the genotype. That is, whenever the genotype cannot be evaluated directly by the fitness function, but needs to be transformed into something else, the phenotype is the *intermediate form* in which the genotype needs to be transformed into[3]. When the genotype can be directly evaluated, genotype and phenotype coincide. The genotype-phenotype mapping is, by definition, strictly deterministic, as there is no environment that could interfere with the process. Whether random elements need to be considered, scholars classify the fitness as *noisy*, assuming that the same phenotype could be evaluated differently. Such cases are not considered here.

In evolutionary paradigms such as Genetic Programming (GP) [Koz92] and Linear Genetic Programming (LGP) [BB07] the distinction between genotype, phenotype and fitness is usually clearly visible. For instance, when GP is applied to symbolic regression, the genotype is the tree, the phenotype is the function, and the fitness relates to the difference between experimental data and those generated by the expression itself (Figure 3.1).

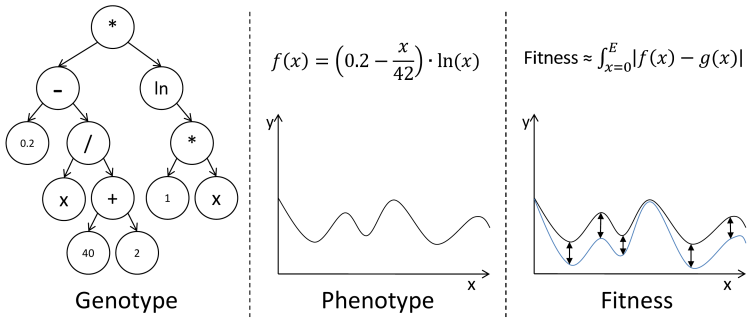


Figure 3.1: Genotype, phenotype and fitness for GP applied to a symbolic regression problem.

It may be worth noting that the phenotype is the function in its mathematical sense[4], or, equivalently, an expression in a defined canonical form. As a consequence, $x+y+2$, $y+2+x$, and $x+3+xy+y-1-yx$ express the same phenotype, although they are likely to derive from different genotypes. Similar considerations hold true for any other EA adopting complex individual encoding.

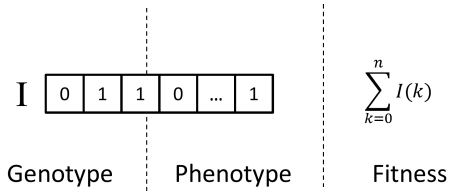


Figure 3.2: Genotype, phenotype and fitness for GA applied to the one-max test function.

On the contrary, in the classic GA, modern EP, ES, PSO, and Differential Evolution (DE) [SP97] such a difference can be less evident. Individuals are usually represented by arrays of numbers – either binary or real values – and for opposite reasons the phenotype has been sometimes identified with the array itself, and sometimes with the fitness value. As noted above, here the former option is adopted (Figure 3.2).

[3]The distinction between "structure" and "behavior" found in [BGK04] is also compatible with the above definition.

[4]A function f from S into T is a subset of the product set $S \times T$ with the property that for each element $x \in S$, there exists a unique element $y \in T$ such that $(x,y) \in f$.

Measuring diversity at different levels

As a matter of fact, detecting whether two individuals are *clones*, i.e., identical, is often an easy task at any level. As seen in section 3.1.2, both the transformation of a phenotype into a genotype, and the evaluation of a phenotype are assumed to be fully deterministic processes. Thus, for two individuals x and y, different fitness values imply different phenotypes, and different phenotypes imply different genotypes:

$$\mathscr{F}_x \neq \mathscr{F}_y \Rightarrow \mathscr{P}_x \neq \mathscr{P}_y \Rightarrow \mathscr{G}_x \neq \mathscr{G}_y \tag{3.1}$$

where $\mathscr{F}$ denotes the fitness, $\mathscr{P}$ the phenotype, and $\mathscr{G}$ the genotype. However, different genotypes are not necessarily translated to different phenotypes, as different behaviors are not necessarily rewarded with different fitness values. Nor should the amount of diversity at one level suggest a similar amount at a different one.

As fitness values are usually real numbers, or vectors in $\mathbb{R}^n$, it may be quite easy to define distance metrics at this level. Such measures have *per se* little significance, however, unless they can be used as a proxy for different distances. When "phenotypic diversity is synonymous with fitness diversity" [Cor+12], the lack of difference between genotypes can be sensibly inferred from the fitness values being equal. Similarly, variety in fitness values could be used as a proxy for measuring population diversity at the level of phenotype. Intuitively, phenotype variations could cause more fitness variations when the fitness is a vector in a high-dimensional space $v_f \in \mathbb{R}^n$, as it happens in multi-objective optimization, because obtaining identical fitness values is less probable.

The *locality principle* states that small modifications of the genotype should correspond to small modifications of the phenotype, and small modification of the latter should yield small variations in the fitness (that is, in the typical terminology of ES scholars, the problem exhibits a *strong causality*). In this situation the fitness distance can be used as an effective measure for both genotypic and phenotypic distance. And, in practical terms, such smoothness may be regarded as a lucky foreteller of an easy problem. Indeed, *locality* is not an intrinsic characteristic of the problem, but of the genotype-phenotype-fitness mapping; and a doltish definition of the fitness function could cause different phenotypes to yield the very same value.

Measuring diversity in a population of individuals is a simple task if a distance metric is available. In a classic GA, for example, the genotype distance can be straightforwardly evaluated resorting to the Hamming distance [Shi97]. In other cases, a considerable number of distance metrics can be defined over real-value genotypes, or over real-value fitnesses, a comprehensive analysis of which can be found in [Cor+12]. For EAs featuring more complex individual structures, such as GP, the literature presents several solutions. In his seminal work, Koza proposes to take into account the number of different genotypes contained in a population [Koz92], and the idea is later considered as a sufficient upper bound of population diversity [Lan98]. Genotypical diversity in GP can also be measured using subtrees and their relative frequency in the population [Kei96; Tac94]. In [MH99], numerical tags are assigned to each node in the population, in order to track their survival and the changes of context from the initial stages, confirming that populations in the final steps of the EA often descend from one single individual. A thorough analysis of diversity measurements in GP and their performances can be found in [BGK04].

The *edit distance*, also known as "Levenshtein distance", is a well-known string metric used to compute the difference between two sequences [Lev66]. It has been applied to GP genotypes considering single nodes insertion, deletions and substitutions as possible operations [DWP01;

ORe97]; a modified version proposed in [EN00] also takes into account the cost of replacing a node with one of a different type; and in [BB02], a similar idea is applied to LGP, considering the coding part of the genome along with a sequence of operators. Interestingly, in many applications the edit distance could be also used to appraise diversity between phenotypes, although, given its computational complexity, it may not be very efficient [GST13b].

It is also possible to define a measure of diversity for a set of individual without relying on distance metrics. Shannon ideas [Sha48] have been applied to fitness values in the population to define *entropy* and *free energy*: absence of changes or monotonic decreases in the grade of disorder in subsequent generations correlate with a fall into local optima [Ros95]. The Shannon entropy is also used in [ST08]: nodes and pattern of nodes in the genotypes are regarded as symbols in messages. Then, individuals that increase the amount of information carried by the whole population are slightly favored regardless of their fitness value. The idea of bits of information is also applied in [GST13b], where the distance between LGP individuals is computed as the symmetric difference between their sets of symbols.

It is always important to consider the possibility of creating *ad-hoc* distance measures, exploiting specific characteristics of the target problem. For example, [BB02] proposes alternative distance metrics for a program evolved through LGP, based on the number and type of test cases satisfied.

Interestingly, methodologies in EC often consider the average distance between solutions as a proxy for the global amount of diversity, and resort to a comparison of all possible pairs in a population, with algorithms of complexity $\mathcal{O}(n^2)$, where n is the size of the population: in [WO03], the authors propose a new algorithm that is proved to obtain the same result in time $\mathcal{O}(n)$.

3.1.3 Proposed taxonomy

It may be maintained that a methodology for promoting diversity alters the selection probability of individuals:

$$\overline{p}_{x|\Psi} = p_{x|\Psi} \cdot \xi(x, \Psi) \tag{3.2}$$

where $\overline{p}_{x|\Psi}$ is the selection probability of individual x given that individuals in set Ψ are also chosen (the set Ψ may be empty); $p_{x|\Psi}$ is the same probability without the adopted methodology; and ξ the corrective factor. Such a definition does not imply that a mechanism operates explicitly on the selection operators, but rather the effects on selection probabilities are assessed to classify it. For instance, in the well-known island model $\xi(a, \{b\}) = 0$ when the two individuals a and b are on different islands.

A simple taxonomic scheme is here proposed, based on three independent categories (Table 3.1). No distinction is made whether the goal is to *preserve* the existing diversity in the population, or rather to *promote* it. The main classification rests on the individual's relevant characteristics, that is, which element influences most the value of ξ; this text considers *lineage* ($\mathscr{L}$), *phenotype* ($\mathscr{P}$), and *genotype* ($\mathscr{G}$). Then, the different methodologies can be categorized considering the *type of selection* influenced by the methodology: *parent selection*, *survival selection*, or both. Finally, it may be considered whether the probability of choosing an individual is influenced by the choice of another one (*context-dependent* methodologies) or not (*context-independent* methodologies).

The first axis of the taxonomy is the more apparent. A mechanism based on lineage considers the topology of the population, the individual's progenitors, or, more generally, the *conditions* of its birth, for calculating ξ. Mechanisms based on the genotype evaluate the individuals' internal structure, while mechanisms based on the phenotype take into account how individuals behave.

Table 3.1: Proposed taxonomic scheme.

Characteristic	Possible values
Element considered	Lineage
	Genotype
	Phenotype
Type of selection	Parent
	Survival
	Both
Context dependency	True
	False

The fitness is not considered here, because of the assumption that it is always used as a proxy to measure distance at a different level. Some methodologies consider the fitness of individuals to determine which one should be removed during, or prior to, diversity promotion, and here they are categorized as phenotype-based.

The second categorization is based on the type of selection modified by the diversity preservation mechanism: *parent selection* or *survival selection*. During the former, parents are chosen to generate offspring; while the latter individuals are chosen to survive up to the next generation.

The last distinction is between *context dependent* preservation mechanisms and *context independent* ones. The former methods base the value of the corrective factor ξ on the value of Ψ; the latter methods assume:

$$\forall \Psi : \xi(x, \Psi) = \xi(x, \emptyset) = \xi(x) \tag{3.3}$$

In other words, a methodology is classified as context dependent if the probability of selection for an individual x (reproduction or survival) is modified in function of the presence of other (possibly specific) individuals in the set Ψ of individuals already selected. On the opposite, a methodology is considered context independent if the probability of selection for an individual x is not influenced by the composition of set Ψ. If a mechanism operates on parent selection and the algorithm does not implement sexual recombination, it is considered context-independent.

The latter distinction is relevant because it may help better understand how the mechanisms operate. Intuitively, context-dependent methodologies support the exploitation of diversity already inside the population. However, it may be generally less straightforward to add such techniques to an existing algorithm, and they may strongly bias the whole evolutionary process. According to this point of view, *migrants* can be seen as the necessary patch to un-bias the evolution after a division into *islands* enforces a strictly context-dependent scheme.

All the techniques presented in the following, well-known approaches found in literature, are organized using the proposed taxonomy. Table 3.2 on page 101 shows the complete summary of the classification performed: the first column reports the part of the section where the technique is discussed in more detail; the *main element* column, $\mathscr{L}$ indicates lineage, $\mathscr{G}$ genotype, and $\mathscr{P}$ phenotype; in the *selection* columns, "P" stands for "parent" and "S" for "survival"; the last column reports whether the considered technique depends on the context.

3.1.4 Lineage-based methodologies

In *lineage-based* methodologies the value of ξ does not depend on an individual structure or behavior, but it can be determined considering the *circumstances* of its birth (e.g., time, location). Such techniques can thus be applied to any kind of problem, even in addition to other diversity preservation methods.

Several techniques belonging to this category are based on limiting interactions between individuals, or creating constraints on selection for reproduction or survival. Island models and cellular EAs, which share the basic idea of fractionating the panmictic population of a classical EA into several sub-populations, are sometimes called *decentralized* EAs and are the most popular lineage-based methods in practice.

Island models

A popular lineage-based diversity method consists of splitting the main population into sub-populations, and greatly limiting the exchange of individuals between the sub-populations. EAs resorting to this technique are often called *island models* [WRH99], or *distributed EAs*. The intuition behind this technique is that, since EAs are stochastic in nature, different populations may explore different parts of the search space. Since a larger population is also proven to be beneficial, many island-model EAs employ a periodic exchange of the best individuals between sub-population that is usually termed *migration*, allowing the recombination operators to mix useful traits emerged separately.

Island models are easy to implement and well-suited for parallel or distributed computing, so they have been and are successfully used in real-world problems [BB01; MG15; OK09]. In addition, the computational overhead of managing several islands is almost negligible, and concentrated in the migration function, usually performed once every *epoch* (few generations). An example of the benefits provided by island models is summarized in Figure 3.3. Belonging to an island influences both the probability of reproduction and survival of an individual. The technique is context-dependent, because individuals in a sub-population can be selected only with others on the same one.

Segregation

Besides island-model and cellular EAs, other diversity preservation approaches are also based on the notion of limiting interaction between individuals. *Segregation* [Aff01] is a lineage-based diversity technique, where the global population is initially split into N sub-populations that evolve independently. Once all sub-population reach *stagnation* (e.g., the best individual has not improved for a user-defined number of generations, or a user-defined limit of iterations is reached), they are merged into $N-1$ populations, and the evolution is resumed. The process is iterated until a single panmictic population reaches stagnation, see Figure 3.4 for an example. Like for islands, the rationale of this method is that different sub-populations will explore different areas of the search space: when each segregated sub-population converges on a local optimum, they are merged with the hope of finding better optima by combining solutions in different parts of the search space. Segregation influences both reproduction and survival, and it is context-dependent.

Cellular EAs

Cellular EAs [AD08; Rob87], also known as *lattice models*, limit the interaction between individuals by creating a structure similar to that of cellular automata [Tom05]: candidate solutions evolve in overlapped neighborhoods, often over a bi-dimensional grid of individuals, such as the one presented in Figure 3.5. When an individual is selected for reproduction, it can only combine with others in

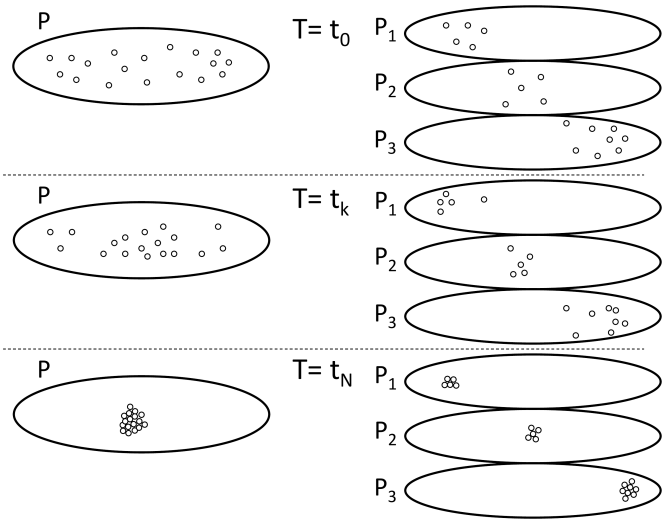


Figure 3.3: Distribution of solutions over generations, in a single-population EA (**left**) against a island-model EA with three sub-populations (**right**). Even if the global number of individuals is the same, the isolated sub-populations converge on different optima in the search space. Eventually, migration of individuals between sub-populations can help the algorithm to escape local points of attraction and converge on global optima.

the same neighborhood. Newly created solutions replace individuals in the same neighborhood of their parents, usually following a fitness-based criteria. Like cellular automata, cellular EAs can be either *asynchronous* or *synchronous* [SR98], depending on the strategy of individual replacement and population update.

Since cellular EAs promote the smooth diffusion of good solutions, creating temporary niches, they preserve diversity better than classical EAs with a panmictic population. In fact, cellular EAs present a large number of tightly connected, small sub-populations, while island models maintain a small number of loosely connected, large sub-populations. Still, this approach is generally more complex to implement than islands. This technique has an influence on both parent and survival selection, and it is classified as context-dependent because individuals can be chosen only among the neighborhoods.

Aging

In *generational approaches*, the parental population is completely replaced by the offspring before applying survival selection: the main advantage of this approach lies in an added resilience against local optima. In its basic form, *Aging* can be seen as a softened version of a generational approach: individuals do age, and are discarded after A_{max} generations [CNR07]. Indeed, a full generational EA can be seen as an EA with aging where the maximum age is set to $A_{max} = 1$.

This technique has an influence on both parent and survival selection, and it is classified as context-independent. However, the aging process could be influenced by the performance of the individual or by its fitness [SSS11a]. In such cases, the methodology should be considered as based on phenotype.

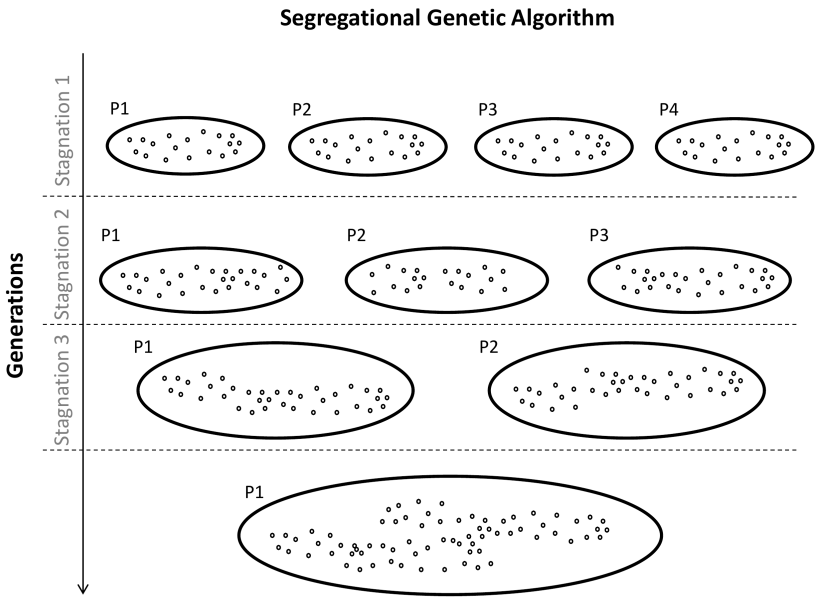


Figure 3.4: Segregational genetic algorithm, starting with 4 subpopulations. Each time a stagnation condition is reached, all individuals are mixed and merged into $N-1$ subpopulations, and the process is iterated until a single panmictic population is obtained.

Deterministic crowding

The basic idea of *Deterministic crowding* [Mah95] is to generate a competition between the children individuals and their parents. This technique resembles *standard crowding* (section 3.1.5). Every time the offspring is created, if it is fitter than the parents it is inserted in the population and the parent individual is removed; otherwise, the child individual is removed. Differently from standard crowding, deterministic crowding does not require the explicit definition of a distance between individuals, and that is why it is classified under lineage-based techniques, while standard crowding falls into the class of genotype-based methods. Deterministic crowding does not influence parent selection, and it is independent from the context.

Allopatric selection

Closely related to deterministic crowding, *allopatric selection* [Ton+12] puts all offspring generated during the same application of certain genetic operators in competition: only the fittest progeny is stored in the population, and then the standard fitness-based survival selection is applied. In a context where several genetic operators may produce a considerable number of children each, this technique is used to avoid early colonizations of the population by successful lineages. For a comparison with deterministic crowding, see Figure 3.6. Allopatric selection is a context-independent technique that acts on survival selection only.

Gender

Another diversity-promoting technique is to add a feature inside the EA, representing the *gender* of an individual: crossovers are then allowed to operate between individuals of different genders, only [All92; GLR03]. Variants of this approach try to exploit the presence of more than two sexes

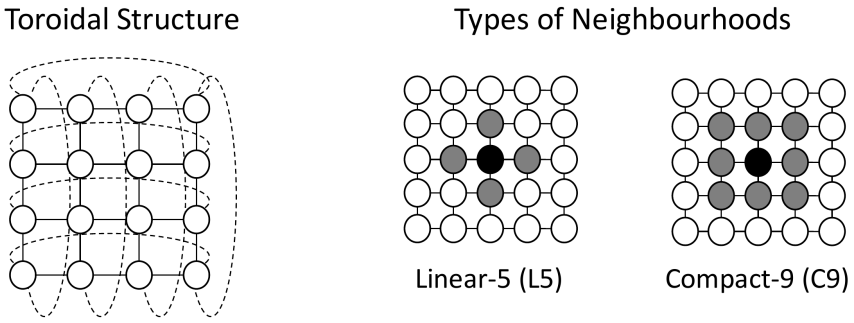


Figure 3.5: On the left, the classical structure of cellular EAs. Solutions are placed in a toroidal bi-dimensional grid. On the right, examples of neighborhoods. Neighborhoods are user-defined, and usually follow one of the portrayed structures: linear-n (covering up to n elements in the same row and column) or compact-n (considering the $n-1$ nearest individuals in every direction, including the diagonal). In practice, two of the most used are linear-5 and compact-9.

to further promote diversity [LE97], while other make use of different mutation rates for each sex [SB03]. In principle, gender could be managed even externally, through a reference table embedded in the algorithm. However, if the gender is inserted inside the genome and it is inheritable, its effect is to add a new trait with the only purpose to ease the evaluation of diversity, thus, the methodology should be classified as genotype-based. Several methodologies that are labeled with the word "gender" use the fitness values as discriminant [RA00], and should therefore be included into phenotype-based methodologies. All gender-based techniques are context-dependent and influence reproduction alone.

3.1.5 Genotype-based methodologies

A considerable number of methods exploit information at genotype level to promote diversity inside the population. Such techniques can be effective especially when it is straightforward to define a distance measure between different individuals: distances are often used to avoid overexploitation of peaks in the fitness landscape and to promote the generation of new solutions very *far* from the most successful ones.

The most prominent techniques in the field belong to the family of *niching methods*, derived from Holland's observations on fitness sharing and the successive refinements of Goldberg, Richardson and Deb [DG89; GR87; Hol75]. The basic idea is to achieve the emergence of artificial *niches* in the search space, following the paradigm of natural niches: in nature, a niche is defined as a subspace in the environment with a finite amount of physical resources, that can support different types of life. An example is reported in Figure 3.7.

Niching methods can be further divided into two classes [SK98]: *explicit neighborhood* methods, that require an explicit definition of the size of a niche through a parameter called *niche radius*; and *implicit neighborhood* methods, where the algorithm requires no information about the search space. It is important to notice that, without a distance metric defined between individuals at genotype or phenotype level, explicit neighborhood techniques cannot operate. All niching methods introduce an overhead in the evolutionary process, but the computational effort required to enforce diversity is usually negligible when compared to fitness evaluation in real-world problems.

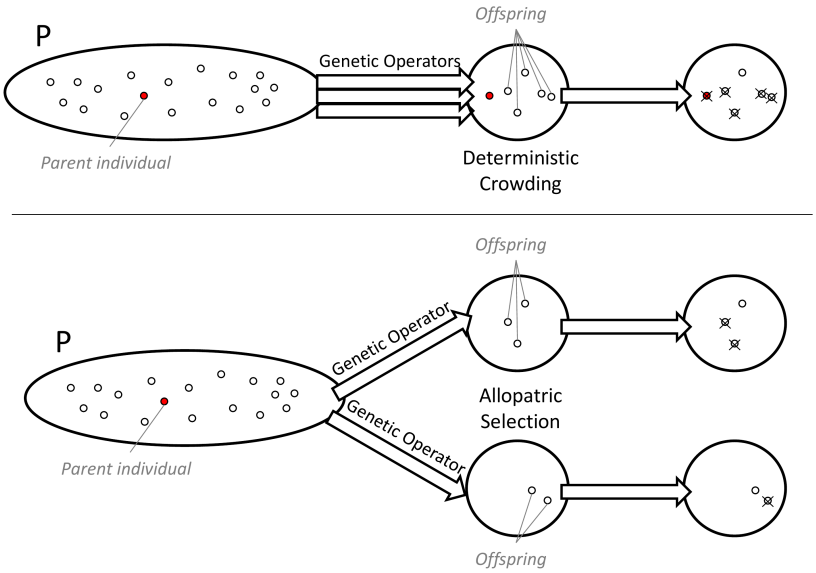


Figure 3.6: Deterministic crowding (**top**) and allopatric selection (**bottom**) compared. In deterministic crowding, both parents and offspring compete for a place inside the population; in allopatric selection, the competition is between children individuals generated during the same application of a genetic operator, only.

Besides niching techniques, many other research lines follow the idea of using information gathered from the genotype of the population to maintain and promote diversity.

Fitness Sharing

A well-known example of niching technique with an explicit niche dimension is *fitness sharing*. Sharing [McK00] reduces the attractiveness of densely populated regions of the search space, lowering the fitness value of individuals in the same niche by a value proportional to the number of individuals. Given an individual I_k and its fitness value $f(I_k)$, its new fitness with sharing $f'(I_k)$, can be expressed as follows:

$$f'(I_k) = \frac{f(I_k)}{\sum_{i=0}^{individuals} sh(I_k, I_{i \neq k})} \tag{3.4}$$

with

$$sh(I_a, I_b) = \begin{cases} 1 - (\frac{d(I_a,I_b)}{\sigma_s})^{\alpha} & d(I_a, I_b) < \sigma_s \\ 0 & d(I_a, I_b) \geq \sigma_s \end{cases} \tag{3.5}$$

where $d(I_a, I_b)$ is the distance function between two individuals I_a and I_b and α is a constant parameter which regulates the shape of the sharing function (commonly $\alpha = 1$). For its characteristics, fitness sharing influences both the probability of reproduction and survival of an individual, and it is context-independent since the selection of an individual is independent from other individuals already chosen for that purpose.

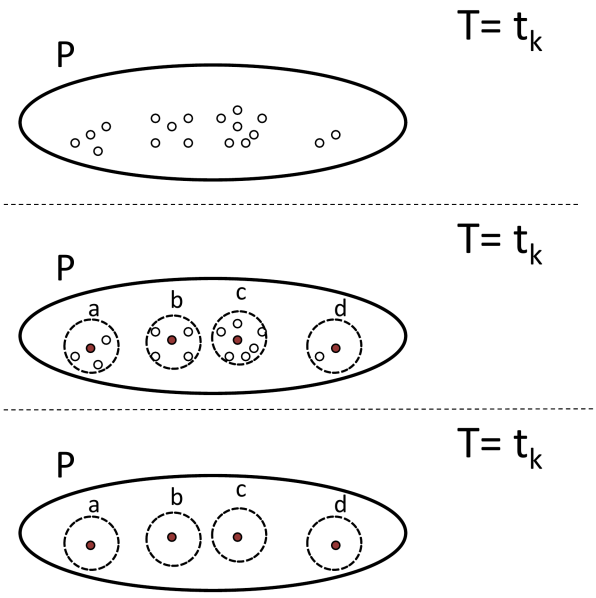


Figure 3.7: When niching is used, individuals are in competition with other individuals in the same niche, only. This method makes it possible for low-fitness solutions to survive, provided they are in a relatively unexploited area of the search space. For example, all individuals in niche **c** could have a higher fitness value than individuals in niche **d**, and in a classic scenario individuals in **d** would be removed from the population. When using niching, an individual in **d** is nevertheless preserved, since it was in a sparsely explored area of the search space.

Clearing

Clearing [Pét96] is part of the niching techniques with an explicit neighborhood: differently from sharing, however, it relies upon the concept of *dominant individuals* of the niche. Inside each niche, the k best individuals preserve their fitness, while all others have their fitness reset. As in the sharing method, individuals belong to the same niche if their distance in the search space is less than a dissimilarity threshold σ_s, here called *clearing radius*. The complexity of the procedure is $\mathcal{O}(qN)$, where q is the number of niches maintained during the search. Clearing is sometimes used along with other diversity-promoting techniques, such as in [BS02a]: it influences the selection probabilities for both reproduction and survival, and it is context-independent.

Standard Crowding

Standard crowding [De 75] is a niching technique with an implicit neighborhood: it makes use of a scheme where only part of the population reproduces and dies at each generation. Every time a new individual is created, a sub-population of size CF is randomly drawn, and the offspring replaces the most similar solution inside this sub-population. The *similarity* is measured at the level of genotype, and if the locality principle is not true, the methodology could lead to *replacement errors*, as noted by Deb and Goldberg [DG89]. This technique influences survival selection, and it is context-independent.

Restricted Tournament Selection

Among niching techniques with implicit neighborhoods, one of the most successful is *Restricted Tournament Selection* (RTS) proposed by [Har95]. RTS selects two individuals from the population, to undergo crossover and mutation; then, each offspring is compared with the closest element in a randomly drawn sub-population of size CF; finally, the winners are inserted into the global population, while the losers are discarded. This process is repeated $N/2$ times, where N is the size of the global population. The complexity of RTS is $\mathcal{O}(CF \cdot N)$, and it can grow up to $\mathcal{O}(N^2)$ if $CF = N$. This technique is context-independent, and it influences the survival probability of individuals.

Sequential Niching

An interesting variant of niching is the *Sequential Niching* [BBM93], whose basic idea is to alter the fitness of parts of the search space where good solutions have already been found. Differently from other fitness sharing approaches, it performs several iterations of the EA: the most promising points in the search space after each run are altered so to become less interesting in further executions. This method might be more performing than standard fitness sharing on multimodal search spaces, since there is evidence that mating between individuals on different peaks often leads to uninteresting solutions [Deb89]. Sequential niching is a context-independent technique that influences both the probability of reproduction and survival of individuals.

Reference points partitioning

To maintain diversity during a multi-objective optimization problem with several objective functions, Deb and Jain improved their *Non-Sorting Genetic Algorithm II*[5] (NSGA-II) creating the *Many-objective NSGA-II* (or NSGA-III, as it is sometimes called) [DJ13a; DJ13b].

The new tool uses a predefined set of *reference points*: reference points are initially set, either manually or with an automatic procedure; then, during evolution, each individual is dynamically associated to the closest reference point, partitioning the population; eventually, a traditional niching is used on each subset. This methodology is a context-independent technique that influences both the probability of reproduction and survival of individuals.

Delta entropy and pseudo entropy

In [ST08], the population is considered as a message, composed of the concatenation of all the individuals, with each gene corresponding to a symbol. Authors calculate the *entropy* associated to the message using Shannon's formula:

$$H = -\sum_{s \in P} f(s) \cdot \log(f(s)) \tag{3.6}$$

where s is a symbol (i.e., a gene) in the population P, and $f(s)$ is its frequency. The effect on the global population entropy caused by each individual is considered as an indication of the amount of diversity brought by it. Then, with a given probability candidate solutions are compared on their capability to increase the global entropy instead on their fitnesses.

The approach was called "pseudo entropy" because authors acknowledged that the computation of the population entropy was not fully correct. This problem was solved in [SSS11a], where the concept of *symbol* was further extended, taking into consideration genes and small sequences of genes.

[5] Other techniques used by NSGA-II can be found in section "Crowded-comparison operator" (3.1.6)

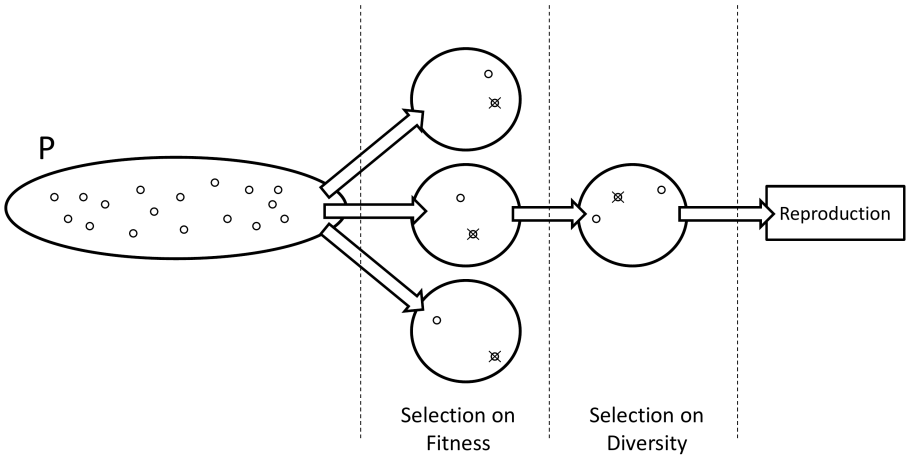


Figure 3.8: An example of two-level diversity selection. In the first part, a fitness-based tournament selection between two individuals is repeated 3 times. In the second part, the three winners are put in competition with each other, and the two most *diverse* are selected for mating.

Two-level diversity selection

When a genotype-level distance between individuals can be defined, diversity can be promoted by putting a selective pressure on diversity as well as fitness. This idea, presented in [BB02], employs a two-level tournament selection, where three individuals are first chosen, based on their fitness values, and subsequently the two with *maximum distance* are finally selected for reproduction. In this case, the distance is evaluated between the three fittest individuals, only, and the rest of the population is not considered. A scheme of the technique is proposed in Figure 3.8. The two-level diversity selection is context-dependent, and acts on the selection of the parents.

Tarpeian method

Named after the Tarpeian rock in Rome, the infamous execution place for traitors and criminals, this method proposes to randomly kill individuals if they do not satisfy a genotype-level metric [Pol03]. In the original paper, the purpose is to limit bloating in GP by lowering the selection probability of programs bigger than a user-defined threshold, but in general the technique can be used to promote diversity by favoring less fit individuals that score well on the considered genotype-level metric. The author argues that this method dynamically and non-deterministically creates *fitness holes* in the fitness landscape, and that it could be superior to just creating static *holes* with other bloat-controlling techniques. The Tarpeian method is context-independent, and it influences both survival and reproduction.

FOCUS and GDEM

Another interesting approach is to exploit the potential of *Multi-Objective Evolutionary Algorithms* (MOEA) by adding a diversity-related function among the objectives to optimize. The contribution of a single individual to the variety of the population is thus evaluated. This idea is first presented in [DWP01] and applied to GP in order to limit the growth of solutions over time, using a technique called *Find Only and Complete Undominated Sets* (FOCUS), where only non-dominated individuals are preserved in the population.

The *Genetic Diversity Evaluation Method* (GDEM) [TB03] makes use of a similar idea. Genetic diversity is used again as a second objective in a MOEA, but this time it is managed so that solutions with the same rank with regards to other objectives will not dominate one another, independently from the values of their diversity measure. In other words, individual $u(rank_u, diversity_u)$ dominates individual $v(rank_v, diversity_v)$ if and only if:

$$rank_u > rank_v \wedge diversity_u \geq diversity_v$$

Both these techniques have been applied to MOEAs, but in principle they could be used for single-objective optimization as well; both alter the probability of selecting an individual for survival; and they are context-independent since choosing a specific individual would not alter the probabilities of selecting a second one for the same purpose.

Diversifiers

A more refined and computationally expensive version of the *random immigrants* strategy (section 3.1.6), presented in [KB95], tries to fill *gaps* of under-represented areas in the search space with individuals, called *diversifiers*, created specifically for the task. If a two-dimensional distance metric d can be defined over a population of individuals, a distance space $D_G \subset \mathbb{R}^2_{0+}$ can be used to approximate areas of *relative emptiness*, where few genomes can be found. In particular, there are algorithms able to find the largest-area rectangle R_{max} that is axis-parallel to the x, y axis of D_G, whose opposite vertices are two genomes, in $\mathcal{O}(n \cdot log(n))$ where n is the number of individuals. The part of the genotype space R_{max} can then be used to generate individuals able to fall inside it.

This technique is context-independent and indirectly influences the probability of reproduction and survival of other individuals.

3.1.6 Phenotype-based methodologies

All techniques that operate directly in the fitness space, either artificially altering the fitness landscape, or relying upon fitness-level information to promote diversity are grouped under the label *phenotype diversity*. At a first glance, diversity preservation at fitness level might look impractical: if the fitness landscape of the problem is multi-modal, there could be several genotype-level points corresponding to the same fitness value, and enforcing a distinction could thus appear meaningless. Still, especially if a genotype-level distance measure is hard to conceive, even single-objective fitness can be exploited to enforce diversity.

Moreover, the thriving sub-field of MOEAs relies upon a multi-dimensional fitness space. Since MOEAs return a set of non-comparable candidate solutions, it is in the user's interest to obtain very diverse solutions, avoiding the concentration of individuals on some parts of this multi-dimensional fitness landscape, only.

Random immigrants

A simple but effective technique to maintain diversity, often adopted in problems where the fitness landscape is dynamic, is to periodically add *random immigrants* [Gre92] to the current population. Such individuals are randomly generated, thus offering fresh material for the genetic operators to exploit: usually, however, their fitness value is very small when compared to the other individuals in the population. For this reason, algorithms employing this technique often act on the survival selection, favoring the unfit but useful random immigrants for, at least, some generations [TY07].

The presence of random immigrants influences the probability of reproduction and survival of other individuals: both these techniques are context-independent.

Extinction

The difficulty to survive faced by *random immigrants* is mitigated by the *extinction* methodology [GFC99], that operates by periodically removing a significant amount of the population. More in detail, at each generation, a stress factor $\eta(t)$ is generated according to $\eta(t) \sim U(0, 0.96)$. Assuming a minimization problem, for each individual I_i the algorithm scales its fitness $f(I)$ to the interval $[\alpha; 1]$ on the basis of the following formula:

$$f'(i) = \alpha + (1 - \alpha) \cdot \frac{f(I_i) - f(I_{max})}{f(I_{min}) - f(I_{max})} \tag{3.7}$$

where $f(I_{max})$ and $f(I_{min})$ are the fitness of the worst and best individuals, respectively, and $\alpha \in [0, 1]$ controls the lower bound of the assigned fitness. The individuals with fitness values f' less than the stress factor are removed, and the empty slots are filled with a tournament selection between mutated variants of survived individuals. If no individual is killed, a percentage m of the population will be replaced by mutants, with a process called a *background extinction*.

This context-independent technique acts exclusively on the survival selection mechanism.

Crowded-comparison operator

Multi-objective optimization addresses all problems that give rise to a set of trade-off optimal solutions (known as *Pareto-optimal solutions*) [Deb05]. A trivial example of a multi-objective problem with two objectives is reported in Figure 3.9. MOEAs strive to find as many Pareto-optimal points as possible, because any two solutions on the Pareto front represent a trade-off between the objectives: when a large groups of solutions is returned, users are in a better position to make an informed decision.

For the same reason, it is interesting for the user to obtain solutions that are well distributed along the Pareto front, in order to have a more complete picture of the problem. The *crowding* mechanism introduced in the MOEA *Non-Sorting Genetic Algorithm II* (NSGA-II) [Deb+02] is designed to tackle this problem, effectively enforcing diversity on the Pareto front through a *Crowded-Comparison Operator* (CCO).

The CCO guides the selection process in various steps of the algorithm, trying to achieve a uniformly spread-out Pareto-optimal front. Every individual I in the population possesses two attributes, a *non-domination rank* I_{rank} and a *crowding distance* $I_{distance}$. The non-domination rank basically identifies the front the individual belongs to, and represents the primary source of comparison. If two individuals have the same non-domination rank, they are then compared on their $I_{distance}$, so that solutions located in less crowded regions are preferred.

NSGA-II uses an estimate of the density of solutions surrounding individual I, by taking the nearest neighbors as vertices of a cuboid, and then calculating the average distance from the vertices along each of the objectives, see again Figure 3.9. This method only operates in the objectives' space, so it can be applied independently from the genotype or phenotype representation of the individuals: there are even LGP tools that exploit this technique for multi-objective problems [SSS11a]. Furthermore, NSGA-III, the successor of NSGA-II, exploits the concept of ε-domination to adaptively discretize the Pareto-optimal front and find a better-distributed set of points [DJ13a; DJ13b].

Hierarchical fair competition

Sub-populations are the focus of *Hierarchical Fair Competition* (HFC) [Hu+05]: in this approach each sub-population tries to contain individuals of similar fitness values, promoting the best ones to

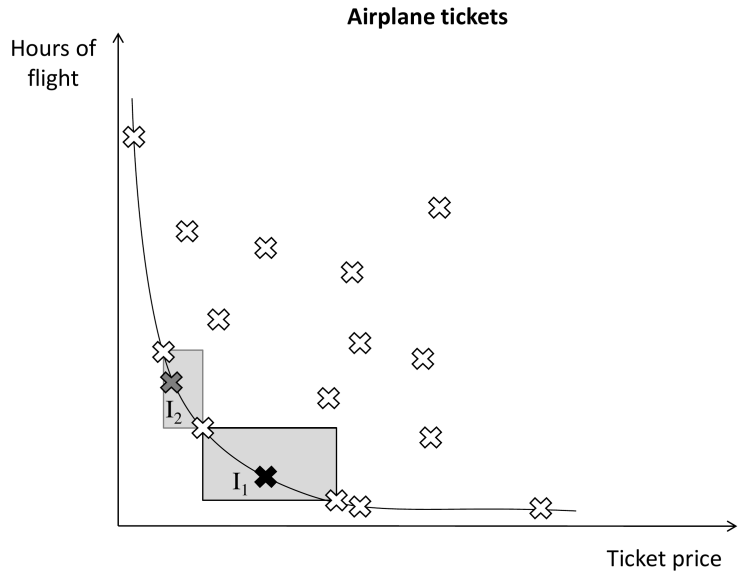


Figure 3.9: Sample multi-objective problem. Each solution, representing an airplane ticket for the same destination, is a trade-off in terms of price and hours of flight. Optimal, non-dominated solutions are on the line representing the Pareto front. Points in the top left of the curve represent cheap tickets with long flight times; points in bottom right represent expensive tickets with the shortest flights. The highlighted areas represent the cuboids used by NSGA-II to estimate the density of solutions surrounding individuals. The cuboid for individual I_1, in black, is clearly bigger than the cuboid of individual I_2, in gray; thus, individual I_1 will be preferred for selection.

upper sub-populations and demoting the worst ones to *lower* sub-populations (see Figure 3.10). The technique proves to be quite effective to help new patterns emerge in the gene pool without being immediately overwhelmed by existing already-adapted individuals. HFC can be seen as an enhanced version of the island model (section 3.1.4) and it shares the assumptions that a multi-start scheme would yield different solutions, however it is based on how individuals behaves rather than where they have been generated.

The main drawback of the methodology is that it leaves several parameters for the user to define, such as the number of sub-populations, the size and the two thresholds for *acceptance* and *upward migration*, for each sub-population. There exist HFC models that self-adapt all parameters, but a good user-defined regulation usually returns better results. Relying on an idea similar to the islands, this technique influences both reproduction and survival, and it is context-dependent.

Vector evaluated genetic algorithm

Besides NSGA-II's CCO (section 3.1.6), other strategies are employed by MOEAs to promote diversity inside the population. The *Vector Evaluated Genetic Algorithm* (VEGA) [Sch85], for example, evenly divides the mating pool into a number of parts equal to the number of objectives: each part is filled with individuals selected on a different objective. A derived technique proposed in [HL92] uses a similar division of the mating pool, considering different trade-offs between objectives as a weighted sum. The weights are encoded in the genotype, and evolved to search for multiple solutions simultaneously. Diversity within weight combinations is promoted by phenotype-level fitness sharing.

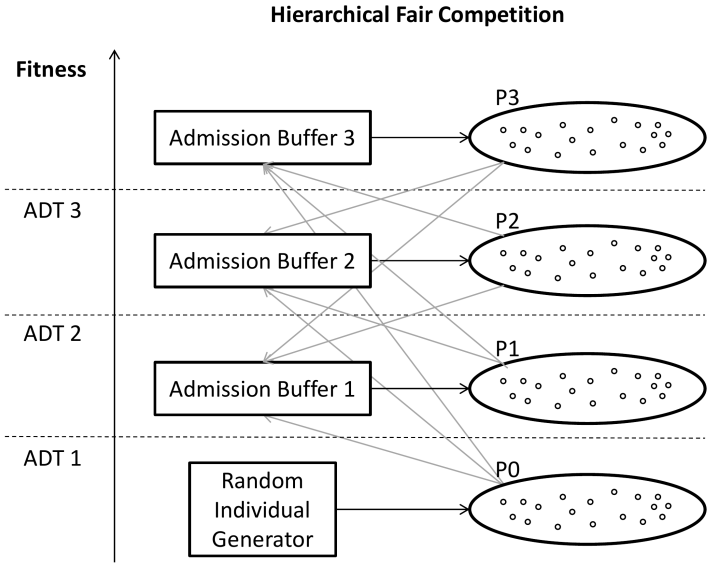


Figure 3.10: Hierarchical Fair Competition scheme. Depending on their fitness value, individuals are divided into sub-populations $P1...Pn$, synchronously or asynchronously through admission buffers, following user-defined *Admission Thresholds* (ADT). Fresh genetic material is added through a random individual generator.

Strength Pareto

This approach promotes diversity by using an external secondary population to store the non-dominated solutions [ZT99]. The *strength* of an individual in the secondary population is proportional to the number of other individuals covered by it, while dominated individuals are assigned a fitness based on the strength of individuals that cover them. The secondary population is updated at every generation and pruned by clustering if the number of the non-dominated individuals exceeds a predefined size.

3.1.7 Promoting diversity: a hands-on approach

The amount of different methodologies proposed to cope with the lack of diversity can be disconcerting for someone approaching EC. This section presents a few rules of thumb that could help practitioners to assess what technique may be best suited for their particular cases, together with hints and tricks to implement it, as source code is not generally available[6]. These spare indications

[6]Scholars willing to evaluate and compare on their own the listed methodologies would need to implement them in a common environment. A good framework that already includes islands and provides a convenient platform is *Open BEAGLE*, available under *GNU Lesser GPL* from `https://code.google.com/p/beagle/`. Moreover, the simplistic implementations and didactic videos showing the effects of few methodologies are available under Creative Commons *Attribution-NonCommercial-ShareAlike 4.0 International License* from `https://bitbucket.org/atonda/eatutorial/src/`.

Table 3.2: Summary of all techniques analyzed in this section, in alphabetic order, classified following the proposed taxonomy. In the *main element* column, $\mathscr{L}$ indicates lineage, $\mathscr{G}$ genotype, and $\mathscr{P}$ phenotype. In the *selection* columns, "P" stands for "parent" and "S" for "survival".

Sec	Methodology name	Main element	Selection P	Selection S	Context dep.
3.1.4	Aging	$\mathscr{L}$	yes	yes	no
3.1.4	Allopatric selection	$\mathscr{L}$	no	yes	no
3.1.4	Cellular EAs	$\mathscr{L}$	yes	yes	yes
3.1.5	Clearing	$\mathscr{G}$	yes	yes	no
3.1.6	Crowded-comparison operator	$\mathscr{P}$	yes	no	no
3.1.5	Delta/pseudo entropy	$\mathscr{G}$	yes	no	no
3.1.4	Deterministic crowding	$\mathscr{L}$	no	yes	no
3.1.5	Diversifiers	$\mathscr{G}$	yes	yes	no
3.1.6	Extinction	$\mathscr{P}$	no	yes	no
3.1.5	Find only and complete undominated set	$\mathscr{G}$	no	yes	no
3.1.5	Fitness sharing	$\mathscr{G}$	yes	yes	no
3.1.4	Gender	$\mathscr{L}$	yes	no	yes
3.1.5	Genetic diversity evaluation method	$\mathscr{G}$	no	yes	no
3.1.6	Hierarchical fair competition	$\mathscr{P}$	yes	yes	yes
3.1.4	Island model	$\mathscr{L}$	yes	yes	yes
3.1.6	Random immigrants	$\mathscr{P}$	yes	yes	no
3.1.5	Reference points partitioning	$\mathscr{G}$	no	yes	no
3.1.5	Restricted tournament selection	$\mathscr{G}$	no	yes	no
3.1.4	Segregation	$\mathscr{L}$	yes	yes	yes
3.1.5	Sequential niching	$\mathscr{G}$	no	yes	no
3.1.5	Standard crowding	$\mathscr{G}$	no	yes	no
3.1.6	Strength pareto	$\mathscr{P}$	yes	no	no
3.1.5	Tarpeian method	$\mathscr{G}$	yes	yes	no
3.1.5	Two-level diversity selection	$\mathscr{G}$	yes	no	yes
3.1.6	Vector evaluated genetic algorithm	$\mathscr{P}$	yes	no	yes

are to be intended as a starting point for further research, and not as absolute rules applicable to every circumstance.

The first step before considering diversity promotion is to understand if the problem at hand requires the effort. It may be assumed that, if a practitioner is questioning how to promote diversity, the evolutionary optimizer already got stuck in suboptimal solutions several times. However, if repeated experiments yield very different results, but the optimizer is able to find at least few acceptable solutions, then the fitness landscape is probably very rough, and the task is probably going to be hard. Nevertheless, in that situation lack of diversity would not represent the most taxing problem: it is recommended to revise the encoding and the fitness function. On the contrary, when repeated experiments yield very similar sub-optimal results, premature convergence may be the issue.

Indeed, it is not required to run the full-fledged evolutionary optimizer for performing this first check. A *random mutation hill climber* [MHF+93] is probably faster and equally effective to this end.

Moreover, the hill climber could be easily built using the same mutation operators already present in the core of the original optimizer, by stripping down the evolutionary process. Unfortunately, for many real-world industrial problems the time required to run such experiments would make them effectively useless. Thus, in several practical applications, the best option is to include some methodologies for promoting diversity as a default. It is generally better to resort to well established methodologies (section 3.1.7).

Even when dealing with quite complex representations where a mathematical analysis of the fitness landscape is impractical or too computationally expensive, it may be possible to obtain useful problem-specific knowledge that can be exploited inside future diversity-preservation mechanisms. In such cases practitioners could also try to prepare their own methodology for promoting diversity (section 3.1.7).

Out-of-the box methodologies

In absence of any domain-specific information on the problem, *Extinction* (section 3.1.6) is a methodology able to return sensible results with minimal effort. Additionally, it features few parameters to be tuned, namely, how to trigger an extinction, and the percentage of the population to be replaced. Several implementations of popular optimization algorithms make use of extinction or similar mechanisms to periodically remove a considerable part of the population.

Island models (section 3.1.4) represent another quick and simple, yet effective, way to promote diversity even when information about the target problem is scarce. This technique only needs few parameters to be set by the user, namely the size of each island and the modalities of migration, and can provide a first assessment of how adding diversity preservation can significantly improve the results. Additionally, it requires only minor modification to the original algorithm: for this reason, island models are often adopted even by commercial evolutionary software [SL09].

Both methodologies can be added to an existing algorithm with limited effort, although the latter could be slightly more complex to implement but easier to parallelize. To be fully effective, both rely on the crossover operator. After an *extinction*, good, fresh traits of new individuals may be merged with the genome of already good solutions, allowing to escape from the local optima. In *island models*, after migration the crossover operator could merge solutions exploiting different local optima, allowing to explore new regions of the search space.

If information about the problem is reliable, adding *niching* to the optimizer is a good solution. Techniques that use an implicit neighborhood, such as *restricted tournament selection* (section 3.1.5) or even *standard crowding* (section 3.1.5), can be easily adopted. Both techniques require to evaluate the distance between genotypes, but dos not rely on assumptions on the local optima distribution. However, if also a good estimation for the distance between local optima in the fitness landscape is available, it is recommended to use niching techniques that operate with an explicit neighborhood, such as *fitness sharing* (section 3.1.5) or *clearing* (section 3.1.5), can be applied to the problem.

Hybrid methodologies

As a general consideration, tackling diversity at the level of genotype is simpler, but operating at the level of phenotype is likely to be more effective. Fortunately, in several problems, genotypes and phenotypes are highly correlated. Two simple tests could help a practitioner to detect this situation: count the collisions and check whether the locality principle is fulfilled.

Collisions happen whenever two or more individuals that are different at the level of genotype are given exactly the same fitness value. A large number of collisions could seriously impair the effectiveness of artificial evolution, because the principle of *differential survival* is a cornerstone of

the whole process: different individuals should have different chances to survive, and this is not true if they share the same fitness value. In general, a high number of collisions is a sign that the fitness function should be redesigned and improved.

To check if the locality principle is fulfilled, the suggestion is to generate a set $\mathbf{P}$ of random individuals. Then, for $i = 1...n$ generate the sets $\mathbf{O_i}$ containing the offspring obtained applying exactly i mutation operators to each parent in P. Finally, examine the difference in the fitness values $\Delta(I_1, I_2)$ between individuals in $\mathbf{P}$ and individuals in $\mathbf{O_i}$ with respect to i. As a rule of thumb, the locality principle is satisfied if there is a correlation between i and $\Delta(I_1, I_2)$ with $I_1 \in \mathbf{P}, I_2 \in \mathbf{O_i}$

If it is possible to assign a *diversity bonus* to each individual, for instance, by measuring the average distance between itself and the whole population, or by measuring population diversity with and without it, then diversity can also be pursued as an explicit goal using multi-objective algorithms [DWP01; TB03]. The consequence of searching for a Pareto-front with some highly-fit and some highly-diverse solutions is to promote diversity during the evolutionary process.

Alternatively, it may be possible to promote diversity by tweaking selection probabilities. The *real* (i.e., the original) fitness could be scaled, or in any other way modified, according to a diversity measure. The main advantage of this technique is that it can be implemented with limited coding effort, because, except for the fitness calculation, the algorithm does not need to be modified. However, the amount of scaling needs to be carefully considered. It is also possible to tweak the selection by changing how individuals are compared, creating artificial *holes* in the fitness landscape. Literature reports several successful stories: in [EN01], such holes are created by taking into account the size of individuals, trying to favor smaller, less effective GP trees over bigger and more performing ones; in [Pol03] the fitness of a certain proportion of the offspring is radically zeroed; while in [CSS05] original fitness values are used with probability h, while with probability $1-h$ an individual able to diversity of the population is preferred.

These techniques do not require the definition of a distance between all pairs of individuals, although such a metric could be easily used to implement them. Even basic information on the problem, that might not be directly exploited to measure the difference between two candidate solution, may be nevertheless capitalized on to introduce randomness in individual selection and indirectly prevent premature convergence. For example, the *tarpeian method* (section 3.1.5) was originally used to limit bloating in experiments with individuals of non-fixed size, but the possibility to create non-deterministic *fitness holes* in the fitness landscape should not be underestimated. If a metric other than fitness value is available to compare two individuals, it could be employed to promote diversity.

3.1.8 Conclusions

As the *divergence of character* is an essential element in natural evolution, the *lack of divergence of character* is an endemic problem in evolutionary optimization. This section surveys notable methods to overcome this problem, promoting diversity. The algorithms are classified on the basis of the elements considered (lineage, phenotype or genotype), the type of selection influenced (reproduction and/or survival) and their dependency from the context.

Looking at the resulting compendium, it is easy to identify the recurring ideas that have been exploited through different domains: restrict the interaction of individuals to random sub-groups for mating (e.g. *gender*), survival (e.g. *niching*) or both (e.g. *islands*, *segregation*); strengthen the competition between solutions that are closely related (e.g. *allopatric selection*, *deterministic crowding*); artificially alter the fitness values of individuals, taking into account their contribution

to population's diversity as well as their goodness with regards to the problem's objectives (e.g. *FOCUS*, *genetic diversity evaluation method*, *fitness holes*, *Tarpeian method*); and regularly add entirely new genetic material to the population (e.g. *random immigrants*, *diversifiers*, *hierachical fair competition*).

In fact, it must be remarked that it is often hard to quantify the improvement that a diversity-promoting technique can add to an evolutionary optimization process. While several extremely challenging test functions are available, modern EAs often feature a set of mechanisms aimed at increasing their efficiency, and separating the contribution of each to the final result can be quite complex. The authors' feeling is that there is still a lack of benchmarks able to consistently evaluate diversity preservation in evolutionary optimization, and especially finding a test case that could be generalized to all different paradigms in EC could constitute an interesting and challenging research line.

Several well-known EAs with proved effectiveness include indirect mechanisms for promoting diversity. However, such techniques are so deeply embedded – or have so many side-effects – that they cannot be easily extrapolated from the original algorithm and analyzed separately. Paradigmatic examples are *generational approaches* [ES10] (or *comma strategies* in ES), the core of CMA-ES [Han06], *competitive co-evolution*, used for example in [SL08], or the effect of increasing the *crossover rate* in DE [CBM14].

From a practitioner's perspective, it is interesting to notice how the great majority of techniques try to exploit information at genotype-level, the easiest to deal with; that several diversity-promoting techniques can be applied to the same algorithm at the same time, acting at different levels (lineage and genotype, for example); and that not necessarily methods that are more complex to implement return better results.

A considerable number of EAs have been adopted by practitioners of other domains as effective means to find reasonable solutions for problems that could not be tackled with classical optimization techniques. Looking at the most popular software, it is striking to notice how almost all provide some default value for the parameters, so that the users can try an approach *out of the box*, without having to tweak the population size or the activation probabilities for the genetic operators.

While diversity preservation is essential, the main challenge for scholars is devising general methodologies that could be applied seamlessly, trying to limit the number of parameters the user has to set, or providing a few default values that work in most cases.

Acknowledgments

The authors would like to thank Evelyne Lutton for her invaluable suggestions and insightful comments.

3.2 Measuring Diversity for Complex Genomes

Defining a distance measure over the individuals in the population of an Evolutionary Algorithm can be exploited for several applications, ranging from diversity preservation to balancing exploration and exploitation. When individuals are encoded as strings of bits or sets of real values, computing the distance between any two can be a straightforward process; when individuals are represented as trees or linear graphs, however, quite often the user must resort to phenotype-level problem-specific distance metrics. This section presents a generic genotype-level distance metric for Linear Genetic Programming: the information contained by an individual is represented as a set of symbols, using n-grams to capture significant recurring structures inside the genome. The difference in information

between two individuals is evaluated resorting to a symmetric difference. Experimental evaluations show that the proposed metric has a strong correlation with phenotype-level problem-specific distance measures in two problems where individuals represent string of bits and Assembly-language programs, respectively.

3.2.1 Introduction

Defining a *distance metric* in an Evolutionary Algorithm (EA) is both theoretically sound and practically challenging – and ultimately useful. Being able to quantify the similarity of two individuals can be used to promote diversity inside the population's gene pool, to avoid the over-exploitation of niches in the fitness landscape, to balance exploration and exploitation, and ultimately to ease the premature convergence problem. Not surprisingly, the topic has been actively investigated by the evolutionary community for many years.

From the theoretical point of view, two different aspects must be examined when a distance is defined: the level at which it is calculated; and the purpose for calculating it. On the other hand, for the practitioner the complexity involved in the calculation is the key point.

The level at which a distance is defined may be: genotype, phenotype, or fitness. The first and the last are probably the most easily definable: the genotype corresponds to the internal representation of the candidate solution; the fitness is ultimately the number, or numbers, returned by its evaluation. In biology, the phenotype is the sum of all the observable characteristics of an organism that result from the interaction of its genotype with the environment. It is hard to translate the concept in Evolutionary Computation since the environment is missing, being indirectly defined by its effects through the fitness function. Yet, in several classical problems – where an individual is a fixed-length bit string, for instance – the need to distinguish between genotype and phenotype is reduced. As a consequence, several works assimilate the fitness to the phenotype.

In many other cases identifying phenotype and fitness is not an option. The fitness is a synthetic information, and may not be able to convey the necessary data to separate individuals. Even in the simplistic one-max problem two solutions may have the same fitness without sharing a single gene (e.g., "0011" and "1100"). Moreover, the very same solution can be encoded in different ways. If the individual is the movement of a robot, for instance, a single 90° turn could also be represented as two consecutive 45° ones. More generally, whenever the genotype cannot be evaluated directly by the fitness function, but needs to be transformed into something else, fitness and genotype should be distinguished. In such scenarios, the phenotype could be easily defined as the "something else" in which the genotype needs to be transformed into.

The final goal for measuring the distance between individuals plays an important role. If the distance metric is used to avoid that a region of the search space becomes overly populous, then it should be defined at the level of phenotype. However, phenotype-level distances are often difficult to define or practically impossible to calculate. Remarkably, NSGA-II, the widely used multi-objective evolutionary optimizer, adopts a sharp and computationally efficient mechanism called *crowding distance* to scatter individuals [Deb+02]. Here, the crowding distance may rely exclusively on information from the fitness because the genotypes are fixed-length arrays of real numbers, requiring no transformation; and the fitness is composed of several different values, reducing the loss of information.

Conversely, if the distance metric is used to promote diversity in the gene pool, balancing exploration and exploitation, it could be based on the genotype. For example, in [Mau84] solutions are encoded as fixed-length bit strings and a metric based on the hamming distance is used to assess

the global diversity inside the population. When the phenotypes are sets of real values of fixed size, computing the distance between them is relatively straightforward, albeit not trivial [Cor+12]. However, phenotypes in Genetic Programming (GP) [Koz92], Linear Genetic Programming (LGP) [BB07] and other complex EAs pose a harder challenge: calculating the similarity between two binary trees, linear graphs, or generic compound structures is an open problem.

This section proposes a new distance metric easily usable in different types of LGPs. The distance is calculated quite efficiently at the level of genotype, yet it is able to convey a considerable amount of information about the individual. Thus, it may be used to reduce crowding in place of a phenotype-level distance. The proposed approach computes the *symmetric difference* [BB91] between the global information contained in two individuals; while the global information itself is evaluated resorting to the concept of n-grams [Sue79].

Experimental results demonstrate that the proposed distance is highly correlated with other phenotype-level problem-specific distance measures, both where individuals are string of bits and Assembly language programs. Further experiments show that exploiting the proposed metric to perform fitness sharing in a sample problem produces results comparable to using a phenotype-level metric.

3.2.2 Linear genetic programming

LGP is a variant of GP that evolves computer programs as sequences of instructions. It was introduced by Markus Brameier and Wolfgang Banzhaf between the late 90s and the early 2000s [Ban+97] [BB07], after the seminal work of Friedberg [Fri58]. A traditional, *Koza-style* GP encodes individuals as trees. Such tree GPs – or TGPs, as they are sometimes called – are commonly used to evolve mathematical expressions: leaves correspond to *terminals*, such as input values, constants or variables; inner nodes represent *functions*. Quite differently, LGP evolves a simplified list structure that represents a sequence of instructions in an imperative programming language. The resulting individuals represent real programs, although in a simplified language, that can grow to a significant complexity. Since their appearance, LGPs have been widely used to solve both practical problems and perform theoretical studies.

In LGP the difference between genotype and phenotype becomes fully apparent. The *genotype* is the internal, list-based representation; the *phenotype* is the actual program resulting from the interpretation of the genotype; the *fitness* is the final result of the evaluation of the program (Figure 3.11).

3.2.3 Symmetric difference

In set theory, the symmetric difference [BB91] of two sets A and B is defined as

$$A \triangle B = A \cup B - A \cap B \tag{3.8}$$

In practice, the symmetric difference contains all elements which are in either of the sets and not in their intersection. The Venn diagram of the symmetric difference is reported in Figure 3.12.

Considering the set as the *information* carried by an individual, the symmetric difference appears a plausible formalization of the intuitive idea of distance: when two sets are almost completely overlapping, their symmetric difference is very small; when they are completely separated, it is big.

Moreover, the symmetric difference exhibits useful properties for a distance: it is commutative ($A \triangle B = B \triangle A$); and the empty set is neutral ($A \triangle \emptyset = A$ and $A \triangle A = \emptyset$). The symmetric distance is also associative, but this fact is negligible in this application.

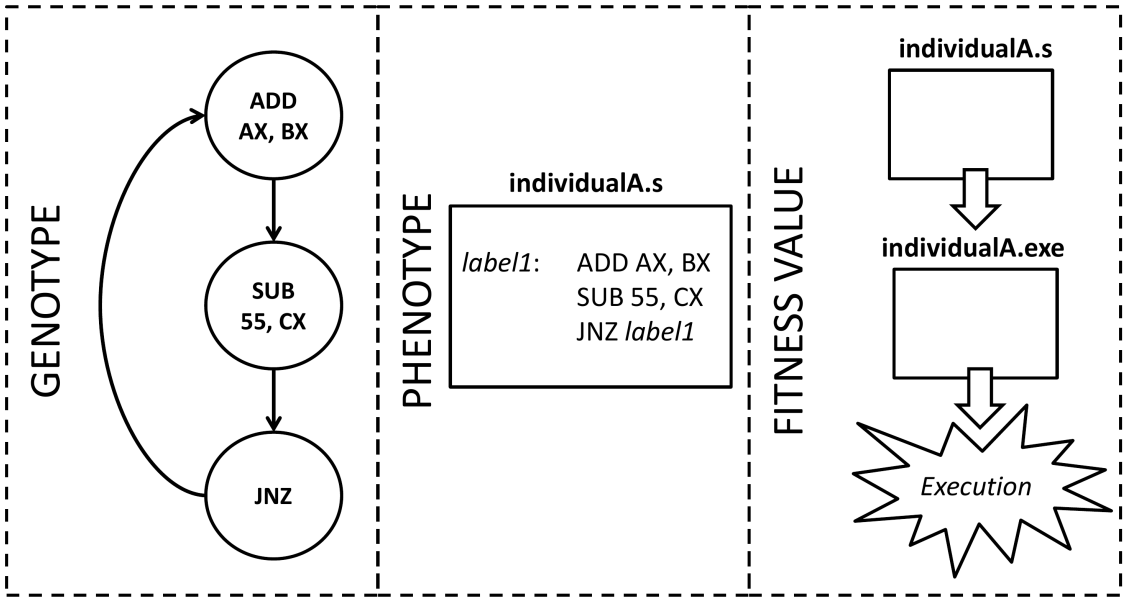


Figure 3.11: Distinction between genotype, phenotype and fitness value in an example with LGP used for Assembly language generation.

3.2.4 Fitness sharing

When a reliable distance metric is defined, one useful application is to exploit it for *fitness sharing*, one of many methods to enforce diversity inside the population of an EA [RB97; SK98].

The general idea of fitness sharing is to artificially decrease the fitness of individuals in crowded areas of the search space. The fitness f_i of an individual I_i is modified in a fitness $f_i' = f/m_i$, where m_i is dependent upon the number of individuals in a given radius σ_s from individual I_i, and their distance from the individual itself. In particular,

$$m_i = \sum_{j=0}^{N} sh(I_i, I_j) \tag{3.9}$$

where N is the number of individuals in the population, and $sh(I_i, I_j)$ is defined as

$$sh(I_i, I_j) = \begin{cases} 1 - (\frac{d(I_i,I_j)}{\sigma_s})^{\alpha} & d(I_i, I_j) < \sigma_s \\ 0 & d(I_i, I_j) \geq \sigma_s \end{cases} \tag{3.10}$$

where σ_s is once again a user-defined radius, and $d(I_i, I_j)$ is a distance measure applicable to the individuals' representation. α is a constant parameter that regulates the shape of the sharing function. In many practical cases $\alpha = 1$, with the resulting sharing function referred to as the triangular sharing function [Gol89].

3.2.5 Shannon entropy

In information theory, entropy quantifies the expected value of the information contained in a message. Shannon et al. [Sha+49] define entropy as the average unpredictability in a random

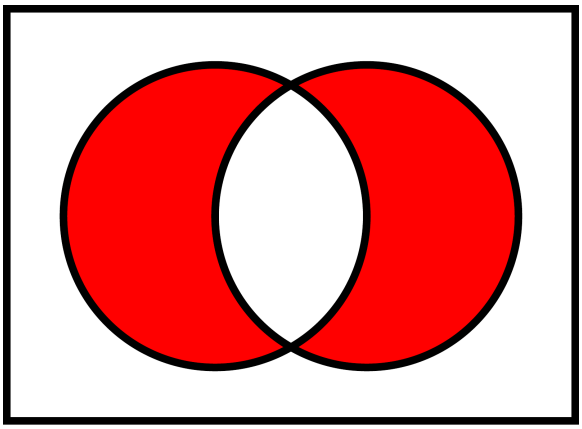


Figure 3.12: Venn diagram of the symmetric difference. The area corresponding to $A \triangle B$ is depicted in grey.

variable, which is equivalent to its information content. The Shannon entropy H of a discrete random variable X with possible values $\{x_1, ..., x_N\}$ is defined as

$$H(X) = -\sum_{i=1}^{n} P(x_i) log_b(P(x_i)) \tag{3.11}$$

where $P(x_i)$ denotes the probability for variable X to assume value x_i, and b is a user-selected base for the logarithm.

3.2.6 Proposed Approach

In LGP, Shannon entropy can be effectively used as a metric to assess the diversity in a population at a given generation [CSS05]. The entire population is considered a message, and each allele appearing in an individual is a symbol: entropy is then computed on the basis of the number of different symbols and their occurrences.

In a preliminary work [ST08], a variant of this approach is sketched. Instead of considering just the alleles of each gene, their disposition inside the individual is also taken into account. A symbol is no longer considered equivalent to a single allele, but to the allele and its position inside the individual instead. Following the idea that recurring structures might possess meaning, *n-grams* of nodes are also regarded as symbols. An n-gram is a group of n items from a longer sequence. For example, `a b`, `b c` and `c d` are all 2-grams from the sequence `a b c d`, while `a b c` and `b c d` are 3-grams. Very often n-grams are used for the purpose of modelling the statistical properties of sequences, particularly natural language [Sue79].

Building on the same principles, a generic genotypic *Universal Information Distance* (UID) for individuals in LGP is proposed. Considering two individuals I_i and I_j, the UID is defined as

$$UID(I_i, I_j) = |S(I_i) \triangle S(I_j)| \tag{3.12}$$

where $S(I)$ represents the set of symbols in individual I, $\triangle$ is the symmetric difference as defined in Equation 3.8, and the operator $|S|$ denotes the cardinality of set S.

In other words, the UID between two individuals is the number of distinct symbols they do not have in common. Intuitively, when two individuals share many common symbols, the UID will be

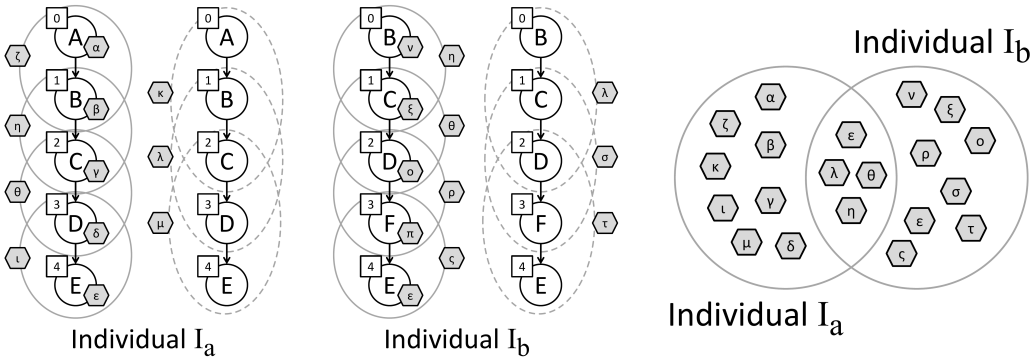


Figure 3.13: Example of symbols computed for alleles and $(2,3)$-grams for two individuals. Symbols are represented as Greek letters inside hexagons, alleles as Roman letters inside circles, while their position in the individual is reported in a square. The symbols common to the two individuals are ε (corresponding to allele E in position 4), η (2-gram $B-C$), θ (2-gram $C-D$) and λ (3-gram $B-C-D$). The UID between the two individuals is thus $|S(A) \triangle S(B)| = |S(A) \cup S(B) - S(A) \cap S(B)| = 16$

small; on the contrary, if they have no symbols in common, their UID will be high. An example is reported in Figure 3.13.

When used in practice, symbols for each individual are computed resorting to a hash function of the n-grams and alleles. It is interesting to notice how the proposed UID, that acts at genotype level and is quite straightforward to compute, could provide the same information of more computationally intensive distance metrics that are evaluated at phenotype level: thus, UID could be used for fitness sharing, delivering the same results as problem-specific metrics.

3.2.7 Experimental evaluation

The correlation between the proposed UID and two phenotypic distance metrics is examined, in two problems where individuals are encoded as strings of bits, and as Assembly language programs, respectively. Experiments with an evolutionary toolkit that supports LGP [SSS11b] are then performed for the two problems, and the effectiveness of UID for fitness sharing is compared to the previously considered phenotypic distance metrics.

In the all the experiments, the computation of UID is limited to n-grams of order 2 and 3, as a trade-off between computational efficiency and thoroughness of the approach. Symbols are computed resorting to the DJB[7] hash function.

The proposed approach is tested on two benchmarks: NK-landscapes, a NP-complete problem where individuals are represented a strings of bits, and a simple Assembly-language generation task.

NK-landscapes

In the NK-landscapes problem [KW89], the individual is a string of bits of fixed length: both the overall size of the fitness landscape and the number of its local optima can be adjusted by tuning the two parameters, N and K. Generally speaking, values of K close to N create more irregular landscapes. Albeit simple, this benchmark is widely studied in the optimization community, because

[7] http://cr.yp.to/djb.html

it is proven to be NP-complete [Wei96]. In the following experiments, values of N and K are very close, in order to obtain a fairly rugged fitness landscape.

Assembly language generation

This second set of experiments targets a simple Assembly language generation problem: the fitness of a program is the number of bits set to 1 in registry *%eax* at the end of its execution.

During the Assembly-generation problem, the minimum length of the variable part of a program is set to 1 instruction, the while the maximum length is set to 1500 instructions. For the initial population, individuals are generated in order to possess an average of 70 instructions, with a large standard deviation of 60 in order to efficiently sample the search space. Table 3.3 recapitulates the possible genes (instructions) appearing in the individuals.

Gene	Parameters	Prob.
<ins> <sreg>, <dreg>	**ins**={addl, subl, movl, andl, orl, xorl, compl}, **sreg**={%eax, %ebx, %ecx, %edx}, **dreg**={%eax, %ebx, %ecx, %edx}	0.33
<ins> <scon>,<dreg>	**ins**={addl, subl, movl, andl, orl, xorl, compl}, **scon**={integer in (0,255)}, **dreg**={%eax, %ebx, %ecx, %edx}	0.33
<ins> <reg>	**ins**={incl, decl, notl}, **reg**={%eax, %ebx, %ecx, %edx}	0.33

Table 3.3: Possible genes appearing inside the individuals during the Assembly generation problem. For each gene, all variables and corresponding values are listed, as well as the probability of occurrence.

The fitness function used for this experiment is based on both the result of a candidate program's execution and the length of its code, and it is defined as

$$f(I) = 10^4 \cdot (\sum_{i=0}^{N=31} \%eax[i]) + max(0, 10^4 - length(I)) \tag{3.13}$$

where $\%eax[i]$ is the value of the ith bit of register *%eax*, while $length(I)$ represents the number of instructions of candidate program I. Thus, the most important objective is to set to 1 bits in register *%eax*, while a small bonus is assigned to individuals that perform the task with a moderate number of instructions.

Correlation

An important result that it is possible to immediately esteem looking at the figures is how much the proposed UID distance is well correlated with problem-specific phenotype-level distances. Figure 3.14 plots the UID against the standard Hamming distance for 500 random 50-bit individuals[8]. The cloud of points does not stretch down to the origin, nor up to the maximum because it is quite unlikely to find two identical strings, or two completely different ones, in a random pool.

Figure 3.15, on the other hand, plots the same data for all individuals generated during a run, until the optimal solution is reached. Since there is a strong similarity between all individuals in the

[8]Several values are overlapping.

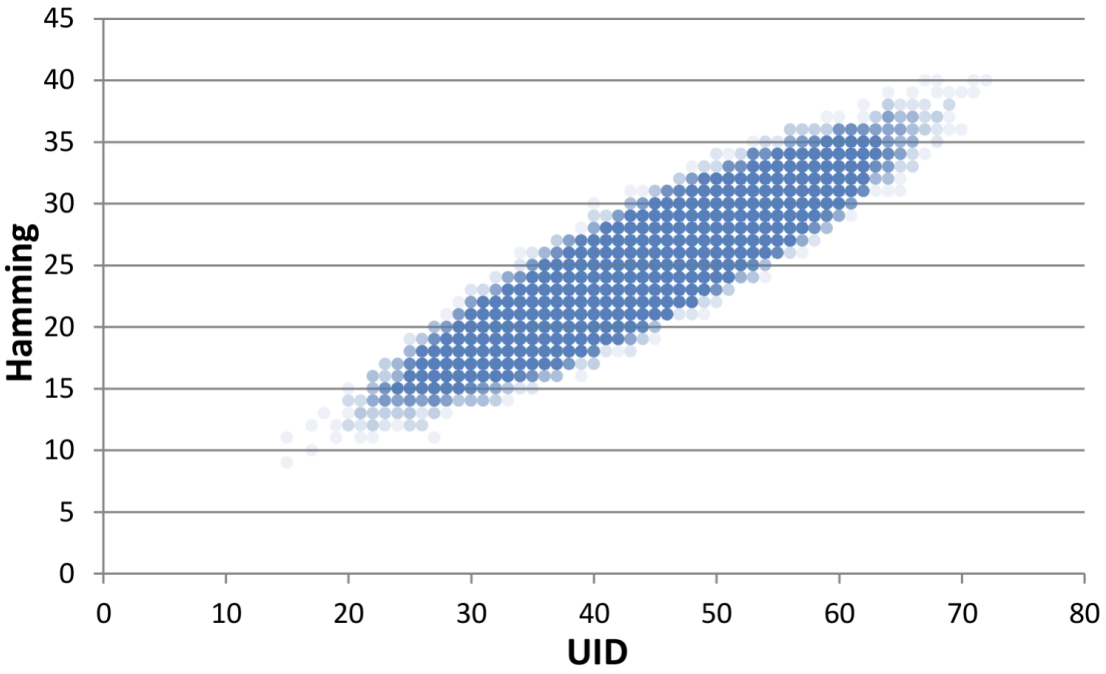


Figure 3.14: Correlation between the proposed UID distance and hamming distance in the standard OneMax problem (50 bits)– Sample of 500 random individuals.

same parental group, the cloud stretches down to distance zero. The correlation is even more evident than in the preceding example.

Figure 3.16 plots the proposed distance against the Levenshtein, or *edit*, distance for 500 random programs on the Assembly OneMax problem. Here, differences are more subtle and the number of overlapping values is reduced compared to the previous case. The triangular shape of the cloud is indicative: the two distances are better correlated for low values – that is, exactly when they are more useful: distinguishing between closely related individuals is in fact quite harder than discriminating between very different ones. The Levenshtein distance is a computationally expensive metric that can be computed only at the level of phenotype. The proposed UID, on the contrary, can be efficiently calculated at the level of genotype and it is effective in estimating the distance between *similar* individuals.

Fitness sharing

Since the proposed metric shows a heavy correlation with phenotypic distance metrics, its effectiveness can now be tested in multi-modal problems where a fitness sharing might help to spread the individuals over the search space. The aim of the following experiments is to show how using the proposed UID for fitness sharing delivers the same results as employing more rigorous problem-specific distance metrics, that are also more computationally expensive.

The first experiments are performed on the NK-landscapes (or NK-model) benchmark, tuned to obtain a rugged fitness landscape: the UID is compared to a classical Hamming distance [Ham50]. A second set of experiments is then executed on a simple Assembly-language generation problem, where the objective is to obtain a program able to set all bits of register *%eax* to 1. In this latter tests,

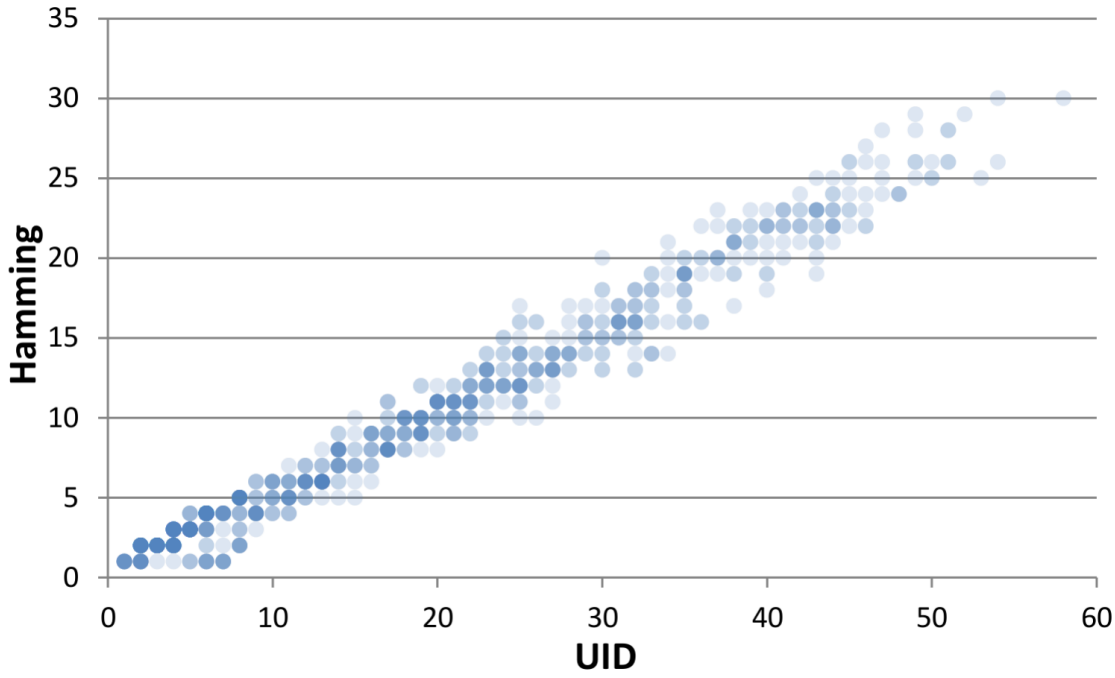


Figure 3.15: Correlation between the proposed UID distance and hamming distance in the standard OneMax problem (50 bits) – Individuals generated during a run.

the UID is weighted against the Levenshtein distance [Lev66].

In the LGP tool used for the experiments [SSS11b], μ is the population size; λ represents the number of genetic operators applied at each generation, rather than the offspring size; and σ is the *strength* of the mutation operators. Each time a mutation operator is applied to an individual, a random number r in the interval $(0,1)$ is generated: if $r < \sigma$, the operator creates another mutation in the same individual, and a new r is generated.

It is important to notice that the parameters of the tool used in the trials have not been tuned to optimality, since the main objective of this experimental evaluation is to assess whether the fitness sharing mechanism behaves comparably when different distance metrics are applied with an equivalent radius, even with sub-optimal settings.

NK-landscapes

Parameters of the LGP tool used in the experiments are reported in Table 3.4. The mutation operator in this case simply changes the value of a random gene, while the one-point crossover selects a random cut point.

For each landscape, 10 experiments are run with the Hamming distance, and 10 with the UID, respectively, using equivalent radius measures derived from the correlation described in Subsection 3.2.7. At the end of the evolution, the fitness of the best individual (*online fitness*) and the average fitness of the final population (*offline fitness*) are compared.

From results in Table 3.5 it is noticeable how, for the same NK-landscape, the final online and offline fitness values are very close, as well as the average distance value between individuals in the population. In fact, running a two-sample Kolmogorov-Smirnov test on corresponding distributions

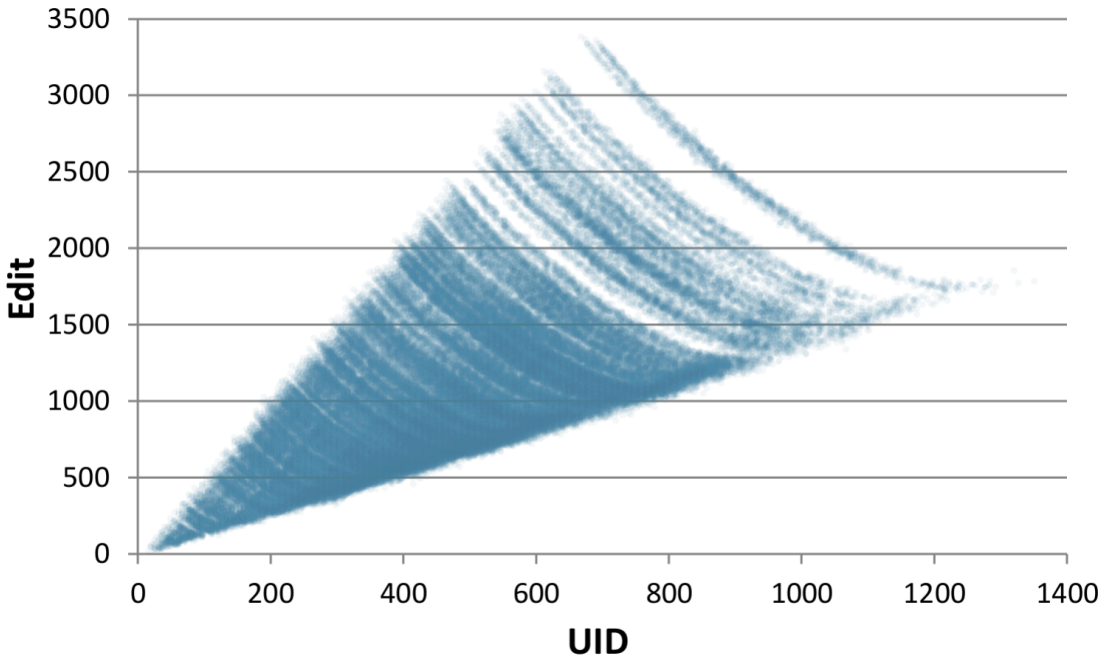


Figure 3.16: Correlation between the proposed UID distance and the Levenshtein distance in the Assembly OneMax problem (32 bits) – Sample of 500 random individuals.

Parameter	Value	Parameter	Value
μ	32	P(one-point crossover)	0.5
λ	32	P(mutation)	0.5
σ	0.5	Max generations	50

Table 3.4: Parameters used during the experiments with fitness sharing in the NK-landscapes benchmark.

for an equivalent radius reveals that the distributions are undistinguishable with $p < 0.01$.

Assembly OneMax

This second set of experiments targets a simple Assembly language generation problem: the fitness of a program is the number of bits set to 1 in registry *%eax* at the end of its execution. Table 3.6 summarizes the parameters used for the LGP tool in this experiment.

The mutation operator, in this case, can add a random instruction; remove a random instruction; or change one or more parameters inside a random instruction, with equal probability. The crossover can operate on one or two cut points, with equal probability.

Table 3.7 shows the results over 10 experiments with each parameter configuration. At the end of the evolution, the fitness of the best individual (*online fitness*) and the average fitness of the final population (*offline fitness*) are compared.

Again, the results for an equivalent radius are indistinguishable through a two-sample Kolmogorov-Smirnov test with $p < 0.01$.

Fitness sharing with Hamming distance					Fitness sharing with UID				
Radius (Hamming)	Online fitness (avg)	StDev	Offline fitness (avg)	StDev	Radius (UID)	Online fitness (avg)	StDev	Offline fitness (avg)	StDev
N=16, K=14, seed=1234567890									
5	0.6710	0.0167	0.3655	0.0256	10	0.6739	0.0326	0.3798	0.0391
N=16, K=14, seed=4242424242									
5	0.6774	0.0318	0.4094	0.0142	10	0.6948	0.0304	0.4099	0.0235
N=16, K=15, seed=1234567890									
5	0.6543	0.0228	0.3819	0.0124 10	0.6468	0.0109	0.3901	0.0137	
N=16, K=15, seed=4242424242									
5	0.6770	0.0209	0.3912	0.0352	10	0.6671	0.0256	0.4067	0.0316

Table 3.5: Results for the set of experiments on the NK-landscapes benchmark. Experiments with fitness sharing with the Hamming distance (**left**) and the UID (**right**); experiments with a corresponding radius are reported on the same line.

Parameter	Value	Parameter	Value
μ	10	P(crossover)	0.25
λ	7	P(mutation)	0.75
σ	0.7	Max generations	10

Table 3.6: Parameters used during the experiments with fitness sharing in the Assembly language generation problem.

3.3 Symbolic Regression of Dynamical Systems

Symbolic regression has many successful applications in learning free-form regular equations from data. Trying to apply the same approach to differential equations is the logical next step: so far, however, results have not matched the quality obtained with regular equations, mainly due to additional constraints and dependencies between variables that make the problem extremely hard to tackle. This chapter illustrates a new approach to dynamic systems learning. Symbolic regression is used to obtain a set of first-order Eulerian approximations of differential equations, and mathematical properties of the approximation are then exploited to reconstruct the original differential equations. Advantages of this technique include the de-coupling of systems of differential equations, that can now be learned independently; the possibility of exploiting established techniques for standard symbolic regression, after trivial operations on the original dataset; and the substantial reduction of computational effort, when compared to existing ad-hoc solutions for the same purpose. Experimental results show the efficacy of the proposed approach on an instance of the Lotka-Volterra model.

3.3.1 Introduction

In recent years, Genetic Programming (GP) gained popularity as an effective optimization technique [SL09], and its capabilities of automatically uncovering hidden relationships in datasets and produc-

Fitness sharing with Levenshtein distance					Fitness sharing with UID				
Radius (Levenshtein)	Online fitness (avg)	StDev	Offline fitness (avg)	StDev	Radius (UID)	Online fitness (avg)	StDev	Offline fitness (avg)	StDev
3	325,927	7,205.14	312,903	19,800.8	2	320,919	12,608.4	297,899	32,800.8
5	324,939	7,992.28	309,902	27,989.6	3	324,949	8,008.23	315,909	17,616.6
10	318,909	11,422.8	292,901	20,998.7	5	314,930	15,015.5	285,923	32014.3

Table 3.7: Results for the set of experiments on the Assembly-language generation benchmark. Experiments using fitness sharing with the Levenshtein distance (**left**) and the UID (**right**); experiments with a corresponding radius are reported on the same line.

ing rules to control complex systems haves been proved in several real-world applications [Pic+10] [SH13].

Differential equations are mathematical equations for an unknown function of one or several variables that relates the values of the function itself and its derivatives of various orders: they play a prominent role in engineering, physics, economics, biology, and other disciplines.

The idea of using symbolic regression to learn differential equations is present since the beginnings of GP [Koz92]: given the great interest towards this topic, several research lines have followed. Babovic and Keijzer [Bab+01] propose a dimensionally-aware GP to learn dynamic systems in hydraulic engineering. Cao et al. [Cao+00] present a GP-based technique where an individual is a set of trees, representing a system of equations. Coefficients of the equations are optimized via a Genetic Algorithm, then the system is solved through a numerical integration method and the resulting equations are finally evaluated against training data. Iba [Iba08] proposes an improvement over the previous approach, where coefficients are optimized through a least mean square technique, and a Runge-Kutta method of 4th order is used to build a solution. Bernardino and Barbos [BB11] use Grammar-Based Immune Programming to tackle the problem. It is important to notice that, while quite effective, all these concepts rely upon the use of ad-hoc individual construction, and significant computational costs to first solve the candidate equations and then compare them to experimental data.

A novel methodology for learning ordinary differential equations (ODE) through symbolic regression is proposed, whose original idea stems from an invited talk given by Maarten Keijzer during the GECCO conference in 2013 [Kei13]. Given a system of ODEs, the problem can be reduced to finding the first-order approximation of each ODE. The approach includes the subsequent steps:

1. For each equation, standard symbolic regression is used to obtain a small group of candidate solutions that represent a trade-off between complexity and fitting;
2. A simple derivation procedure, following the properties of the first-order approximation of an ODE, is applied to each candidate solution, transforming them in ODEs;
3. Finally, corresponding equations are coupled in systems and examined with respect to dynamical behavior and fitting on the original data. The best system is returned to the user as the solution for the original problem.

Important advantages of the proposed method are the possibility of learning differential equations using established symbolic regression techniques, instead of devising ad-hoc individual representations and fitness functions; the greatly reduced computational cost, since the most expensive procedures are performed *a posteriori* on a reduced set of candidate solutions; and the possibility of separately learning each differential equation in a target system, since the first-order approximation removes dependencies between variables.

Using the Lotka-Volterra model as a case study, the applicability of the proposed methodology is proved through experimental validation. The described approach is able to regularly find the correct structure of the original model, even in presence of noise. Results are discussed, and future works outlined.

The rest of the section is structured as follows: subection 3.3.2 recalls a few necessary concepts related to symbolic regression and differential equations. The proposed approach is outlined in subsection 3.3.3. The case study is presented in subsection 3.3.4, while the experimental evaluation is described in subsection 3.3.5. Results are discussed in subsection 3.3.6, and finally subsection 3.3.7 draws the conclusions and prospects future works.

3.3.2 Background

Genetic Programming and symbolic regression

Symbolic regression is an evolutionary technique able to extract free-form equations that correlate data from a given experimental dataset. The original idea is presented in [Koz92]. Candidate solutions are encoded as trees, with terminal nodes corresponding to constants and variables of the problem, while intermediate nodes encode mathematical functions such as $\{+,-,*,/,...\}$. The fitness function is usually proportional to the absolute or squared error between experimental data, with parsimony corrections to favor more compact solutions. An example of an individual for a symbolic regression problem is presented in Figure 3.1.

Differential equations and first-order approximation

In order to clarify the scope of this work, a few basic concepts related to differential equations are summarized in the following. A differential equation is defined as *an equation containing the derivatives of one or more dependent variables, with respect to one of more independent variables* [Zil08]. The focus of this work is on *ordinary differential equations* (ODEs), that contain derivatives as a function of a single variable (e.g. the time). A classical example of a differential equation is the first-order ordinary differential equation :

$$y'(t) = f(t, y(t)) \qquad y(t_0) = y_0 \tag{3.14}$$

where $y(t)$ is a function and y_0 is an initial condition.

The *(Explicit) Euler method* [Eul68] is a first-order numerical procedure for solving ordinary differential equations with a given initial value: it is the most basic explicit method for numerical integration of ordinary differential equations. With reference to Equation 3.14, the finite difference formula can be used to approximate $y'(t)$:

$$y'(t_n) = \lim_{\Delta_t \to 0} \frac{y(t_n + \Delta t) - y(t_n)}{\Delta_t} \cong \frac{y(t_n + \Delta t) - y(t_n)}{\Delta_t} \tag{3.15}$$

Choosing a value Δt for the size of every step and setting $t_n = t_0 + n \cdot \Delta t$, one step of the Euler method from t_n to $t_{n+\Delta t} = t_n + \Delta t$ is:

$$y_{n+\Delta t} = y_n + \Delta t \cdot f(t_n, y_n) \tag{3.16}$$

where the value of y_n is an approximation of the solution to the ODE at time t_n, so that $y_n \approx y(t_n)$. The error per step of this method is proportional to the square of the step size, while its error at a given time is proportional to the step size. It is important to notice how the selection of the step size plays a crucial role in the quality of the results.

A remarkable property of the Euler approximation is the possibility of reconstructing the initial ODE, under specific conditions. In particular, one can rewrite Equation 3.16 as follows:

$$y_{n+\Delta t} - y_n = F(t_n, y_n, \Delta_t) \tag{3.17}$$

where F is a function which allows to evaluate $y_{n+\Delta t}$ for any value Δ_t. From Equation 3.17 and looking at the derivative according to Δ_t around 0, the result is

$$\lim_{\Delta t \to 0} \frac{y_{n+\Delta t} - y_n}{\Delta_t} = \lim_{\Delta t \to 0} \frac{F(t_n, y_n, \Delta_t) - F(t_n, y_n, 0)}{\Delta_t} \tag{3.18}$$

which can be rewritten as

$$f(t, y(t)) = y'(t) = \left. \frac{\partial F(t_n, y_n, \Delta_t)}{\partial \Delta_t} \right|_{\Delta_t = 0} \tag{3.19}$$

going back to Equation 3.14.

In a practical scenario, Equation 3.17 can be used to iteratively build the approximate solution of Equation 3.14. At the opposite, assuming that an analytical form of the approximate solution of Equation 3.14 is available, Equation 3.19 can be used to obtain function f.

3.3.3 Proposed approach

Using Equation 3.19, it is possible to return to the original ODE starting from the first-order approximation given in Equation 3.17. It is sufficient to find the classical function F in Equation 3.17.

In order to find F, additional data must be computed. Given a standard dataset with values of y for different values of time t, extra information needs to be added to each line y_n, t_n, by computing the values of Δt and $y_n + 1$: in fact, in a real-world dataset, it is not given that $\Delta t = t_{n+1} - t_n$ will be constant for every n. Nevertheless, the procedure is trivial: an example is reported in Table 3.8. Once the new data are obtained, symbolic regression can be straightforwardly applied to the new dataset, to learn F.

t	y
0	20
1.8	16.1
3.5	13.2
5.4	10.9
7.4	8.8
...	...

$\Longrightarrow$

t	y	Δt	$F = y_{n+\Delta t} - y_n$
0	20	0	0
0	20	1.8	-3.9
1.8	16.1	0	0
1.8	16.1	1.7	-2.9
3.5	13.2	0	0
3.5	13.2	1.9	-2.3
5.4	10.9	0	0
5.4	10.9	2.0	-2.1
...	...	...	...

Table 3.8: An example on how the values of the additional variables (**right**) can be easily produced starting from the original dataset (**left**). In this case, for each line, computed the values of Δt and F to the next point, only, are computed.

One of the known issues of symbolic regression and GP in general is the so-called *overfitting*: solutions that closely approximate training data often exploit exclusive features of the dataset, for example by including terms that model the noise as well. This leads to poor performances on validation sets. Overfitting is sometimes associated with *bloating*, that is, the tendency of GP algorithms to produce bigger and bigger solutions as the evolution goes on. Connections between overfitting and bloating are still being investigated [VCS10] [ONe+10], but empirical evidence shows

how it can be beneficial to add parsimony measurements in the fitness function or preserve solutions of different complexity, in order to contain the phenomenon.

While overfitting is always undesired, it is particularly deleterious for the proposed approach: even if the F found through symbolic regression performed reasonably well on validation data, when using the proposed procedure to go back to the original ODE, terms with a limited influence on F could create degenerate solutions. For this reason, instead of just using the best solution obtained at the end of the process, it is preferable to have a set of candidate equations, each one a different compromise on a Pareto front between complexity and fitting on data.

Dynamic systems are usually represented by a set of ODEs and the proposed approach allows the user to run a symbolic regression algorithm independently on each equation: however, since it is preferable to work with a set of candidate solutions for each equation, an extra step is needed to choose the best combination to represent the original system. Thus, the procedure described in Equation 3.19 is applied to every candidate solution of each set; a set of n-uples, where n is the number of equations in the original system, is generated by permuting solutions in all sets; degenerate n-uples, showing a behavior dissimilar from the original data, are discarded; and finally the n-uple with the least absolute error with regards to the training data is selected. The whole procedure is summarized in Figure 3.17.

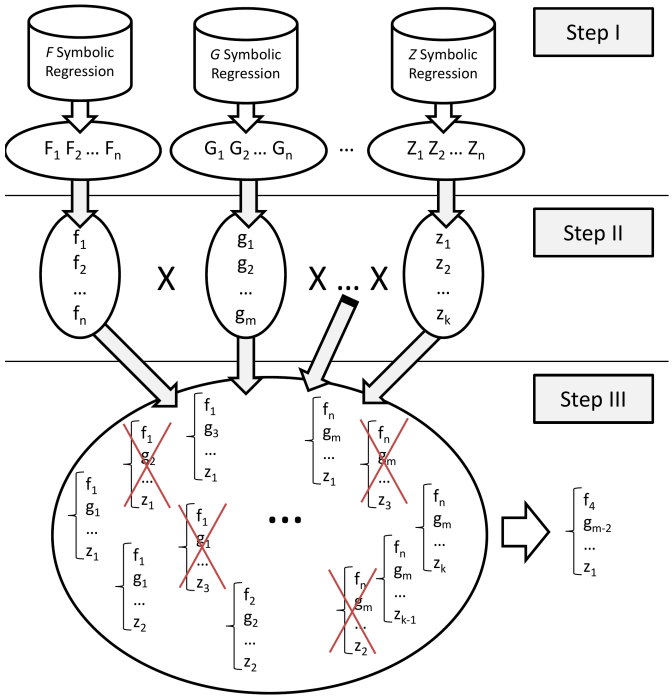


Figure 3.17: Summary of the proposed approach. In **Step I**, standard symbolic regression is executed independently on each equation of the original dynamic system: each run returns a set of candidate solutions of variable size, representing different compromises between complexity and fitting on training data. During **Step II**, the obtained sets are transformed into sets of ODEs, following the proposed methodology, and then permuted. Finally, in **Step III**, the resulting set of systems of ODEs is pruned of degenerate equations, the remaining candidate solutions are sorted by fitting on the original data, and the best solution is returned to the user.

3.3.4 Case study

In order to attest the viability of the proposed approach, the Lotka-Volterra model [Lot10] is selected as a case study. This model, also known as *predator-prey equations*, is a system composed of two first-order, non-linear, differential equations frequently used to describe the dynamics of biological systems in which two species interact, one as a predator and the other as prey. The equations have been extensively used in biology and other fields, such as economic theory [Goo67]. Their form is:

$$\begin{cases} \frac{dx}{dt} = x(\alpha - \beta y) \\ \frac{dy}{dt} = -y(\gamma - \delta x) \end{cases} \tag{3.20}$$

where x is the number of prey, y is the number of predators, t represents time, $\frac{dx}{dt}$ and $\frac{dy}{dt}$ represent the growth rates of the two populations over time. α, β, γ and δ are parameters that describe the interaction between the two species.

A particular configuration of the Lotka-Volterra model is selected, where the parameters' values have been chosen so that no population goes extinct, leading to periodic solutions: $\alpha = 0.04$, $\beta = 0.0005$, $\gamma = 0.2$ and $\delta = 0.004$. Initial populations were taken as $x_0 = y_0 = 20$. A plot of the chosen con

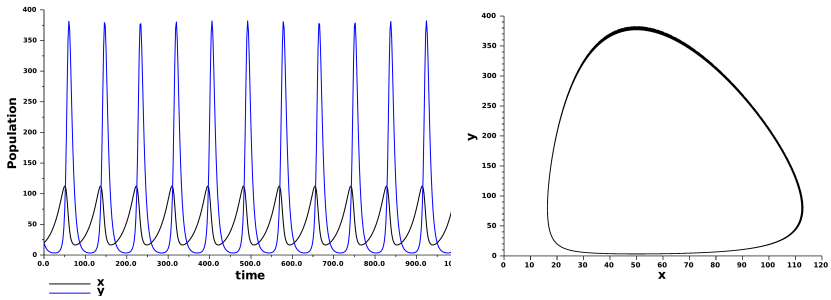


Figure 3.18: Plots of the Lotka-Volterra model with parameters used in the experiments. On the left, the variation of the two population with respect to time (x in black, y in blue/light grey). On the right, the state plane with x on the horizontal axis and y on the vertical axis.

Following Equation 3.17, the objective is then to find the two functions F and G, first-order approximations of the first and second differential equation of the Lotka-Volterra model, respectively:

$$x_{n+\Delta t} - x_n = F(\Delta t, x_n, y_n) \tag{3.21}$$
$$y_{n+\Delta t} - y_n = G(\Delta t, x_n, y_n) \tag{3.22}$$

A major feature of the proposed approach is the ability to learn the two functions in two separate and independent runs of the symbolic regression algorithm. Indeed, the reciprocal dependency of the Lotka-Volterra system has been removed.

3.3.5 Experimental results

Since one of the main advantages of the proposed approach is the possibility of exploiting existing tools for standard symbolic regression, for the present study the software *Eureqa Formulize*[9] [SL09],

[9] `http://formulize.nutonian.com/`

considered a state-of-the-art in the field, is selected. Eureqa has one feature of particular interest for the stated purpose: instead of returning a single solution per run, it presents the user a group of solutions that represent a Pareto front for the objectives of fitting and complexity: see Figure 3.20 for an example. In Eureqa, each symbol that can appear in a GP tree is associated with a weight, and the complexity of a candidate solution is simply the sum of all weights of terms appearing in it; fitting is computed with respect to the squared error with regards to the training data. It must be noted that, in principle, any GP-based technique able to preserve individuals of different complexity in the final population could be used for the proposed methodology.

Each dataset is modified following the procedure described in Section 3.3.3: 200 points are used for the training set. Since the interest is in exploring the influence of noise and regularity of sampling on the quality of the final results, for each experiment two datasets are considered: a first dataset sampled every 2 s, and a second one, where every point of data is sampled between 1.5 and 2.5 s from the previous one, with uniform probability.

Eureqa is configured to employ its *Basic* set of functions `{+,-,*,/,negation}` and terminal symbols `{integer constant, float constant, variable}`. In each experiment Eureqa is run once to stagnation, that is, until the index for the maturity of the population hits the threshold value of 90%. On the machine used for the experiments, a laptop with an Intel i5-2430M CPU (2 cores, 2 threads per core) at 2.40 GHz and 4 GB of RAM, running to stagnation takes 15-20 minutes, and around 10^{10} total fitness evaluations. After each run, Eureqa typically returns about 20 solutions on its Pareto front.

Noise-free data

In the simplest scenario, no noise is added to the datasets. The first run, with data regularly sampled, returns 20 candidate solutions for F and 20 candidate solutions for G. Each equation is transformed into an ODE, following the proposed approach. The resulting 400 systems are then pruned of degenerate solutions, that is, solutions that converge towards a point in the x,y plane (see Figure 3.19 for an example). The remaining systems of ODEs are finally sorted by fitting on the original unmodified training data. The same procedure is followed for the dataset with irregular sampling. This time, 21 candidate solutions are produced for F and 25 for G. The best ODE systems are:

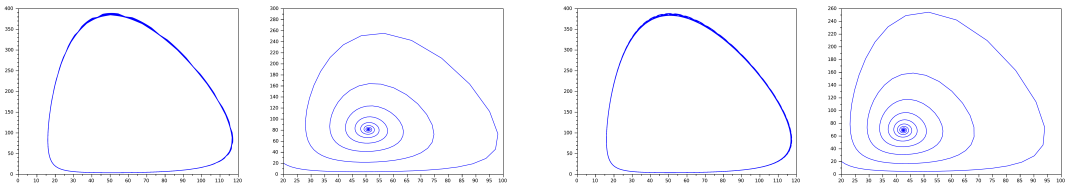
$$\begin{cases} \frac{dx}{dt} = 0.04114x - 0.0004946xy \\ \frac{dy}{dt} = 0.00367xy - 0.1861y \end{cases} \qquad \begin{cases} \frac{dx}{dt} = 0.04116x - 0.0004924xy \\ \frac{dy}{dt} = 0.003599xy - 0.1826y \end{cases} \tag{3.23}$$

with the result for regular sampling on the left, and the result for irregular sampling on the right. Both show the same form of the original Lotka-Volterra model, and a remarkable approximation of the parameters' values. As a comparison, in Figure 3.19 the two systems found with the proposed approach are compared to the systems obtained by simply coupling the best fitting-wise candidate solutions produced in each run.

Absolute noise

In a second trial, random noise (selected from the interval $(-5,5)$ with uniform probability) is added to the x and y outputs of the model. On the regularly sampled dataset, Eureqa finds 17 candidate solutions for F and 20 for G. On the irregularly sampled dataset, 13 solutions for F and 19 for G are obtained. The best resulting systems are:

$$\begin{cases} \frac{dx}{dt} = 0.03992x - 0.0005548xy \\ \frac{dy}{dt} = 0.003525xy - 0.1916y \end{cases} \qquad \begin{cases} \frac{dx}{dt} = 0.03946x - 0.0005354xy \\ \frac{dy}{dt} = 0.003662xy - 0.1948y \end{cases} \tag{3.24}$$



(a) Regular sampling, noise-free

(b) Irregular sampling, noise-free

Figure 3.19: Side-by-side comparison on the noise-free dataset, of the best system found through the proposed approach (**left**), and the system obtained by pairing the two fitting-wise best solutions of each run (**right**). It is easy to notice how simply pairing the best candidate solutions leads to degenerate forms or to a lowest fitting on the original training data.

with the result for regular sampling on the left, and the result for irregular sampling on the right.

Noise 5%

In the third experimental run, random noise proportional to the output value is added, ranging from -5% to +5% with uniform probability. On the regularly sampled dataset, Eureqa returns 16 candidate solutions for F and 15 for G. On the irregularly sampled dataset, 16 candidate solutions are obtained for F and 16 for G. The best resulting systems are:

$$\begin{cases} \frac{dx}{dt} = 0.03947x - 0.0004883xy \\ \frac{dy}{dt} = 0.003706xy - 0.1902y \end{cases} \quad \begin{cases} \frac{dx}{dt} = 0.03743x - 0.0004522xy \\ \frac{dy}{dt} = 0.003707xy - 0.1916y \end{cases} \tag{3.25}$$

with the result for regular sampling on the left, and the result for irregular sampling on the right.

Noise 10%

In the last experiment, random noise proportional to the output value is added, ranging from -10% to +10% with uniform probability. On the regularly sampled dataset, 23 candidate solutions for F and 20 for G are obtained. On the irregularly sampled dataset, Eureqa finds 17 candidate solutions for F and 18 for G. The best systems are:

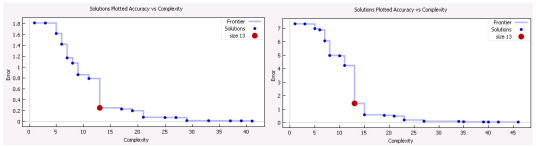
$$\begin{cases} \frac{dx}{dt} = 0.0362x - 0.0004797xy \\ \frac{dy}{dt} = 0.003306xy - 0.1841y \end{cases} \quad \begin{cases} \frac{dx}{dt} = 0.03874x - 0.0004959xy \\ \frac{dy}{dt} = 0.003587xy - 0.1898y \end{cases} \tag{3.26}$$

with the result for regular sampling on the left, and the result for irregular sampling on the right.

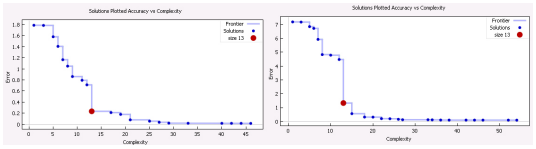
3.3.6 Results discussion

The proposed approach is able to find the correct model for the Lotka-Volterra function during each run, even if the parameters (α, β, γ, δ) might slightly differ, especially when dealing with noise. Remarkably, the irregularity of the sampling for the training set does not seem to influence the final outcome; while the presence of noise predictably returns results of lower quality.

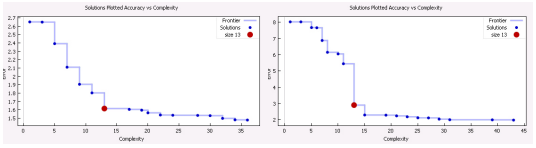
From the experimental evaluation, it is noticeable how Eureqa consistently returns a set of candidate solutions in the order of 10^1: since there are only two differential equations in the model, the search space for coupling the candidate solutions and assessing the results in the second step of the proposed process explores a search space of 10^2. However, when dealing with huge systems of



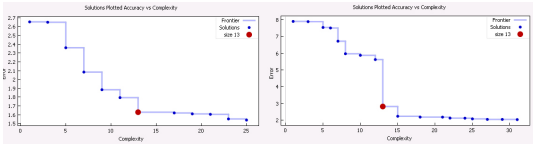
(a) Noise-free dataset, regular sampling: Pareto fronts for F (left) and G (right).



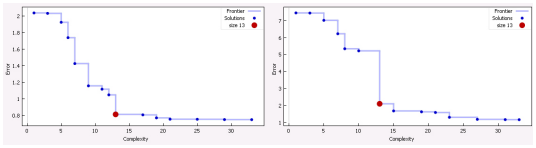
(b) Noise-free dataset, irregular sampling: Pareto fronts for F (left) and G (right).



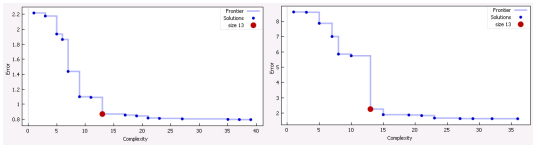
(c) Dataset with absolute noise, regular sampling: Pareto fronts for F (left) and G (right).



(d) Dataset with absolute noise, irregular sampling: Pareto fronts for F (left) and G (right).



(e) Dataset with 5% noise, regular sampling: Pareto fronts for F (left) and G (right).



(f) Dataset with 5% noise, irregular sampling: Pareto fronts for F (left) and G (right).

Figure 3.20: Pareto fronts of the solutions found by Eureqa during some of the experiments. The individual with the correct form of the Lotka-Volterra function is highlighted in red, and it is noticeable how it almost always lies in the middle of the Pareto front, often showing the biggest improvement over the previous step.

differential equations, the complexity quickly explodes: if the GP routinely returns n solutions, the search space of possible systems of m equations would become $O(n^m)$. Thus, it would be beneficial to reduce the number of viable equations in each set before the coupling process. For example, all equations that, after the derivation process from Equation 3.19, are reduced to a constant, can be dismissed. This subset, however, includes only 1-2 candidate solutions per set: other methods to prune the Pareto front from uninteresting models should be explored. From the experimental results, it is observable how most of the exact forms for the Lotka-Volterra equations always lie in the middle part of the Pareto front fitting/complexity provided by Eureqa (see Figure 3.20). It would be interesting to investigate whether this property can be generalized to all problems: in that case, the extremes of the Pareto front could be excluded; also, from the Pareto fronts, it looks that often the correct solution shows the biggest improvement with regards to the previous one. These considerations could be included in a heuristic coupling to reduce the number of associations.

3.3.7 Conclusions and future works

In this chapter, a GP-based methodology to learn ordinary differential equations starting from experimental data is proposed. The basic idea is reducing the problem to finding Euler's first-order approximation of an ODE, that is, a regular equation. Once the starting dataset is modified accordingly, a standard symbolic regression technique can be applied, obtaining a group of candidate solutions that represent a trade-off between complexity and fitting to data. Through an inverse procedure to reconstruct an ODE starting from its first-order approximation, used on the whole group of candidate solutions, a group of ODEs is acquired. Finally, by coupling the ODEs obtained, discarding degenerate solutions, and sorting the remaining ones by fitting on the training data, it is possible to find a system of ODEs that solves the initial problem. From the preliminary experiments,

it is clear that the coupling step might lead to a combinatorial explosion for the systems to evaluate. Future works will explore an automated coupling of candidate solutions, using theoretical and heuristic measurements to return the best set of solutions.

Acknowledgments

The authors would like to thank Luuk van Dijk of SpaceX for his interesting ideas and insightful discussions.

Bibliography

[Aff01] Michael Affenzeller. “A new approach to evolutionary computation: segregative genetic algorithms (SEGA)”. In: *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*. Springer, 2001, pages 594–601 (cited on page 89).

[AD08] Enrique Alba and Bernabé Dorronsoro. *Cellular genetic algorithms*. Volume 42. Springer, 2008 (cited on page 89).

[All92] Robin Allenson. *Genetic algorithms with gender for multi-function optimisation*. Technical report EPCC-SS92-01. Edinburgh Parallel Computing Centre, Edinburgh, Scotland, 1992 (cited on page 91).

[Bab+01] Vladan Babovic et al. “An evolutionary approach to knowledge induction: Genetic programming in hydraulic engineering”. In: *Proceedings of the World Water and Environmental Resources Congress*. Volume 111. 2001, pages 64–64 (cited on page 115).

[BR07] Khaled Badran and Peter I Rockett. “The roles of diversity preservation and mutation in preventing population collapse in multiobjective genetic programming”. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. ACM. 2007, pages 1551–1558 (cited on page 83).

[Bai+13] S. F. Bailey et al. “Competition both drives and impedes diversification in a model adaptive radiation”. In: *Proceedings of the Royal Society B: Biological Sciences* 280.1766 (July 2013), pages 20131253–20131253 (cited on page 82).

[BM12] Wolfgang Banzhaf and Barry McMullin. *Artificial Life*. Springer, 2012, pages 1805–1834 (cited on page 83).

[Ban+97] Wolfgang Banzhaf et al. “Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications (The Morgan Kaufmann Series in Artificial Intelligence)”. In: (1997) (cited on page 106).

[BBM93] David Beasley, David R Bull, and Ralph R Martin. “A sequential niche technique for multimodal function optimization”. In: *Evolutionary computation* 1.2 (1993), pages 101–125 (cited on page 95).

[BS02a] S Ben Hamida and Marc Schoenauer. “ASCHEA: new results using adaptive segregational constraint handling”. In: *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*. Volume 1. IEEE. 2002, pages 884–889 (cited on page 94).

[BB11] Heder S. Bernardino and Helio J.C. Barbosa. “Inferring Systems of Ordinary Differential Equations via Grammar-Based Immune Programming”. In: *Artificial Immune Systems*. Edited by Pietro Lio, Giuseppe Nicosia, and Thomas Stibor. Volume 6825. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pages 198–211. ISBN: 978-3-642-22370-9 (cited on page 115).

[BS02b] Hans-Georg Beyer and Hans-Paul Schwefel. “Evolution Strategies – A comprehensive introduction”. In: *Natural computing* 1.1 (2002), pages 3–52 (cited on page 83).

[BB91] “Symmetric Difference”. In: *The HarperCollins Dictionary of Mathematics*. Edited by E. J. Borowski and J. M. Borwein. HarperCollins, 1991 (cited on page 106).

[BB01] Markus Brameier and Wolfgang Banzhaf. “A comparison of linear genetic programming and neural networks in medical data mining”. In: *Evolutionary Computation, IEEE Transactions on* 5.1 (2001), pages 17–26 (cited on page 89).

[BB02] Markus Brameier and Wolfgang Banzhaf. *Explicit control of diversity and effective variation distance in linear genetic programming*. Springer, 2002 (cited on pages 87, 96).

[BB07] Markus Brameier and Wolfgang Banzhaf. *Linear Genetic Programming*. Volume 117. Springer, 2007 (cited on pages 85, 106).

[BGK04] Edmund K Burke, Steven Gustafson, and Graham Kendall. “Diversity in genetic programming: An analysis of measures and correlation with fitness”. In: *Evolutionary Computation, IEEE Transactions on* 8.1 (2004), pages 47–62 (cited on pages 83, 85, 86).

[Cao+00] Hongqing Cao et al. “Evolutionary modeling of systems of ordinary differential equations with genetic programming”. In: *Genetic Programming and Evolvable Machines* 1.4 (2000), pages 309–337 (cited on page 115).

[CNR07] Mario Castrogiovanni, Giuseppe Nicosia, and Rosario Rascunà. “Experimental analysis of the aging operator for static and dynamic optimisation problems”. In: *Knowledge-Based Intelligent Information and Engineering Systems*. Springer. 2007, pages 804–811 (cited on page 90).

[CLY09] Gang Chen, Chor Ping Low, and Zhonghua Yang. “Preserving and exploiting genetic diversity in evolutionary programming algorithms”. In: *Evolutionary Computation, IEEE Transactions on* 13.3 (2009), pages 661–673 (cited on page 83).

[CBM14] Leandro dos Santos Coelho, Teodoro Cardoso Bora, and Viviana Cocco Mariani. “Differential evolution based on truncated Lévy-type flights and population diversity measure to solve economic load dispatch problems”. In: *International Journal of Electrical Power & Energy Systems* 57 (2014), pages 178–188 (cited on page 104).

[CSS05] F. Corno, E. Sánchez, and G. Squillero. “Evolving assembly programs: how games help microprocessor validation”. In: *Evolutionary Computation, IEEE Transactions on* 9.6 (2005), pages 695–706 (cited on pages 103, 108).

[Cor+12] Guillaume Corriveau et al. “Review and Study of Genotypic Diversity Measures for Real-Coded Representations”. In: *IEEE Transactions on Evolutionary Computation* 16.5 (Oct. 2012), pages 695–710. ISSN: 1089-778X (cited on pages 86, 106).

[Dar59] Charles Darwin. *On the Origin of the Species by Means of Natural Selection: Or, The Preservation of Favoured Races in the Struggle for Life*. John Murray, 1859 (cited on page 82).

[Daw99] R. Dawkins. *The extended phenotype: the long reach of the gene*. Popular Science. Oxford University Press, 1999. ISBN: 9780192880512 (cited on page 84).

[DWP01] Edwin De Jong, Richard Watson, and Jordan Pollack. “Reducing bloat and promoting diversity using multi-objective methods”. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. L. Spector et al. Eds., July 2001 (cited on pages 86, 96, 103).

[De 75] Kenneth Alan De Jong. “Analysis of the behavior of a class of genetic adaptive systems”. PhD thesis. University of Michigan, Ann Arbor, 1975 (cited on page 94).

[Deb+02] K. Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *Evolutionary Computation, IEEE Transactions on* 6.2 (Apr. 2002), pages 182–197. ISSN: 1089-778X (cited on pages 98, 105).

[Deb89] Kalyanmoy Deb. “Genetic algorithms in multimodal function optimization”. PhD thesis. Clearinghouse for Genetic Algorithms, Department of Engineering Mechanics, University of Alabama, 1989 (cited on page 95).

[Deb05] Kalyanmoy Deb. “Multi-Objective Optimization”. In: *Search Methodologies*. Edited by Edmund K. Burke and Graham Kendall. Springer US, 2005, pages 273–316. ISBN: 978-0-387-28356-2 (cited on page 98).

[DG89] Kalyanmoy Deb and David E. Goldberg. “An investigation of niche and species formation in genetic function optimization”. In: *Proceedings of the 3rd International Conference on Genetic Algorithms*. Morgan Kaufmann Publishers Inc. 1989, pages 42–50 (cited on pages 92, 94).

[DJ13a] Kalyanmoy Deb and Himanshu Jain. “An Evolutionary Many-Objective Optimization Algorithm Using Reference-point Based Non-dominated Sorting Approach, Part I: Solving Problems with Box Constraints”. In: *IEEE Transactions on Evolutionary Computation* (2013). Early Access (cited on pages 95, 98).

[DJ13b] Kalyanmoy Deb and Himanshu Jain. “An Evolutionary Many-Objective Optimization Algorithm Using Reference-point Based Non-dominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach”. In: *IEEE Transactions on Evolutionary Computation* (2013). Early Access (cited on pages 95, 98).

[ES10] Agosten E Eiben and James E Smith. *Introduction to evolutionary computing*. Volume 2. Springer Berlin, 2010 (cited on pages 84, 104).

[EN01] Anikó Ekárt and Sandor Z. Nemeth. “Selection based on the pareto nondomination criterion for controlling code growth in genetic programming”. In: *Genetic Programming and Evolvable Machines* 2.1 (2001), pages 61–73 (cited on page 103).

[EN00] Anikó Ekárt and Sandor Z Németh. “A metric for genetic programs and fitness sharing”. In: *Genetic Programming*. Springer, 2000, pages 259–270 (cited on page 87).

[Eul68] Leonhard Euler. *Institutionum calculi integralis*. Volume 1. imp. Acad. imp. Saènt., 1768 (cited on page 116).

[Fog98] David B Fogel. *Evolutionary computation: the fossil record*. Wiley-IEEE Press, 1998 (cited on page 83).

[Fog62] L. J. Fogel. "Autonomous Automata". In: *Industrial Research* 4 (1962), pages 14–19 (cited on page 83).

[Fri58] Richard M Friedberg. "A learning machine: Part I". In: *IBM Journal of Research and Development* 2.1 (1958), pages 2–13 (cited on page 106).

[Gau+14] Sébastien Gaucel et al. "Learning Dynamical Systems Using Standard Symbolic Regression". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2014, pages 25–36 (cited on page 81).

[GST13a] Marco Gaudesi, Giovanni Squillero, and Alberto Tonda. "An efficient distance metric for linear genetic programming". In: *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference - GECCO '13*. ACM Press, 2013 (cited on page 81).

[GST13b] Marco Gaudesi, Giovanni Squillero, and Alberto Tonda. "An efficient distance metric for linear genetic programming". In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. ACM. 2013, pages 925–932 (cited on page 87).

[GST14] Marco Gaudesi, Giovanni Squillero, and Alberto Tonda. "Universal information distance for genetic programming". In: *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion - GECCO Comp '14*. ACM Press, 2014 (cited on page 81).

[GLR03] Kai Song Goh, Andrew Lim, and Brian Rodrigues. "Sexual selection for genetic algorithms". In: *Artificial Intelligence Review* 19.2 (2003), pages 123–152 (cited on page 91).

[Gol89] D.E. Goldberg. "Genetic algorithms in search, optimization, and machine learning". In: (1989) (cited on page 107).

[GR87] David E. Goldberg and J. Richardson. "Genetic algorithms with sharing for multimodal function optimization". In: *Proceedings of the 2nd International Conference on Genetic Algorithms*. 1987, pages 41–49 (cited on page 92).

[Goo67] Richard M Goodwin. "A growth cycle". In: *Socialism, capitalism and economic growth* (1967), pages 54–58 (cited on page 119).

[GFC99] GW Greewood, Gary B Fogel, and Manuel Ciobanu. "Emphasizing extinction in evolutionary programming". In: *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*. Volume 1. IEEE. 1999 (cited on page 98).

[Gre92] John J Grefenstette. "Genetic algorithms for changing environments". In: *Parallel Problem Solving from Nature (PPSN)*. Volume 2. 1992, pages 137–144 (cited on page 97).

[HL92] Prabhat Hajela and C-Y Lin. "Genetic search strategies in multicriterion optimal design". In: *Structural optimization* 4.2 (1992), pages 99–107 (cited on page 99).

[Ham50] R.W. Hamming. “Error detecting and error correcting codes”. In: *Bell System technical journal* 29.2 (1950), pages 147–160 (cited on page 111).

[Han06] Nikolaus Hansen. “The CMA evolution strategy: a comparing review”. In: *Towards a new evolutionary computation*. Springer, 2006, pages 75–102 (cited on page 104).

[Har95] Georges R Harik. “Finding multimodal solutions using restricted tournament selection”. In: *Proceedings of the Sixth International Conference on Genetic Algorithms*. San Francisco, CA. 1995, pages 24–31 (cited on page 95).

[Hol75] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975 (cited on pages 82, 83, 92).

[Hu+05] Jianjun Hu et al. “The hierarchical fair competition (HFC) framework for sustainable evolutionary algorithms”. In: *Evolutionary computation* 13.2 (Jan. 2005), pages 241–77 (cited on page 98).

[Iba08] Hitoshi Iba. “Inference of differential equation models by genetic programming”. In: *Information Sciences* 178.23 (2008), pages 4453–4468 (cited on page 115).

[KW89] S.A. Kauffman and E.D. Weinberger. “The NK model of rugged fitness landscapes and its application to maturation of the immune response”. In: *Journal of theoretical biology* 141.2 (1989), pages 211–245 (cited on page 109).

[Kei13] M. Keijzer. “Inducing Differential / Flow Equations”. Invited talk to the GECCO conference. July 2013 (cited on page 115).

[Kei96] Maarten Keijzer. “Efficiently representing populations in genetic programming”. In: *Advances in genetic programming*. MIT Press. 1996, pages 259–278 (cited on page 86).

[KB95] R Keller and Wolfgang Banzhaf. *Explicit maintenance of genetic diversity on genospaces*. Technical report. University of Dortmund, 1995 (cited on page 97).

[Koz92] John R Koza. *Genetic Programming, On the Programming of Computers by Means of Natural Selection. A Bradford Book*. MIT Press, Cambridge, MA, USA, 1992 (cited on pages 83, 85, 86, 106, 115, 116).

[Lan98] William B Langdon. *Genetic programming and data structures: genetic programming+ data structures= automatic programming!* Volume 1. Springer, 1998 (cited on page 86).

[Lev66] Vladimir I Levenshtein. “Binary codes capable of correcting deletions, insertions and reversals”. In: *Soviet physics doklady*. Volume 10. 1966, page 707 (cited on pages 86, 112).

[LE97] Joanna Lis and AE Eiben. “A multi-sexual genetic algorithm for multiobjective optimization”. In: *Evolutionary Computation, 1997., IEEE International Conference on*. 1997, pages 59–64 (cited on page 92).

[Lot10] Alfred J Lotka. “Contribution to the theory of periodic reactions”. In: *The Journal of Physical Chemistry* 14.3 (1910), pages 271–274 (cited on page 119).

[Mah95] Samir W Mahfoud. “Niching methods for genetic algorithms”. PhD thesis. University of Illinois, Urbana-Champaign, 1995 (cited on page 91).

[Mau84] M.L. Mauldin. “Maintaining diversity in genetic search”. In: *Proceedings of the national conference on artificial intelligence (AAAI conference on artificial intelligence)*. Volume 247. 1984, page 250 (cited on page 105).

[May92] Ernst Mayr. “Darwin’s principle of divergence”. In: *Journal of the History of Biology* 25.3 (1992), pages 343–359 (cited on page 82).

[McK00] Robert I McKay. “Fitness Sharing in Genetic Programming.” In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. 2000, pages 435–442 (cited on page 93).

[MH99] Nicholas Freitag McPhee and Nicholas J Hopper. “Analysis of genetic diversity through population history”. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. Volume 2. Citeseer. 1999, pages 1112–1120 (cited on page 86).

[MG15] Juan Julián Merelo-Guervós and Pablo García-Sánchez. “Modeling browser-based distributed evolutionary computation systems”. In: *arXiv preprint arXiv:1503.06424* (2015) (cited on page 89).

[MHF+93] Melanie Mitchell, John H Holland, Stephanie Forrest, et al. “When will a genetic algorithm outperform hill climbing?” In: *NIPS*. 1993, pages 51–58 (cited on page 101).

[ONe+10] Michael O’Neill et al. “Open issues in genetic programming”. In: *Genetic Programming and Evolvable Machines* 11.3-4 (2010), pages 339–363 (cited on page 117).

[ORe97] U-M O’Reilly. “Using a distance metric on genetic programs to understand genetic operators”. In: *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*. Volume 5. IEEE. 1997, pages 4092–4097 (cited on page 87).

[OK09] Ryozo Ooka and Kazuhiko Komamura. “Optimal design method for building energy systems using genetic algorithms”. In: *Building and Environment* 44.7 (2009), pages 1538–1544 (cited on page 89).

[Pét96] Alan Pétrowski. “A clearing procedure as a niching method for genetic algorithms”. In: *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*. IEEE. 1996, pages 798–803 (cited on page 94).

[Pic+10] Christoph Pickardt et al. “Generating dispatching rules for semiconductor manufacturing to minimize weighted tardiness”. In: *Simulation Conference (WSC), Proceedings of the 2010 Winter*. IEEE. 2010, pages 2504–2515 (cited on page 115).

[Pol03] Riccardo Poli. “A simple but theoretically-motivated method to control bloat in genetic programming”. In: *Genetic Programming*. Springer, 2003, pages 204–217 (cited on pages 96, 103).

[PKB07] Riccardo Poli, James Kennedy, and Tim Blackwell. “Particle swarm optimization”. In: *Swarm intelligence* 1.1 (2007), pages 33–57 (cited on page 83).

[RA00] Jalel Rejeb and M AbuElhaij. “New gender genetic algorithm for solving graph partitioning problems”. In: *Circuits and Systems, 2000. Proceedings of the 43rd IEEE Midwest Symposium on*. Volume 1. IEEE. 2000, pages 444–446 (cited on page 92).

[Rob87] George G Robertson. “Parallel implementation of genetic algorithms in a classifier system”. In: *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*. L. Erlbaum Associates Inc. 1987, pages 140–147 (cited on page 89).

[Ros95] Justinian P Rosca. “Entropy-driven adaptive representation”. In: *Proceedings of the workshop on genetic programming: From theory to real-world applications*. Volume 9. Citeseer. 1995, pages 23–32 (cited on page 87).

[RB97] Christopher D Rosin and Richard K Belew. “New methods for competitive coevolution”. In: *Evolutionary Computation* 5.1 (1997), pages 1–29 (cited on page 107).

[SSS11a] Ernesto Sanchez, Massimiliano Schillaci, and Giovanni Squillero. *Evolutionary Optimization: the μGP toolkit*. Springer, 2011, pages 594–601 (cited on pages 90, 95, 98).

[SSS11b] Ernesto Sanchez, Massimiliano Schillaci, and Giovanni Squillero. *Evolutionary Optimization: the μGP toolkit*. Springer, 2011 (cited on pages 109, 112).

[SB03] J. Sánchez-Velazco and J.A. Bullinaria. “Sexual Selection with Competitive/Cooperative Operators for Genetic Algorithms”. In: *Neural Networks and Computational Intelligence*. 2003 (cited on page 92).

[SK98] Bruno Sareni and Laurent Krahenbuhl. “Fitness sharing and niching methods revisited”. In: *Evolutionary Computation, IEEE Transactions on* 2.3 (1998), pages 97–106 (cited on pages 92, 107).

[Sch85] J David Schaffer. “Multiple objective optimization with vector evaluated genetic algorithms”. In: *Proceedings of the 1st international Conference on Genetic Algorithms*. L. Erlbaum Associates Inc. 1985, pages 93–100 (cited on page 99).

[SL09] Michael Schmidt and Hod Lipson. “Distilling free-form natural laws from experimental data”. In: *science* 324.5923 (2009), pages 81–85 (cited on pages 102, 114, 119).

[SL08] Michael D Schmidt and Hod Lipson. “Coevolution of fitness predictors”. In: *Evolutionary Computation, IEEE Transactions on* 12.6 (2008), pages 736–749 (cited on page 104).

[SR98] Birgitt Schönfisch and André de Roos. “Synchronous and asynchronous updating in cellular automata”. In: *Cellular Automata: Research Towards Industry*. Springer, 1998, pages 42–46 (cited on page 90).

[Sha+49] C.E. Shannon et al. *The mathematical theory of communication*. Volume 117. University of Illinois press Urbana, 1949 (cited on page 107).

[Sha48] Claude E. Shannon. “A Mathematical Theory of Communication”. In: *Bell System Technical Journal* 27.3 (1948), 379?423 (cited on page 87).

[Shi97] Hisashi Shimodaira. “DCGA: A diversity control oriented genetic algorithm”. In: *Tools with Artificial Intelligence, 1997. Proceedings., Ninth IEEE International Conference on*. IEEE. 1997, pages 367–374 (cited on pages 84, 86).

[Shi01] Hisashi Shimodaira. “A diversity-control-oriented genetic algorithm (DCGA): Performance in function optimization”. In: *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*. Volume 1. IEEE. 2001, pages 44–51 (cited on page 83).

[SH13] Terence Soule and Robert B Heckendorn. “A Practical Platform for On-Line Genetic Programming for Robotics”. In: *Genetic Programming Theory and Practice X*. Springer, 2013, pages 15–29 (cited on page 115).

[Spe64] H. Spencer. *The Principles of Biology*. A system of synthetic philosophy. vol. II-[III] v. 1. Williams and Norgate, 1864 (cited on page 84).

[ST08] Giovanni Squillero and Alberto Tonda. "A novel methodology for diversity preservation in evolutionary algorithms". In: *Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation - GECCO '08*. ACM Press, 2008 (cited on pages 87, 95, 108).

[ST16] Giovanni Squillero and Alberto Tonda. "Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization". In: *Information Sciences* 329 (Feb. 2016), pages 782–799 (cited on page 81).

[SP97] Rainer Storn and Kenneth Price. "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces". In: *Journal of global optimization* 11.4 (1997), pages 341–359 (cited on page 85).

[Sue79] C.Y. Suen. "N-gram statistics for natural language understanding and text processing". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 2 (1979), pages 164–172 (cited on pages 106, 108).

[SB07] Richard Svanbäck and Daniel I Bolnick. "Intraspecific competition drives increased resource use diversity within a natural population". In: *Proceedings of the Royal Society B: Biological Sciences* 274.1611 (2007), pages 839–844 (cited on page 82).

[Tac94] Walter Alden Tackett. "Recombination, selection, and the genetic construction of computer programs". PhD thesis. University of Southern California, 1994 (cited on page 86).

[TY07] Renato Tinós and Shengxiang Yang. "A self-organizing random immigrants genetic algorithm for dynamic optimization problems". In: *Genetic Programming and Evolvable Machines* 8.3 (2007), pages 255–286 (cited on page 97).

[TB03] Andrea Toffolo and Ernesto Benini. "Genetic diversity as an objective in multi-objective evolutionary algorithms". In: *Evolutionary Computation* 11.2 (2003), pages 151–167 (cited on pages 97, 103).

[Tom05] Marco Tomassini. "Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time". In: *Natural Computing* (2005) (cited on page 89).

[Ton+12] Alberto Paolo Tonda et al. "Bayesian Network Structure Learning from Limited Datasets through Graph Evolution". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pages 254–265 (cited on page 91).

[Tur50] Alan M Turing. "Computing machinery and intelligence". In: *Mind* (1950), pages 433–460 (cited on page 83).

[Urs02] Rasmus K Ursem. "Diversity-guided evolutionary algorithms". In: *Parallel Problem Solving from Nature (PPSN)*. Springer, 2002, pages 462–471 (cited on page 83).

[VCS10] Leonardo Vanneschi, Mauro Castelli, and Sara Silva. "Measuring bloat, overfitting and functional complexity in genetic programming". In: *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM. 2010, pages 877–884 (cited on page 117).

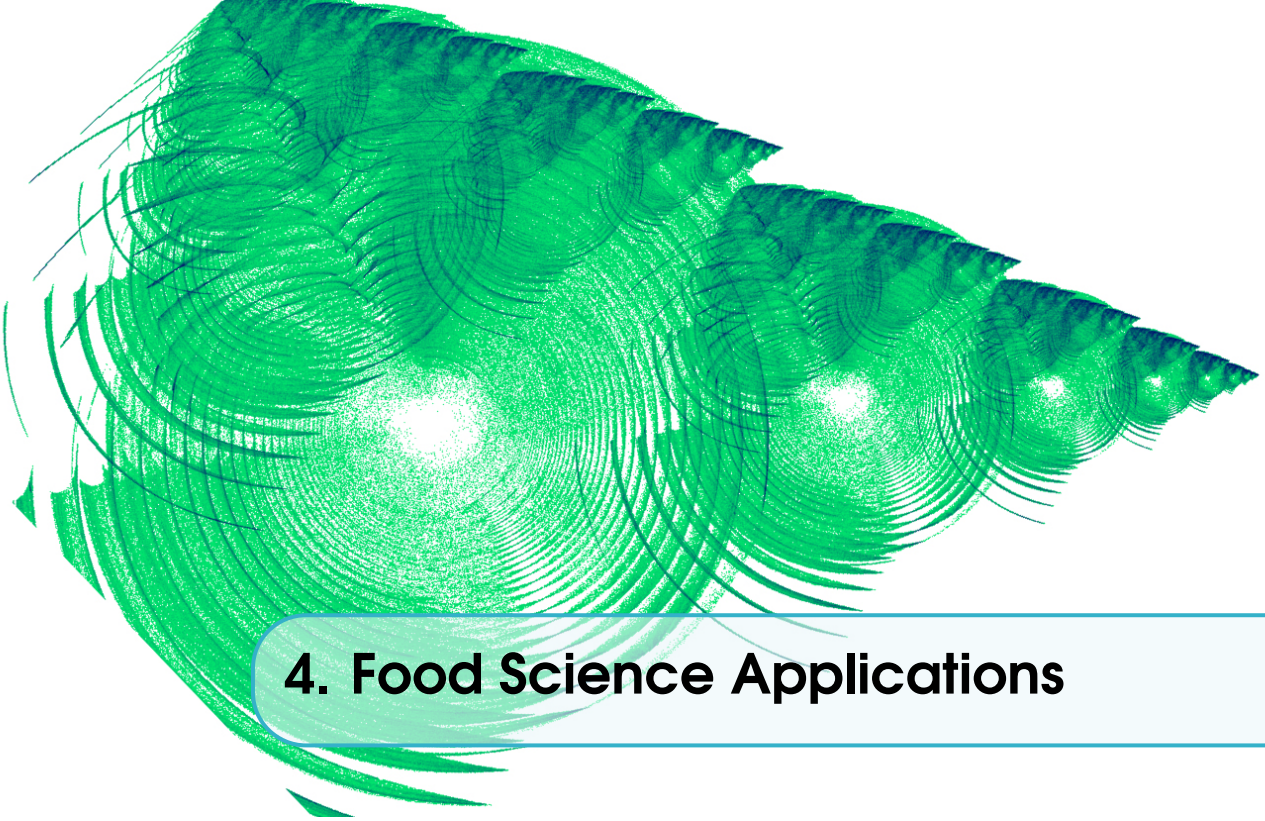
[Wei96] E.D. Weinberger. "NP completeness of Kauffman nk model, a tuneably rugged fitness landscape". In: *Santa Fe Institute Technical Reports* (1996) (cited on page 110).

[WRH99] Darrell Whitley, Soraya Rana, and Robert B. Heckendorn. “The island model genetic algorithm: On separability, population size and convergence”. In: *Journal of Computing and Information Technology* 7 (1999), pages 33–48 (cited on page 89).

[WO03] Mark Wineberg and Franz Oppacher. “The underlying similarity of diversity measures used in evolutionary computation”. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. Springer. 2003, pages 1493–1504 (cited on page 87).

[Zil08] Dennis G Zill. *A First Course in Differential Equations: With Modeling Applications*. Cengage Learning, 2008 (cited on page 116).

[ZT99] Eckart Zitzler and Lothar Thiele. “Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach”. In: *Evolutionary Computation, IEEE Transactions on* 3.4 (1999), pages 257–271 (cited on page 100).



4. Food Science Applications

Since I joined INRAE, the bulk of the applications I worked on were related to modeling and optimization of agri-food processes. Food transformation, in particular, has been the focus of my activities. This chapter provides a summary of the most important research lines I worked on, together with the description of a international resaerch project I chaired, focused on modelling in food science and industry.

A first part (Section 4.1) summarizes my work on interactive modelling of food processes, carried out resorting to machine learning methodologies and evolutionary optimization. The results illustrated there are the outcome of my co-supervision of Ph.D. students Etienne Deschamps and Thomas Chabin, AgroParisTech. A second part (Section 4.2) describes a stochastic model I developed for the action of the pepsin enzyme during digestion. A last part (Section 4.3) is devoted to a summary of COST Action CA15118 FoodMC, a European networking project on modeling of food processes that I led between 2016 and 2020.

4.1 Interactive Modelling of Food Processes

The apparent simplicity of food processes often hides complex systems, where physical, chemical and living organisms' processes co-exist and interact to create the final product. Data can be plagued by *uncertainty*; *heterogeneity* of available information is likely; *qualitative* and *quantitative* data may also coexist in the same process, from expert perception of food quality to nano-properties of ingredients. In order to obtain reliable models, it then becomes necessary to acquire additional information from external sources. Experts of a domain can provide invaluable insight in products and processes, but this precious knowledge is often available only in the form of intuition and implicit expertise. Including expert insight in a model can be tackled by having humans interacting with a machine learning process, through visualization or via specialists in encoding implicit domain knowledge. In this chapter, three selected case studies in food science portray different success stories of combining machine learning and expert interaction. This section will show how expert knowledge can be integrated at different stages of the modelling process, either online or offline, to initialize, enrich or guide this process.

4.1.1 Introduction

When dealing with meaningful representations of food systems, several important issues have to be considered: data can be plagued by *uncertainty*, particularly when chemical, physical, and biological phenomena concur to define the process; *heterogeneity* of available information is also likely, as a vegetable involved in a process can be characterized by more than 40,000 genes, whereas the quality of the final product can be assessed using just a few sensory features; *qualitative* and *quantitative* information, from expert perception of food quality, to nano-properties of ingredients, may also coexist in the same process. Consequently, when applying machine learning to agri-food data, the user has to carefully account for variance, manage heterogeneous data, and be able to include both qualitative and quantitative values in the final model.

As gathering data in food science is an expensive and time-consuming process, available datasets are often sparse and incomplete, which poses a challenge to both human modelling practitioners and machine learning algorithms. This issue has been long acknowledged by the community, and ongoing projects have been approved to tackle it, by defining roadmaps to achieve an e-infrastructure for open science [1], and by fostering cooperation between food scientists and modelling experts [2]. In order to obtain reliable models, it thus becomes necessary to acquire additional information from external sources. Experts in a specific domain can provide invaluable insight into products and processes, but this precious knowledge is often available only in the form of intuition and non-coded expertise. Including expert insight in a model is not a straightforward process, but it can effectively be tackled by having humans interacting with a machine learning process, through visualization, or via specialists in encoding implicit domain knowledge [LPT16].

In the following, three selected case studies portray different ways of combining machine learning with expert interaction, in the domain of food processing:

- first, a model for Camembert cheese ripening is built, encompassing variables from the micro-scale (presence of bacteria and chemical components) to the macro-scale (sensory evaluations), relying upon experts to help design the structure of a dynamic Bayesian network;
- a second dynamic Bayesian network model is constructed to help winemakers assess the appropriate time for harvesting grapes, depending on weather conditions
- a graphical model based on symbolic regression is used to help experts create a model of bacterial production and stabilization.

Interaction with the experts of each specific process is always mediated by visualization, complemented by the use of targeted questionnaires (first case study), fuzzy-logic models (second case study), or human-readable equations (third case study). In all considered cases, oriented graphs are used to provide experts with an intuitive and transparent representation of the model under construction. While the models' inner working, ranging from conditional probability inference to computation of free-form equations, is mostly hidden, users can easily interact with oriented graphs, where arcs represent correlation between variables, and modify connections created by learning algorithms, if they are deemed incorrect. For most users, graphs are familiar representations, and manipulating them is intuitive. When users are dealing with graphs that can be considered small, with less than 50 variables, node-link diagrams are a well suited portrayal, while matrices become more appropriate for larger or denser graphs [GFC].

[1] eRosa European project, `http://www.erosa.aginfra.eu/`

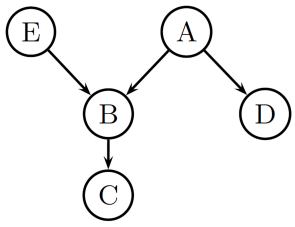
[2] COST Action CA15118 FoodMC, `http://www.inra.fr/foodmc`

4.1.2 Dynamic Bayesian network model for Camembert ripening

Cheese ripening is a good example of a process that human practitioners can achieve with success but for which several scientific details remain poorly understood. Nevertheless, even for these processes it is possible to create effective models by harnessing knowledge from experts in the domain and coupling it with experimental data. This can be achieved by using an appropriate machine learning framework, that is able to take into account such heterogeneous information. The work presented in [Sic+11] shows how the described methodology can be applied to the case of Camembert, a popular French cheese. The desired model goes from micro-scale properties such as concentration of lactose and bacteria, to macro-scale properties such as color and consistency of the crust, with the goal to describe the development of the ripening process, up to the prediction of the current phase of ripening. In Figure 4.1, a few pictures of the cheese ripening process are reported: experts find it useful to divide the ripening into 4 distinct phases.

Figure 4.1: Pictures of Camembert cheese during the ripening process. There are visible changes in the cheese's rind, color, and aroma during the ripening.

The approach used in this experiment is a dynamic Bayesian network (DBN) [Mur02], a variation on a classical Bayesian network [Pea14]. Bayesian networks are probabilistic models widely used to encode knowledge in several different fields: computational biology and bioinformatics (gene regulatory networks, protein structure, gene expression analysis), medicine, document classification, information retrieval, image processing, data fusion, decision support systems, engineering, gaming and law. BNs are directed acyclic graphs, where each node represents a variable in the problem, and links encode correlations between variables. An example of BN is reported in Figure 4.2.



Node	Parents	Probabilities
A		$P(A=a_1) = 0.99$ $P(A=a_2) = 0.01$
B	A,E	$P(B=b_1 \mid A=a_1, E=e_1) = 0.5$ $P(B=b_2 \mid A=a_1, E=e_1) = 0.5$ $P(B=b_1 \mid A=a_1, E=e_2) = 0.1$ $P(B=b_2 \mid A=a_1, E=e_2) = 0.9$ $P(B=b_1 \mid A=a_2, E=e_1) = 0.4$ $P(B=b_2 \mid A=a_2, E=e_1) = 0.6$ $P(B=b_1 \mid A=a_2, E=e_2) = 0.2$ $P(B=b_2 \mid A=a_2, E=e_2) = 0.8$

Node	Parents	Probabilities
C	B	$P(C=c_1 \mid B=b_1) = 0.3$ $P(C=c_2 \mid B=b_1) = 0.7$ $P(C=c_1 \mid B=b_2) = 0.5$ $P(C=c_2 \mid B=b_2) = 0.5$
D	A	$P(D=d_1 \mid A=a_1) = 0.8$ $P(D=d_2 \mid A=a_1) = 0.2$ $P(D=d_1 \mid A=a_2) = 0.7$ $P(D=d_2 \mid A=a_2) = 0.3$
E		$P(A=e_1) = 0.75$ $P(A=e_2) = 0.25$

Figure 4.2: On the left, a directed acyclic graph. On the right, the parameters it is associated with. Together they form a Bayesian network BN whose joint probability distribution is $P(BN) = P(A)P(B|A,E)P(C|B)P(D|A)P(E)$.

Like a BN, a DBN is a graph-based model of a joint multivariate probability distribution that captures properties of conditional independence between variables; in the graph, nodes $X_i(t), i = 1,...,N$, represent random variables, indexed by time t. Differently from a regular BN, a DBN is in fact able to encode dependencies between the same variable over multiple instants of time, providing a compact representation of the joint probability distribution P for a finite time interval $[1,\tau]$ defined as follows:

$$P(X(1),...,X(\tau)) = \prod_{i=1}^{N}\prod_{t=1}^{\tau} P(X_i(t)|Pa(X_i)(t)) \tag{4.1}$$

where $X(t) = X_1(t),...,X_N(t)$, is called a *slice*, and represents the set of all variables indexed by the same time t. $Pa(X_i)(t)$ denotes the parents of $Xi(t)$. $P(X_i(t)|Pa(X_i)(t))$ denotes the conditional probability function associated with the random variable $X_i(t)$ given $Pa(X_i)(t)$. The joint probability $P(X(1),...,X(\tau))$ represents the beliefs about possible trajectories of the dynamic process $X(t)$. DBNs are useful tools for combining expert knowledge with data at different levels and length scales. The structure of a model (e.g. the directed graph) can be explicitly built on the basis of expert knowledge, or automatically learned from data by an algorithm [CBL97]. In practice, a combination of the two approaches is commonly used, with a first, automatically-learned structure subsequently corrected by humans, resorting to graphical user interfaces such as BayesiaLab [3] or GeNie [Dru99][4]. Once the structure of a DBN is defined, parameters (i.e. conditional probability functions) can be automatically obtained without a priori knowledge on the basis of a dataset, all through a deterministic machine learning procedure known as *parameter learning*.

In this case study, data is gathered from 6 experiments on the cheese ripening process, each experiment lasting 41 days, with a sampling every day. The information obtained concerns the temperature of the ripening chamber (T, °C), relative humidity (RH, %), and the concentration of lactose (lo, g/kg), lactate (la, g/kg), and the bacteria *Kluyveromyces marxianus* (*Km*, cfu/kg), *Geotrichum candidum* (*Gc*, cfu/kg), *Penicillium camemberti* (*Pc*, cfu/kg), and *Brevibacterium aurantiacum* (*Ba*, cfu/kg). During each experiment, several Camemberts are destroyed to be analyzed, with a considerable economic investment for the producer. At the same time, experts are interviewed to provide additional information. The study involves two groups of experts: 4 cheesemakers with over 15 years of expertise in the industry, and 8 scientists with a track record of over 10 years of research on cheese processes. The questions posed to the experts are carefully constructed in order to elicit expert knowledge, with methods ranging from open-ended questions to focus groups. Values of the variables are discretized in 2 to 12 classes each, depending on expert judgment [Bau+10].

Following cheesemakers' considerations on the ripening process, the global model is divided into two parts, that are built independently and then linked: **M1** reproduces the temporal links between measured experimental data, simulating how such quantities vary during the ripening process; while **M2** is derived almost entirely from the expert knowledge gathered using questionnaires, and provides a more qualitative assessment between sensory information such as flavor, texture, color, and the ripening phase. Camambert cheesemakers traditionally identify four different phases in the ripening process. Figure 4.3 shows the final structure of the DBN obtained after the learning process. Variables between **M1** and **M2** are used to link variations in measurable quantities to sensory properties of the cheese.

[3] `http://www.bayesia.com`

[4] `https://www.bayesfusion.com/`

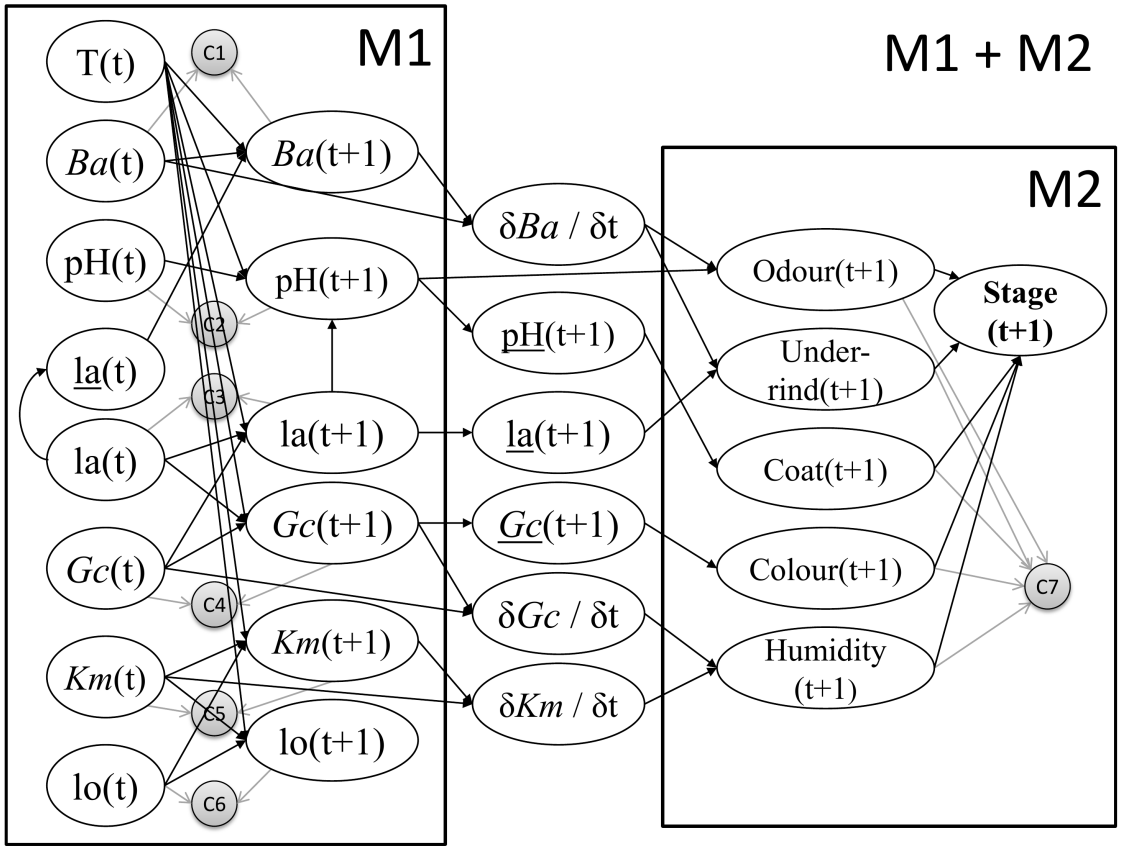


Figure 4.3: Final DBN model for the Camembert cheese ripening process. The part denominated M1 represents the variables taken mainly from experimental data, whereas part M2 represents variables derived from expert knowledge and assessment. Grey nodes represent constraints defined by experts. Figure redrawn from [Sic+11], with permission from Elsevier.

Figure 4.4 presents an example of predictions of the dynamics in the process. It is noticeable how the model is able to satisfyingly reproduce the dynamics of variables tied to microbial growth, substrate consumption, and sensory properties, for different temperature conditions. Experts ultimately assessed model simulations resorting to classical two-dimensional plots against test data, and were satisfied with the results.

4.1.3 Decision-support system for grape maturity prediction

Predicting the right moment to harvest grapes intended for wine production is a task that traditionally is left to specialists in the field. Still, as repercussions of climate change make local weather more unpredictable, experts can use machine learning techniques as a decision support tool, helping them to deal with modified conditions. Such decision support systems are commonly defined as interactive computer-based systems that help organizations in decision-making activities.

In viticulture, some decision support systems are already in use, for example to prevent mildew [Ray+10]. Grape berry maturity is analyzed in [Dai+09] where the authors built mechanistic models to predict the concentration of sugar in grapes. Other modelling techniques based on

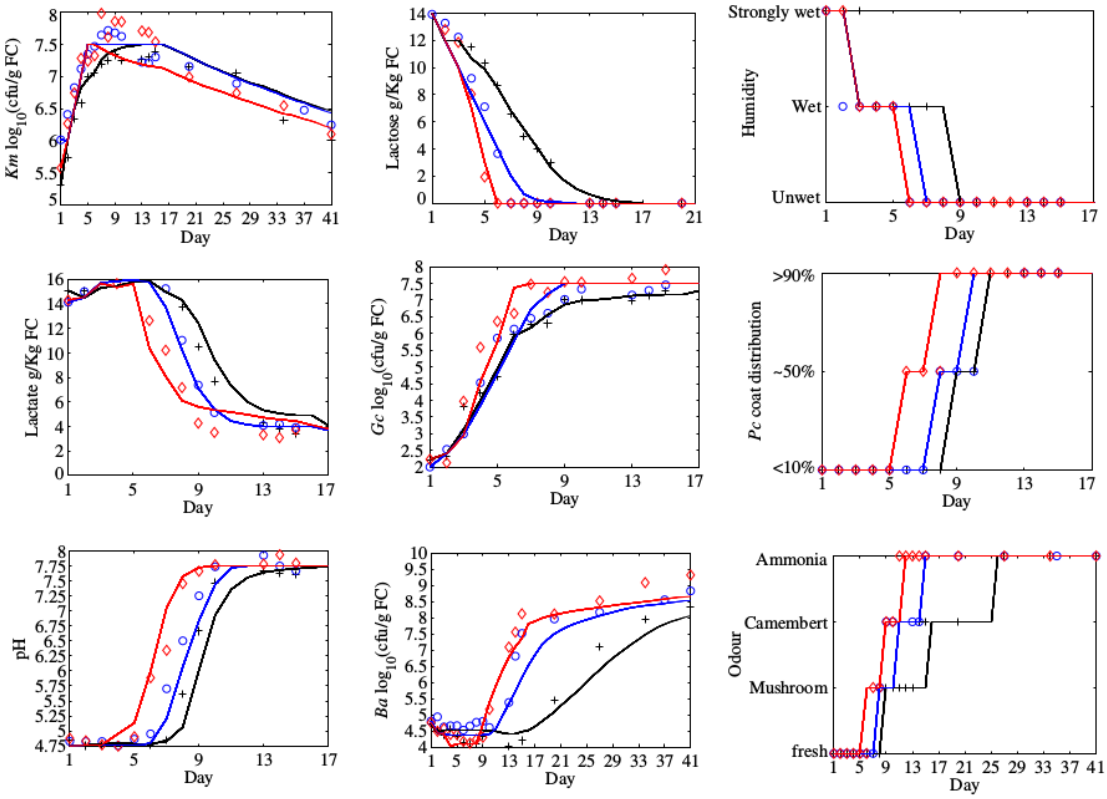


Figure 4.4: (color online) Predictions of the Camembert cheese ripening model for the evolutions of **(top row)** microbial growth (*Km*, *Gc*, *Ba* in decimal logarithm scale); **(middle row)** substrate consumption (*lo*, *la*) and **(bottom row)** sensory properties (*RH*, Pc coat and odor). The DBN model's prediction are represented as lines, versus raw data, represented as points, for three different ripening processes, carried out at 8 °C (marked with +), 12 °C (marked with ○) and 16 °C (marked with ◇). Figure reproduced from [Sic+11], with permission of Elsevier.

spectroscopy predict maturity indicators [FBR15]. These decisions support systems are based solely on experimental data, and do not integrate experts knowledge in order to predict grape maturity. As the human knowledge gained over years of wine production is invaluable and often includes conditions that have not been measured in recent times, it is only sensible to include it as much as possible in the target framework. Expert knowledge handling was already successfully used in the field of viticulture in [Cou+12]. Similar to the proposed approach, their model relies on fuzzy logic but to predict vine development with two indicators, vigor and precocity. In order to predict grape maturity, the innovative work presented in [Per+15] offers a good example of how human expertise can be employed to fill the gaps in experimental data, with the final objective of training a machine learning approach. This study represents the basis of the current work.

For this case study, data related to 66 parcels of land in the Loire Valley is collected over the course of 27 years (1988-2015), for a total of 1,086 data points describing weekly average temperature (T, °C), relative humidity (RH, %), insolation (Ins, hours of sunlight received per day, h/day) and rainfall (Pl, mm). Further data on sugar concentration (S, g/l) and acidity (Ac, g/l Eq H_2SO_4) of the grapes are collected every week, when 200 berries of Cabernet-Franc randomly

sampled from the parcels are crushed with a blender and subsequently analyzed. It is important to notice again how obtaining data is an expensive and time-consuming process, and it has to be integrated by expert knowledge, in order to improve the knowledge base eventually used for modelling. For this case study, human expertise is collected through a synthesis of the available literature and industrial reports, performed by 4 scientists and 5 winegrowers working in the areas considered in the study.

As for the previous case study, a Dynamic Bayesian Network proves particularly suited for this application, as such technique makes it possible to employ qualitative and quantitative variables, at different scales, in the same model. The network is designed with the help of the experts, through a trial-and-error process that includes several steps of structure visualization, correction, and analysis of the predictions, initially presented in [Bau+15]: the resulting structure is shown in Figure 4.7 (top). In this particular case study, even with an established structure, computing the parameters of each node is not trivial. Following experts' assessment, in fact, input is discretized into 8 to 15 classes for sugar, acidity, sugar variation, acidity variation, insolation, pluviometry, humidity and temperature. This discretization, featuring a relatively high number of classes when compared to more traditional applications of BNs, leads to conditional probability tables with a considerable amount of combinations: so many, that some of these combinations are not present in experimental data, and thus probabilities for these cases cannot be straightforwardly learned; resorting to experimental data for parameter learning, only, would leave too many gaps. A possible solution to the issue is to resort to experts again, formalizing their knowledge of the process through fuzzy logic mathematical functions.

Fuzzy logic [Zad65] is an extension of the binary logic, where a set is defined by its membership function. A value, x, belongs to a fuzzy set with a membership degree μ_L, with $0 \leq \mu_L(x) \leq 1$, see Figure 4.5. If define L as a set of *Low* insolation, the membership degree $\mu_L(x)$ of a given insolation value x can be defined as the level up to which insolation x should be considered as *Low*.

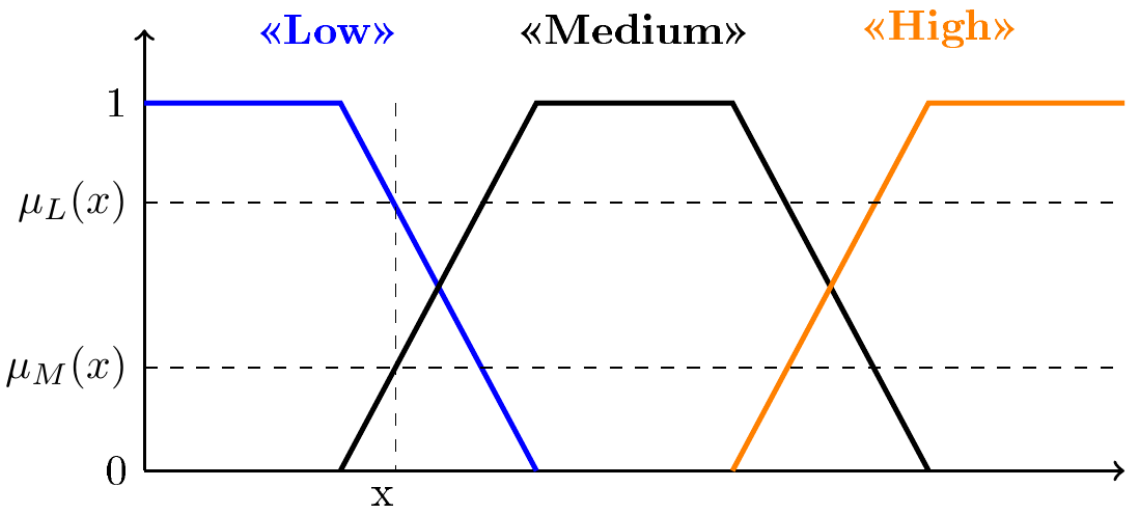


Figure 4.5: Example of three fuzzy sets *Low*, *Medium*, *High*, with $\mu_L(x)$: the membership degree in the *Low* fuzzy set and $\mu_M(x)$: the membership degree in the *Medium* fuzzy set.

Fuzzy sets for the four meteorological variables are then used to build 46 linguistic rules, e.g. *if insolation and pluviometry are Low, then the sugar increase is high*. Each rule is associated by

the experts to one of the four classes of meteorological condition, see Figure 4.6, and is activated according to the activation degree of each rule which define the class. Each class of meteorological condition corresponds to a certain variation of sugar and acidity for one day. The sum of variations on 7 days is performed to produce global variation over the week. This variation of sugar or acidity is added as an input to the DBN.

Figure 4.6: Definition of classes related to meteorological conditions defined into four classes for sugar and acidity concentration evolution expressed in g/L. Class index 0: Bad climatic conditions; Class index 1: Not favorable climatic conditions; Class index 2: Standard climatic conditions; Class index 3: exceptional climatic conditions.

The fuzzy logic model is created to produce data for combinations of input variables associated to equiprobability in the probability tables of the DBN; equiprobability, in turn, is associated to combination of input variables never observed in experimental data.

The complete structure of the framework, including the coupling fuzzy logic-DBN is shown in Figure 4.7. The first step (top) corresponds to the DBN learning based on experimental data. This step allows to produce probability table necessary to perform global predictions. However, some combinations of variable are absent from experimental data. For these specific cases, a probability table is updated using a fuzzy model (bottom). A simulated database is created in variable ranges of interest and variations of sugar and acidity can be produced. These data are included in parallel to experimental data and make it possible to define probabilities in any meteorological conditions necessary.

In order to evaluate the benefit of adding human expertise, the predictions were successively performed with the DBN model, the fuzzy model, and then with combined DBN-fuzzy models, see Figure 4.8. Best results are obtained by learning from both experimental data and expert knowledge. The resulting model is able to obtain satisfactory predictions, showing good R^2 values (a statistical measure of how close the data are to the fitted regression line) [SJ60] for both sugar content and

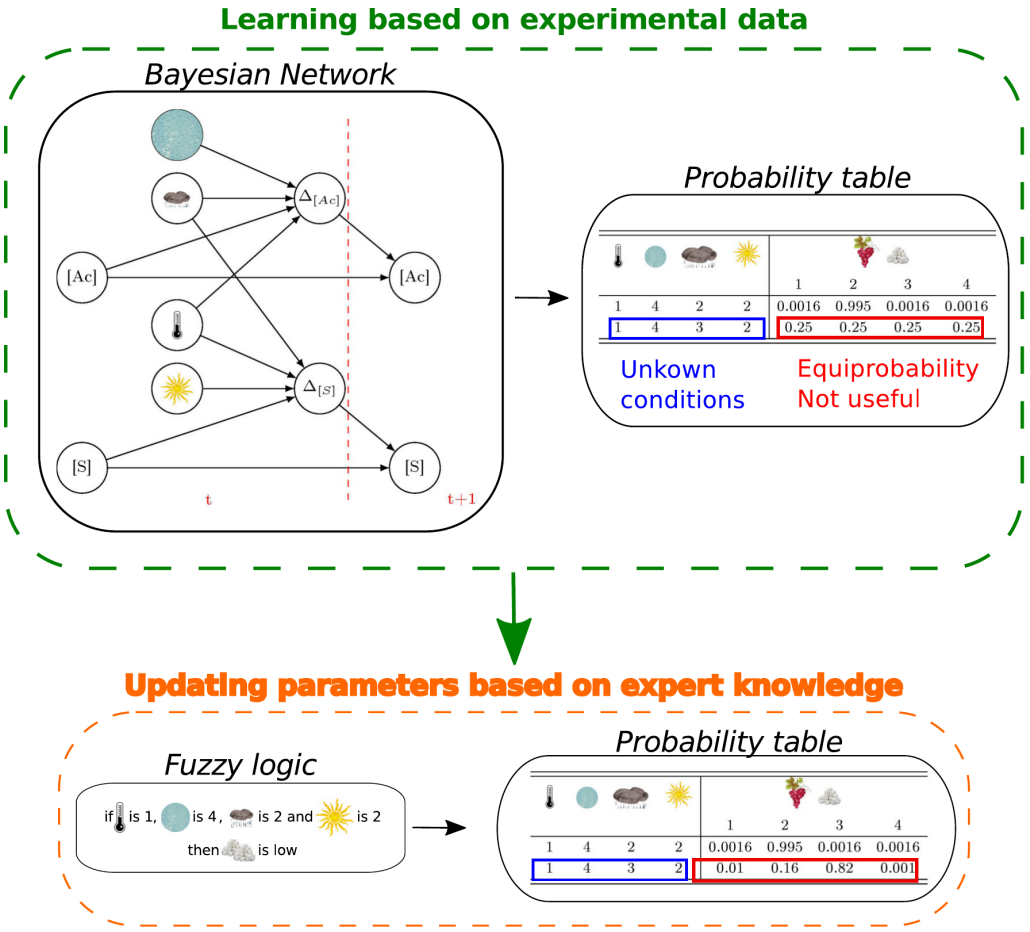


Figure 4.7: Proposed framework for the prediction of acidity and sugar content in grapes. (top) Structure of the DBN designed for the prediction of acidity (Ac) and sugar content (S) of grapes. (bottom) Parameters of DBN are updated with data produced by expert knowledge, making it possible to learn robust conditional probability tables for the nodes.

acidity, with $R^2_S = 0.85$ and $R^2_{Ac} = 0.83$, respectively. In comparison, the DBN model alone obtains $R^2_S = 0.80$ and $R^2_{Ac} = 0.74$ and the expert model alone obtains $R^2_S = 0.81$ and $R^2_{Ac} = 0.83$. Errors of predictions are shown in Figure 4.8. It is noticeable that at extremes values, the influence of the coupling DBN-Fuzzy approach is visible with significant improvement.

In the current context of climate change, exceptional meteorological conditions are expected to become more frequent. Learning process performed on experimental data of past years, only, risk to be unsatisfactory. The building of fuzzy models to integrate DBNs offers the possibility to enlarge the range of possible meteorological conditions and make the model more flexible and more robust.

4.1.4 Interactive symbolic regression modelling for bacterial production and stabilization

Concentrates of lactic acid bacteria are widely used in the food industry for products such as yogurt, cheese, fermented meat, vegetables and fruit beverages. The quality of bacterial starters, defined by the viability and acidification activity of the cells, depends on numerous control parameters across the different steps of the production and stabilization process, summarized in Figure 4.9 and

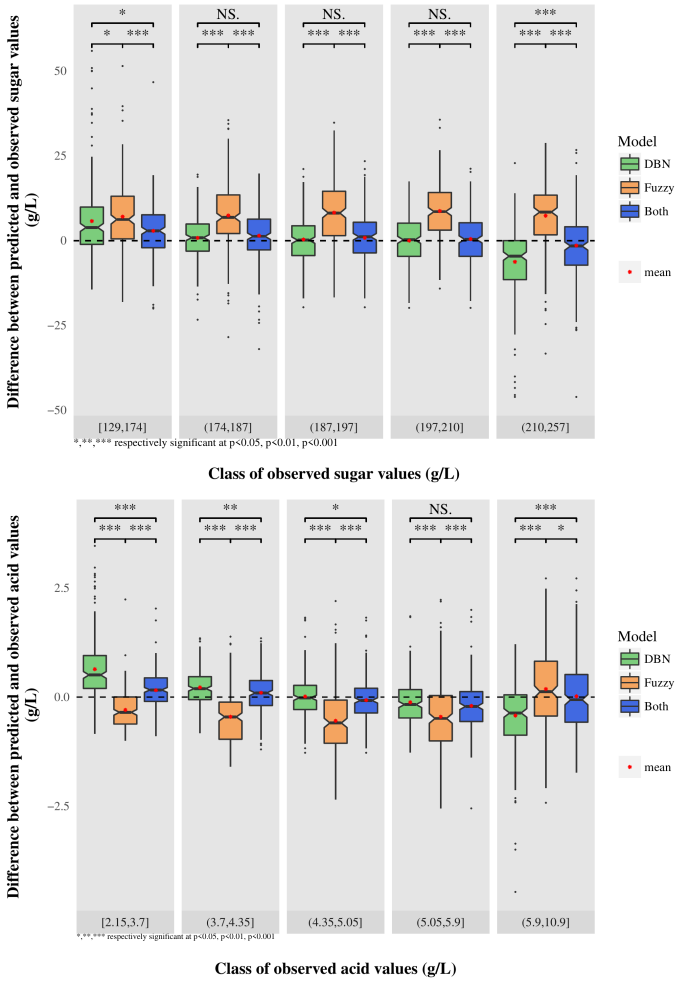


Figure 4.8: Prediction error according to class of values, for sugar (top) and acidity (bottom). For each class, the error is reported for the DBN model (green), the fuzzy model (orange) and combined model with DBN and fuzzy method (blue). The combined model clearly obtains the best results.

described in more details by Champagne and al. [Cha+91]. The bacterias levels of resistance to the processes is also dependent to the biochemical and biophysical properties and organization of their membrane [Vel+15; Vel+14] which in turn is determined by the genomic expression of the bacteria itself. For these reasons, modelling the bacteria resistance to the process is a complex problem due to many possible non-linear dependencies between the different length scales and steps of the process. In addition, no models are available for several sub-parts of the process, and even those that can be found in literature [Pas+11] are often too simple to be included in a wider framework.

One successful approach in modelling complex processes is to stack smaller models such that predictions are propagated between multiple layers formed by these sub-components [Cro+03; DVG07]. In such cases, typically, rich datasets and vast amounts of knowledge are available to describe the stacked components and their interactions. When little data is available, and prior knowledge is limited, mathematical regression techniques can be used to model these complex systems [Ped+11]. However, a multitude of candidate models can be obtained through these

techniques. Deciding which of these models is the best with respect to the study domain and problem at hand, may be carried out automatically based on a fitness criteria, or delegated to domain experts [KPB14]. While the former is efficient but can result in models that do not capture the reality of the underlying system, the former may be grounded albeit time-consuming. Similarly to Turkay et a. [Tur+17], the proposed approach uses mathematical regression to generate candidate solutions. However, the novelty is the combination of automatic evaluation of candidate models with expert evaluations to ensure both model robustness and validity.

The dataset in this case study concerns the full process of bacteria production and stabilization, with 49 variables measured at 4 different steps (fermentation, freezing, and storage) and at 4 different fermentation conditions (22 °C and 30 °C, with the fermentation stopped at the beginning of the stationary growth phase and 6 hours later). The variables consists of transcriptomics, composition of fatty acid membrane, acidification activity and viability [Vel+15]. Such a large number of variables requires peculiar methods to deal with. Using machine learning capacity to provide automatic modelling enable us to find possible dependencies.

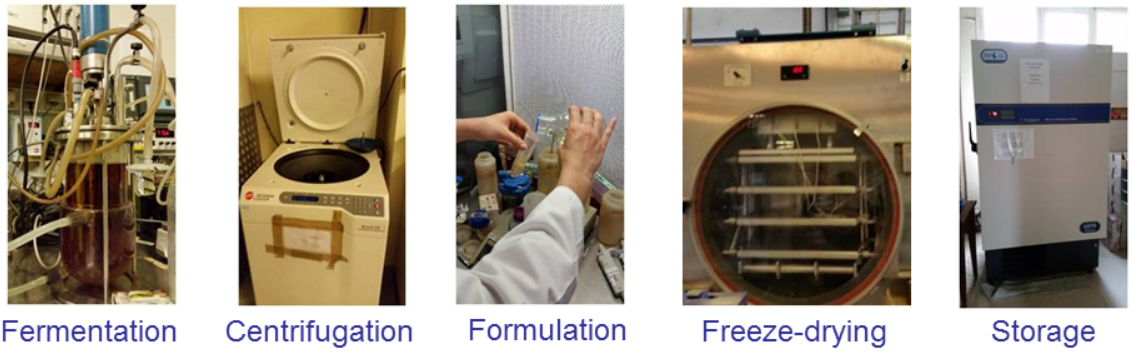


Figure 4.9: Steps of the freeze-drying process. Control parameters at every point in the process chain can influence the quality of bacterial starters.

From a vast number of possible dependencies between the measured variables, an automatic methodology can identify the most relevant ones, and combine them to obtain a global model. The main problem of this approach is that the number of variables is far superior to the number of samples in the dataset. The key idea is to remember that experts possess invaluable process knowledge that can considerably improve the robustness of the global model. While formalizing this often-implicit knowledge is not trivial, experts' insights can be effectively included in the modelling process by resorting to interactive approaches. To achieve these objectives, the software *LIDeOGraM* (Life science Interactive Development of Graph-based Models) is introduced, a semi-supervised model learning framework, based on regression analysis [Cha+17a; Cha+17b]. LIDeOGraM is able to obtain free-form equations for each variable in the process, as a function of all other variables. Each equation, describing a sub-part of the global process, can be considered a local model. Such models should fit the experimental data, and at the same time be deemed plausible by the experts. However, when using an automatic technique without expert guidelines, these two goals are often incompatible: it is always possible to find a polynomial equation that perfectly fits the data points, for example with a complex equation featuring as many parameters as data points available but such an equation could overfit the dataset, failing to represent the underlying relationship between the variables, and ultimately poorly predict the unseen data.

To avoid this issue, every variable in LIDeoGraM is associated with a set of candidate equations, obtained through symbolic regression [Koz92]. Eureqa[5] [SL09], a commercial software specialized in symbolic regression, is able to obtain a set of possible equations for every variable in a given dataset. A local model can thus be associated to each variable by selecting one of the equations in the set. Symbolic regression makes it possible to effectively search the vast space of all possible mathematical expressions, taking into account both the fitting of the equation and its complexity – indeed, more complex equations tend to be overfitted, while simpler ones are often unable to characterize the data. A collection of local models will then constitute the base for a global model, built using an evolutionary optimization algorithm [De 06] that stochastically searches the space of all sets of local models for the one that best fits the global dataset. To evaluate a candidate global model, the input nodes are set to known experimentally-measured values, and the errors in the prediction are averaged over all nodes, thus obtaining a global error, that the evolutionary algorithm aims to minimize.

Human experts are then involved in the modelling process, via a graphical user interface, showing a node-link graph visualization of the global model, where each node represents a variable, and each link marks a possible dependency between two variables. This interface allows experts to visualize the results from Eureqa, contribute with their knowledge, and finally lead the search for an efficient global model.

For this objective, two views are available. The **Local model view** shows an overall qualitative view of the equation sets given by Eureqa for each variable. This view enables nodes with no satisfactory equation in terms of fitting and/or complexity to be easily spotted. The **Global model view** shows the predictive capability of the current global model, for each variable. This view enables users to rapidly assess which variables in the global model are poorly predicted, but also which ones may be responsible for the poor predictions of their dependent nodes.

LIDeoGraM has several ways to add expert knowledge. First, it is possible to attribute a category to each variable, and specify the available dependencies between categories for the symbolic regression. A category of nodes can represent a step in the process, or a scale of information. This interface is presented in Figure 4.10.

After obtaining a set of equations for every node, experts can then filter it by specifying that certain kinds of node-to-node dependencies are not allowed. Experts can then manually add new equations in the set of candidate local models for a node, and eventually restart the search for a global model after putting all their constraints in place. With LIDeOGraM, it is possible to learn global models for the production and stabilization of bacteria. Such models can then be used to better understand how to preserve the quality of the culture during the process, foster the emergence of new hypotheses, and design new experiments, whose data could in turn be used to further improve the global model. These functionalities are demonstrated in Figure 4.11.

Results obtained for the previously described dataset [Vel+15] are presented in Figure 4.12.

In a preliminary experiment on the presented framework, a user with 20 years of experience on freeze-drying process is able to inject their knowledge into the optimization process. Out of a total of 232 equations generated for the local models, the expert deletes 5 equations, and 2 nodes, removing in turn 14 more equations in which the 2 deleted variables are involved. The expert then restarted symbolic regression on 3 nodes, obtaining 12 new equations. At the end of this process, the global optimization results are better than without the expertise, with the average error computed on all nodes being 0.801, using only the automatic approach, and 0.787 combining the automatic

[5] `http://nutonian.com/products/eureqa/`

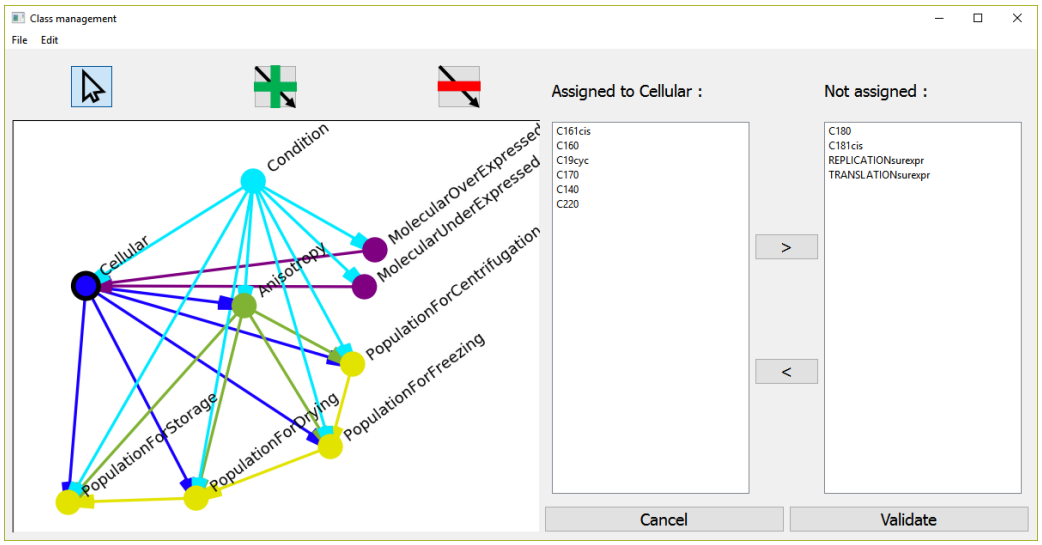


Figure 4.10: Screenshot of the interface allowing to choose the authorized links between the defined classes. A link between two classes means that all variables associated to the parent class can be used in the equations for all variables associated to the child class. The displayed graph represents the selected constraints chosen for the presented results.

approach with expert interaction. Figure 4.13 shows the evolution of the mean error per node, for both the automatic and the combined approaches. The results are still not completely satisfactory, as the prediction error for some of the nodes remains large, but the positive influence of the expert on the machine learning process is already substantial. In future works, more data points will be collected, and experiments with several other experts on the freeze-drying process are scheduled.

4.1.5 Discussion and guidelines

Computational Modelling is an iterative process that comprizes three main activities: *designing* a model where the aim is to define a suitable representation for objects and their relationships; *exploring* the model to understand its behavior, and *tuning* it to find the best or optimal parameter values to obtain good predictions. The proposed approach in building interactive machine learning systems for food science and technology focuses on involving experts of the process in one or more stages of this modelling pipeline, facilitating their interactions with the machine learning process through visual representations.

For the first two case studies, on modelling Camembert cheese ripening and grape maturity prediction, expert knowledge is integrated primarily at the design stage of model building. Using established methodologies from the knowledge elicitation domain (e.g. interviews, case studies, and observations), expert knowledge can be collected, coded and formalized into a probabilistic model. The goal, in these cases, is to create a knowledge representation of the process that matches the domain expert's mental model.

For the last case study, on modelling bacterial production and stabilization, experts knowledge is integrated at each stage of the modelling process. At the design stage, to structure the relationship of variables and system constrains prior to launching the automatic machine learning and optimization algorithms; and post-model learning through various user interactions via the LIDeOGraM interface.

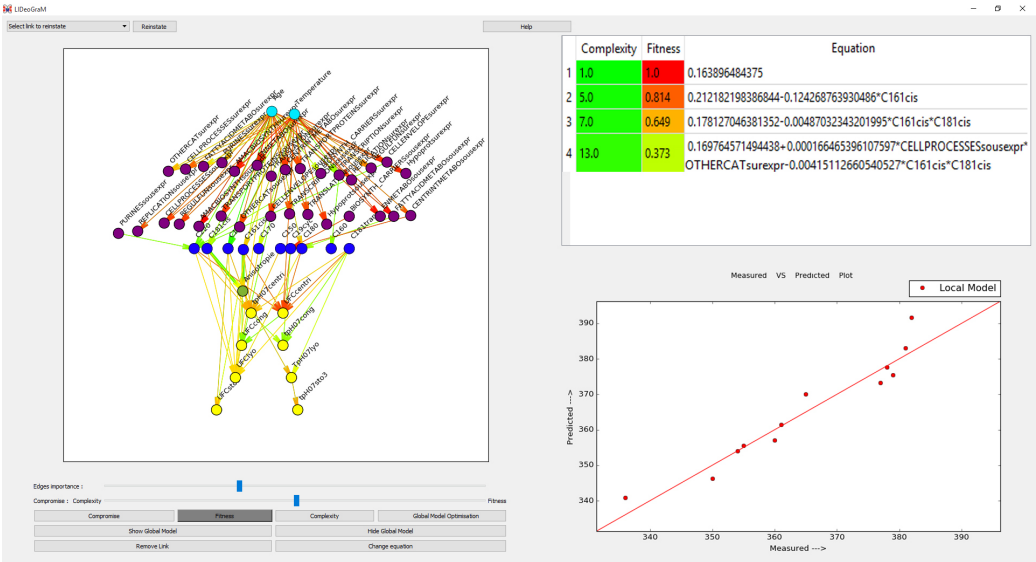


Figure 4.11: Screenshot of LIDeOGraM. The left side shows a graphical model representing the mean fitness of the local models obtained by symbolic regression. The top-right part is the list of equations proposed by Eureqa for the selected node, and the bottom-right part shows a plot of the measured versus predicted data associated to the selected equation.

For instance, domain experts can add or remove variables, classes and constraints. They could filter local models, or add new equations to explore how well they fit their data.

While interaction with experts is invaluable even with classical approaches, in the food science domain it is necessary to argue for a more user-centered design approach to machine learning, whereby users can participate at each stage of modelling process, from design to exploration and tuning. This involvement not only helps domain experts understand computational models better, but it allows them to confront their domain knowledge and know-how with with the results of machine learning, ultimately making machine learning more transparent. The authors' informal evaluations and discussions with domain experts allowed them to observe the following:

- providing visual representations of machine learning models improves user engagement and encourages feedback, especially if domain experts are involved at the design stage and exploration stages.
- graph-based model representations are easy to understand, but multiple linked representations are more helpful when trying to understand the model.
- experts tend to take a multi-step approach to model validation, first to verify existing knowledge (most likely to build trust in the ML algorithm), then to assess new predictions. When doing so, they first look at the general high-level dependencies between variables, before looking at detailed information such as values of weights, or data in the conditional probability tables when DBNs are involved.

It remains to prove whether making machine learning more transparent helps domain experts better explore and validate computational models in food science. More research is needed to study whether user-centered design for modelling improves decision making and helps indeed build trust in constructed models.

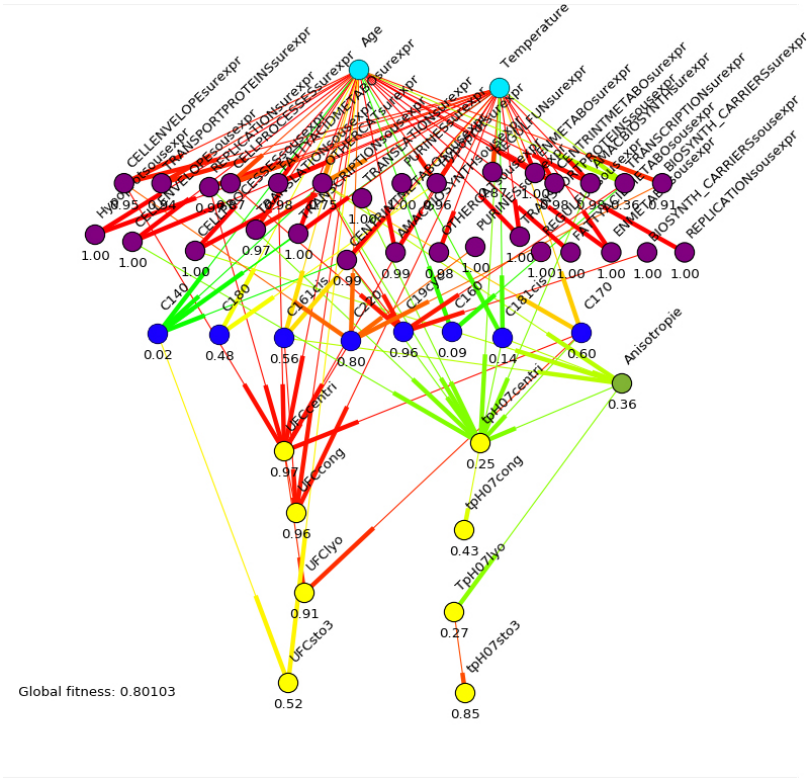


Figure 4.12: (color online) Graphical model generated in LIDeOGraM representing a (optimized) global model. Nodes are organized in 4 categories: experimental conditions, genomic scale, cellular scale, and population scale. A Pearson correlation coefficient, calculated using the predictions from the global model compared to the experimental measurements, is printed below each node. An edge between two nodes means that the parent variable is used in the equation chosen to calculate the child variable. The color of an edge depends on the Pearson correlation coefficient, which represent the quality of the prediction. The color varies from red for a poor-quality prediction to green for a satisfying one.

4.2 Modelling the Action of the Pepsin Enzyme During Digestion

This section presents a novel model of protein hydrolysis and release of peptides by endoproteases. It requires the amino-acid sequence of the protein substrate to run, and makes use of simple Monte-Carlo *in silico* simulations to qualitatively and quantitatively predict the peptides that are likely to be produced during the course of the proteolytic reaction. In the present study, the model is applied to the case of pepsin, the gastric protease. Unlike pancreatic proteases, pepsin has a low substrate specificity and therefore displays a stochastic behavior that is particularly challenging to model and predict. Two versions of the model are studied and compared with peptidomic data obtained during pepsin hydrolysis of bovine lactoferrin. The first version of the model takes into account cleavage probabilities according to the amino acids in position P1-P1' only, whereas the second version also accounts for the influence of neighbor amino acids (P4, P3, P2, P2', P3', P4') and peptide terminal ends. The second version of the model was able to reproduce many real-world features of the reported behavior of pepsin, such as the peptide size distribution, or the quantity of free amino-acids. More remarkably, 50% of the experimentally monitored peptides (44/87) lay

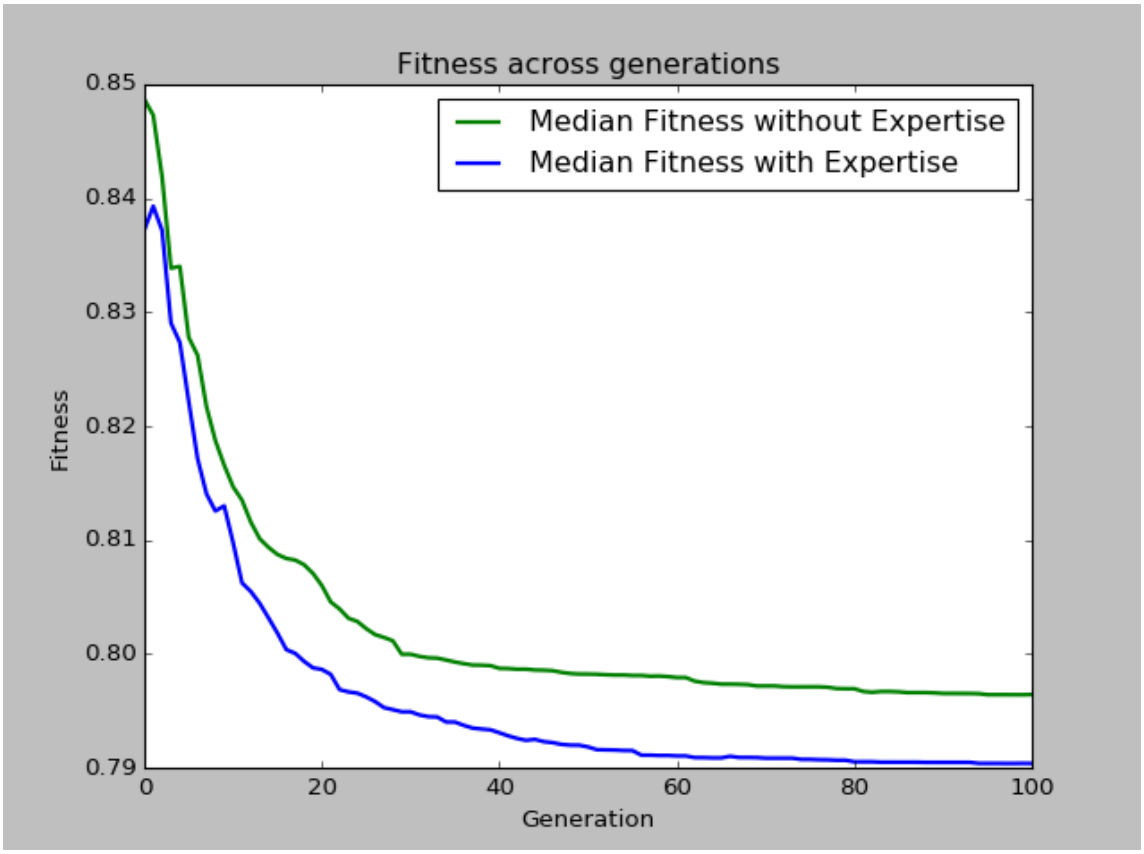


Figure 4.13: Comparison of an experiment on the learning of the freeze-drying model, using LIDeOGraM with and without human interaction. The term *fitness* here refers to the average error, computed on all variables in the problem. The *generations* are the iterations of the evolutionary algorithm used for optimizing the global model.

within the 120 most abundant simulated peptides. The presented methodology has the advantage of being applicable not only to different proteins, but to different enzymes as well, as long as cleavage frequency data are available.

4.2.1 Nomenclature

P**n**: amino acid in the **n-th** position to the left (i.e. N-terminal) of a target peptide bond.
P**n**': amino acid in the **n-th** position to the right (i.e. C-terminal) of a target peptide bond.
$\mathscr{P}(A)$: probability of event A.
$\mathscr{P}(A = a_1)$: probability of event A obtaining outcome a_1.
$\mathscr{P}(A|B)$: conditional probability of event A, given event B.
$\mathscr{P}(A = a_2|B = b_1, C = c_3)$: conditional probability of event A obtaining outcome a_2, knowing that event B obtained outcome b_1 and event C obtained outcome c_3.

4.2.2 Background

Gastric digestion, while integral to the interaction of animal bodies with foods, is far from being fully characterised. Various factors affect food digestion - some connected to the properties of the

food (e.g. buffering capacity, mechanical and structural properties) and others to the gastric process (e.g. endogenous secretions, shearing forces). [Che+11][Guo+15][FXS14] If a complete modeling of the fate of foods in the stomach is a very challenging area, several of these aspects have been modeled by *in silico* approaches. [Le +14][FS10] With the development of peptidomic approaches, an increasing number of studies are also dedicated to the identification of bioactive peptides during food digestion. [Bar+14][NF15] Computer-aided methods have been developed alongside, to predict which peptides are susceptible to be released by digestive enzymes. [Min+08] However, the first protease of the GI tract, pepsin, has a low substrate specificity and therefore releases a great variety of peptides at the stomach level. It is therefore very difficult to predict the composition of protein hydrolysate that is emptied into the small intestine, i.e. the substrates of pancreatic proteases.

4.2.3 Pepsin

Pepsin is the enzyme responsible for protein digestion in the stomach. It is active in the low pH gastric environment; activity is zero at pH 6.5 and is highest around pH 2. [Joh+07] Pepsin in dilute hydrochloric acid is routinely used to simulate gastric conditions for *in vitro* assessment of protein digestion, [Min+14][GVS86] and since the 1950s, reports have been released that describe its proteolytic interaction with substrate proteins. [Chr55][Tan63][Kei92] One paper, by Hamuro *et al.*, [Ham+08] describes the substrate specificity of pepsin in some detail, presenting the probability of proteolytic cleavage according to the amino acids flanking the cleavage site.

4.2.4 In-silico models

In silico simulators have been developed to predict the sequence of peptides that may be released after peptic digestion. These include Expasy PeptideCutter [Gas+05] [6] and MEROPS [Raw+14]. The present study details an *in silico* model for protein hydrolysis by pepsin that departs from such qualitative prediction tools, in that the relative abundance of both substrates and products are quantitatively modelled during the course of the reaction. Additionally, pepsin substrate specificity values from Hamuro *et al.* [Ham+08] are used, which differ slightly from those used by PeptideCutter [Gas+05] or MEROPS[7] [Raw+14]. To develop and optimise the model, it was compared with experimental data from the peptic digestion of bovine lactoferrin, as published by Grosvenor *et al.*. [GHD14] This data was selected as suitable for model development because it used a simple purified protein system, and tracked not only the sequence of peptides released over the digestive time course, but provided quantitative information on peptide relative abundance that could be compared to the model outputs. The cleavage probability values from Hamuro *et al.* [Ham+08] were selected because of the greater specificity this team identified for pepsin, compared to previous reports. Also included was the effect of terminal residues on cleavage probabilities presented in Powers *et al.* [PHM77]

Featuring stochastic components, the proposed approach belongs to the category of Monte-Carlo models [Rip87], computational algorithms that rely on repeated random sampling to obtain numerical results. The basic idea behind these techniques is to use randomness to solve problems that might be deterministic in principle, but for which complete information is not available. Monte-Carlo models are often used in physical and mathematical problems, [Kro+14] and are most useful when it is difficult or impossible to tackle the case study with other approaches.

[6] `expasy.org/peptidecutter`

[7] `merops.sanger.ac.uk`, accessed December 2016.

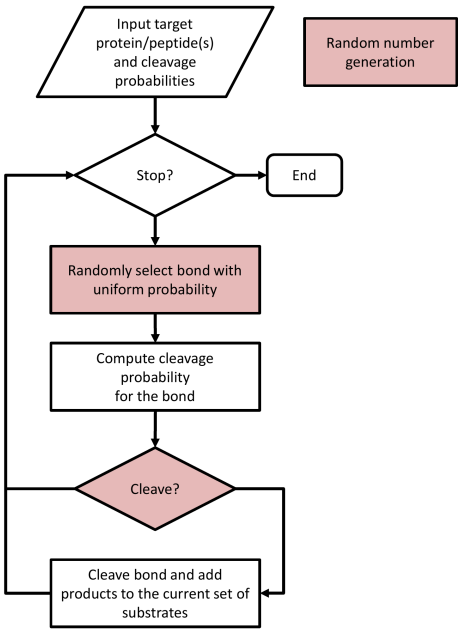


Figure 4.14: Flowchart of the proposed approach for the *in silico* simulation of enzyme behavior. The parts highlighted in red include random number generation.

4.2.5 Mathematical model

The objective of this article is to propose an *in silico* model for enzyme behaviour, simple enough to be run without resorting to computational clusters or super-computers, and at the same time able to return results that are qualitatively interesting for the end user. For the model to be computationally feasible, it cannot take into account the exact 3D structure of a protein, as finding the correct three-dimensional structure of a target amino acid sequence is in itself very difficult (NP-hard), requiring intensive calculations in a time that grows non-polynomially with the problem dimension. [Fra93] In practice, these problems cannot be solved exactly. Nonetheless, implicit information on enzyme behavior related to protein structure can be gathered from other sources: experiments on enzyme cleavage frequency on different proteins, [Ham+08] databases reporting disulfide bonds or glycosylations on the protein, [Con+14] and results for enzyme activity in proximity to the peptide termini [PHM77].

The proposed approach is a simple Monte-Carlo model, [Rip87] with the following flow (a more rigorous description of the proposed algorithm is reported in Figure 4.14):

1. The model is initialized with a user-defined number of copies of the target protein. For each copy, only the amino acid sequence is considered, including the presence of disulphide bonds or glycosylations.
2. At each iteration, a random bond between two adjacent amino acids is selected, with uniform probability on all peptide bonds.
3. The model then evaluates the cleavage probability for the bond P1-P1', $\mathscr{P}(cleavage|P1,P1')$, based on experimental data [Ham+08] and possibly heuristic considerations (see below the discussion for **Model II**). A random number $(0,1)$ is generated, and if the number is below the computed probability, the cleavage is considered successful.
4. If the cleavage is successful, the selected protein is removed, and the two resulting peptides are added to the model.

5. The algorithm goes back to point 2, unless a user-defined stop condition is satisfied. Such a stop condition might entail reaching a specific degree of hydrolysis (DH), or a time limit.

As the key element of the model lies in the computation of the cleavage probability for a certain bond, it is worth discussing this feature in more detail. Data from Hamuro *et al.*, [Ham+08] reported in Table 4.1, provides a good estimation of the frequency of pepsin action, if the amino acids to the left (P1) and to the right (P1') of a bond are known. Considering pepsin's activity as a stochastic process, the information on frequency can be easily interpreted as a conditional probability of cutting a bond, for example, $\mathscr{P}(cleavage|P1 = Y, P1' = F) = 0.65$.

Table 4.1: Frequency of pepsin cleavage between two amino acids, P1 (columns) - P1' (rows). Data taken from **Table 3a** in Hamuro *et al.* with permission from Wiley [Ham+08]

	F	L	M	C	E	W	Y	D	A	Q	N	T	S	G	V	I	K	H	R	P	Ave.
Y	65%	68%	30%	25%	70%	71%	36%	48%	48%	47%	39%	23%	31%	17%	11%	7%	0%	0%	0%	0%	33%
F	85%	84%	64%	75%	53%	40%	33%	37%	38%	24%	28%	21%	5%	9%	8%	8%	0%	0%	0%	0%	28%
W	60%	60%	50%	57%	63%	-	50%	17%	17%	33%	38%	17%	23%	17%	24%	30%	0%	0%	0%	0%	28%
I	65%	63%	62%	40%	36%	15%	20%	42%	30%	40%	25%	18%	13%	9%	9%	0%	2%	0%	0%	0%	24%
M	83%	58%	42%	0%	29%	33%	25%	20%	30%	20%	0%	11%	0%	11%	6%	0%	0%	0%	0%	0%	21%
V	50%	53%	61%	21%	31%	10%	16%	25%	28%	23%	16%	11%	12%	0%	4%	2%	0%	0%	0%	0%	19%
L	64%	56%	66%	36%	21%	29%	25%	12%	16%	7%	4%	8%	4%	13%	7%	1%	0%	0%	0%	0%	18%
C	20%	54%	0%	50%	36%	22%	0%	33%	6%	33%	25%	0%	20%	12%	0%	18%	0%	0%	0%	0%	17%
A	55%	54%	38%	18%	35%	8%	9%	13%	14%	7%	9%	7%	6%	4%	3%	2%	0%	0%	0%	0%	16%
E	42%	45%	29%	20%	9%	24%	19%	6%	6%	2%	0%	0%	4%	0%	4%	6%	0%	0%	0%	0%	10%
D	44%	46%	38%	0%	11%	21%	17%	5%	5%	0%	5%	2%	2%	4%	0%	0%	0%	0%	0%	0%	10%
R	42%	34%	26%	29%	9%	13%	16%	9%	8%	6%	4%	0%	5%	3%	0%	0%	0%	0%	0%	0%	10%
N	42%	45%	7%	0%	13%	0%	11%	0%	4%	5%	0%	0%	4%	0%	0%	0%	0%	0%	0%	0%	9%
S	52%	42%	22%	0%	4%	6%	14%	2%	5%	0%	0%	5%	2%	0%	3%	0%	0%	0%	0%	0%	9%
T	31%	27%	25%	35%	3%	29%	4%	12%	5%	3%	6%	9%	3%	2%	0%	0%	0%	0%	0%	0%	9%
H	43%	33%	29%	17%	6%	0%	10%	15%	22%	15%	0%	11%	3%	0%	0%	0%	0%	0%	0%	0%	9%
K	47%	33%	32%	0%	12%	13%	0%	2%	3%	3%	0%	4%	0%	2%	0%	2%	0%	0%	0%	0%	8%
Q	33%	26%	17%	0%	11%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	6%
P	17%	24%	18%	20%	8%	0%	6%	0%	5%	0%	0%	0%	2%	0%	11%	2%	0%	5%	0%	0%	6%
G	28%	7%	8%	6%	1%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	2%	0%	0%	0%	3%
Ave.	46%	44%	35%	23%	20%	17%	16%	13%	13%	10%	9%	7%	5%	4%	4%	2%	0%	0%	0%	0%	14%

In a first variant of the proposed model, labeled **Model I**, probabilities are computed according to the cleavage frequencies given the amino acids in positions P1 and P1' only (Table 4.1). The literature, however, reports additional data on how the cleavage probability may be influenced by amino acids in further positions (P4, P3, P2, ..., P2', P3', P4'), see Table 4.2, originally from Hamuro *et al.* [Ham+08], or by the proximity to the peptide termini, see Table 4.3, taken from Powers *et al.*.[PHM77]

While the aforementioned experimental data is unfortunately not quantitatively sufficient to properly compute complex conditional probabilities such as $\mathscr{P}(cleavage|P4, P3, P2, P1, P1', P2', P3', P4')$, it might still be heuristically exploited to modify the $\mathscr{P}(cleavage|P1, P1')$ obtained from Table 4.1. For example, in Table 4.2, it is easy to notice how the presence of certain amino acids in positions P2 and P2' can considerably alter cleavage frequency; and from Table 4.3 it is noticeable that proximity to a terminal end of a peptide can impede pepsin's action. For these reasons, a second variant of the proposed approach is presented, **Model II**, where the cleavage probability originally obtained from Table 4.1 is modified according to the following two heuristics:

1. to take into account the relative contribution of a specific amino acid in positions (P4,P3,P2,P2',P3',P4'), an altered probability is proposed:

$$\mathscr{P}_{alt} = \mathscr{P}_{or} * \frac{\mathscr{P}(cleavage|P_i = x)}{\mathscr{P}_{avg_i}}$$

, where $\mathscr{P}_{or}$ (the original probability) is multiplied by the ratio between the cutting probability

Table 4.2: Frequency of pepsin cleavage when each amino acid residue occupies positions P4...P4', reproduced from **Table 2** in Hamuro *et al.* with permission from Wiley. [Ham+08] The last column reports the number of each type of amino acid observed in the study.

	P4	P3	P2	P1	P1'	P2'	P3'	P4'	
F	17.8%	6.5%	6.8%	45.8%	28.1%	7.6%	13.8%	13.8%	542
L	12.0%	15.3%	11.1%	44.2%	18.1%	10.9%	17.3%	14.7%	1375
M	12.7%	21.3%	12.2%	35.0%	20.7%	12.2%	19.8%	14.1%	331
C	10.8%	20.7%	20.1%	23.4%	16.8%	12.1%	13.1%	16.5%	214
E	12.7%	14.7%	19.1%	19.9%	10.5%	17.9%	14.9%	11.2%	955
W	15.8%	9.2%	5.1%	16.7%	27.6%	11.6%	15.2%	15.2%	198
Y	17.8%	7.1%	8.6%	16.2%	33.1%	13.7%	13.1%	15.7%	451
D	16.7%	12.3%	17.5%	13.1%	10.5%	11.8%	11.7%	13.1%	799
A	11.8%	19.5%	16.6%	12.9%	16.4%	17.5%	10.0%	11.6%	989
Q	11.8%	12.1%	15.4%	9.7%	6.1%	12.7%	14.6%	12.9%	607
N	14.2%	9.9%	21.7%	9.0%	9.0%	8.5%	11.5%	14.3%	467
T	14.6%	18.9%	14.2%	6.7%	8.8%	15.6%	14.5%	11.0%	657
S	11.4%	19.5%	18.4%	5.3%	9.0%	17.6%	12.5%	13.5%	851
G	13.1%	19.1%	9.4%	4.1%	3.2%	8.2%	7.3%	15.1%	933
V	13.0%	17.7%	18.1%	4.0%	18.6%	22.1%	17.4%	12.0%	876
I	13.2%	19.0%	17.2%	2.4%	24.3%	18.3%	20.2%	12.8%	706
K	12.7%	1.6%	11.0%	0.3%	7.7%	14.3%	17.0%	12.7%	763
H	11.7%	1.7%	7.7%	0.2%	8.6%	2.6%	9.1%	11.5%	405
R	13.3%	1.2%	13.3%	0.0%	9.8%	21.1%	15.3%	13.9%	737
P	19.4%	13.7%	0.2%	0.0%	5.6%	1.3%	2.7%	20.7%	63

when amino acid x is in position i, and the average cutting probability over column i of Table 4.2. As an example, consider

$$\mathscr{P}(cleavage|P1 = Y, P1' = F) = 0.65$$

if amino acid K is in position P3 with respect to the considered bond, the original probability will become

$$\mathscr{P}_{alt} = 0.65 \cdot \mathscr{P}(C|P3 = K)/\mathscr{P}_{avg_3}$$

From the table, it is observable that $\mathscr{P}(C|P3 = K) = 0.016$, while the average probability in column P3 is 0.1305, so

$$\mathscr{P}_{alt} = 0.65 \cdot 0.016/0.1305 = 0.07969$$

with a considerable reduction of the original probability. Of course, to obtain the complete $\mathscr{P}_{alt}$, the alterations for P4, P2, P2', P3', P4' still need to be computed and applied.

2. from data in Table 4.3, appearing in *Powers et al.* [PHM77], it appears that pepsin is less likely to cut bonds close to a peptide terminal end. Comparing the frequency of cutting a bond

Table 4.3: Effect of terminal residues on pepsin cleavage probabilities, data taken from Powers *et al.* with permission from Springer [PHM77]. AA = any amino acid residue. HPA = an amino acid residue with high cleavage probability (e.g. F, M, L, C). AAZ = any terminal amino acid. The * indicates the position of the cleaved bond.

Positions	Probability	Positions	Probability
-HPA*AA-	0.44	AAZ-HPA*AA	0.11
HPA*AA-	0.18	AAZ-AA-HPA*AA-	0.47
-HPA*AAZ	0.11	-HPA*AA-AAZ	0.29

```
  1 MKLFVPALLS LGALGLCLAA PRKNVRWCTI SQPEWFKCRR WQWRMKKLGA PSITCVRRAF
 61 ALECIRAIAE KKADAVTLDG GMVFEAGRDP YKLRPVAAEI YGTKESPQTH YYAVAVVKKG
121 SNFQLDQLQG RKSCHTGLGR SAGWIIPMGI LRPYLSWTES LEPLQGAVAK FFSASCVPCI
181 DRQAYPNLCQ LCKGEGENQC ACSSREPYFG YSGAFKCLQD GAGDVAFVKE TTVFENLPEK
241 ADRDQYELLC LNNSRAPVDA FKECHLAQVP SHAVVARSVD GKEDLIWKLL SKAQEKFGKN
301 KSRSFQLFGS PPGQRDLLFK DSALGFLRIP SKVDSALYLG SRYLTTLKNL RETAEEVKAR
361 YTRVVWCAVG PEEQKKCQQW SQQSGQNVTC ATASTTDDCI VLVLKGEADA LNLDGGYIYT
421 AGKCGLVPVL AENRKTSKHS SLDCVLRPTE GYLAVAVVKK ANEGLTWNSL KDKKSCHTAV
481 DRTAGWNIPM GLIVNQTGSC AFDEFFSQSC APGADPKSRL CALCAGDDQG LDKCVPNSKE
541 KYYGYTGAFR CLAEDVGDVA FVKNDTVWEN TNGESTADWA KNLNREDFRL LCLDGTRKPV
601 TEAQSCHLAV APNHAVVSRS DRAAHVKQVL LHQQALFGKN GKNCPDKFCL FKSETKNLLF
661 NDNTECLAKL GGRPTYEEYL GTEYVTAIAN LKKCSTSPLL EACAFLTR
```

Figure 4.15: Amino acid sequence for bovine lactoferrin. The first 19 amino acids in the chain, marked in yellow, form a signal peptide that was not considered during the simulations. Amino acids highlighted in red and green mark the positions of glycosylations and disulfide bonds, respectively. [Con+14]

between two specific amino-acids in positions P1-P1', and the same situation, but closer to a terminus, it is clear that there is a huge interference with pepsin's activity, on average reducing the frequency to about a third of the original. For this reason, another heuristic consideration is added: if at least one amino acid is missing in positions P2, P2', P3, P3' (and thus the cutting point is close to a peptide terminus), the already altered probability is multiplied by 0.33.

Given these models, the DH at each iteration can be computed by simply comparing the number of cleaved bonds in the simulation with the total number of bonds in the target protein. It is important to notice that DH, not reaction time, is the only reference that can be used to compare the models' behavior to experimental data.

4.2.6 Simulations on the case study

The case study considered for the proposed methodology is the hydrolysis of bovine lactoferrin by pepsin. The amino acid sequence of the target protein, shown in Figure 4.15, is taken from Grosvenor *et al.*, [GHD14] which is also the source of the experimental data that was used to validate the approach. It should be noted that the first 19 amino acids in the chain are not considered, as they are part of a signal peptide. [Con+14] **Model I** and **Model II**, described in the previous section, are both tested.

To enable comparing the results produced by the simulations with the experimental data, the

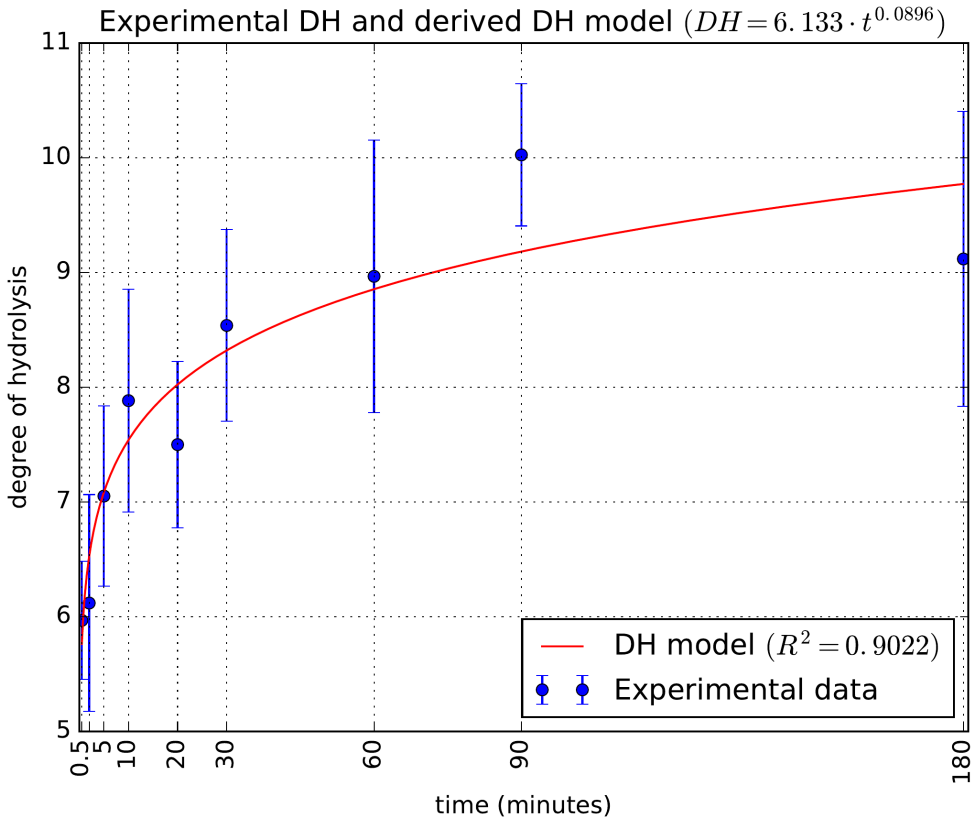


Figure 4.16: Power law model used to fit the experimental data on the degree of hydrolysis obtained from the trials. The original data from three trials (averaged) is shown in blue, the model prediction is shown in red.

time evolution of DH during the pepsin digestion experiments (3 replicates) was estimated by fitting a power law model, as shown in Figure 4.16.

For **Model I** and **Model II**, ten simulations are performed, each one including 500 copies of bovine lactoferrin (sequence reported in Figure 4.15), and using the probability tables and heuristic alterations described in Section 4.2.5, with a stop condition tied to reaching DH = 10%.

4.2.7 Results and discussion

The following results include, for each model, the sum of peptides generated during the ten simulations. While the model was allowed to reach a simulated DH of 10% in each simulation, the following results show comparisons up to DH = 9.18%, to allow comparison with the maximum DH reached during the reference experimental trials.

Experimental data

The experimental data, taken from Grosvenor *et al.*, [GHD14] includes a list of 105 monitored peptides. Since the original trials were performed on both pasteurized and unpasteurized bovine lactoferrin, with a different peptide list, the comparison will consider only peptides that are found in both pasteurized and unpasteurized versions of the protein. This is justified by Ahn *et al.*, [Ahn+13]

which shows that the reproducibility of peptic peptides is remarkably low, even when performed in the same conditions. Furthermore, a few monitored peptides contain a mutation of the original protein sequence, or are usually involved in disulfide bonds, and are thus discarded. At the end of this filtering process, a total of 87 monitored peptides remain. In the following, these peptides are denoted as monitored peptides. For each monitored peptide, the experimental concentration profile according to the DH evolution in relative quantities is available.

It is important to note that, while the proposed models keep track of all peptides produced, only a few can be experimentally observed through mass spectrometry; to perform a fair comparison, the non-observable simulated peptides have to be discarded. Three filters will then be alternatively applied to the data during the comparison. The first removes all peptides larger than 30 AA in length to analyze the peptide size distribution of the simulated peptides. The second only considers peptides of sizes 5-18 AA for comparison with the experimentally monitored peptides that all fell within this range of sizes. The third one discards all peptides containing an amino acid involved in a disulfide bond or glycosylation (Figure 4.15), since such peptides should not be detected by mass spectrometry *a priori* unless their post-translational modification is removed. These filters have considerable effects on the number of considered peptides: for example, applying all of them on the output of **Model I** at DH = 9.18%, reduces the number of simulated peptides from 6,201 to 1,847.

Comparison

A first qualitative assessment of the simulation results can be performed by analyzing the size distribution of the peptides as a function of DH, reported in Figure 4.17. Both plots seem to follow the expected behavior: quantities starts low for all categories, and build up as the DH increases. Interestingly, **Model I** predicts that cleavage products of just one amino acid in length will prevail, while **Model II** forecasts higher quantities of peptides of intermediate sizes.

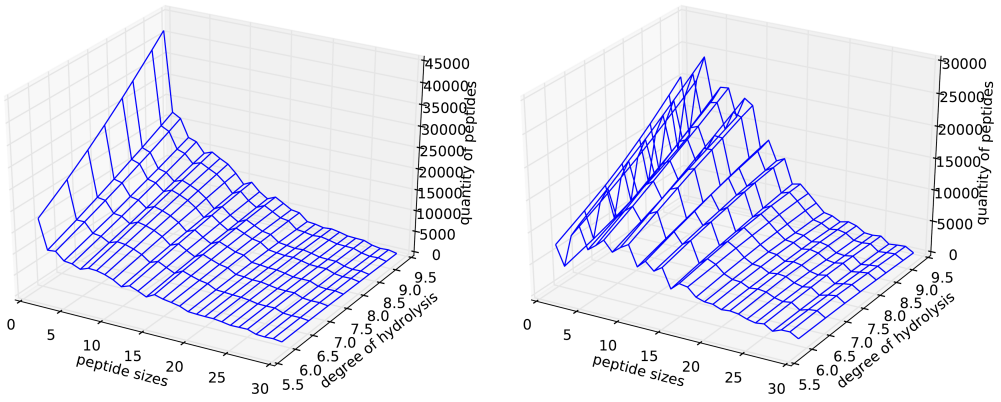


Figure 4.17: Size distribution of simulated peptides of sizes 1-30 amino acids, for **Model I** (left) and **Model II** (right).

The size distribution for DH = 9.18%, the final value registered during the experimental trials, is reported in Figure 4.18. **Model II**, taking into account the modified behavior when pepsin attacks links closer to the terminal parts of a chain, [PHM77] produces a higher percentage of peptides of 3-10 amino acids in length. These results seem in fact more consistent with the literature on pepsin-derived peptides, that report alternatively peptides of 6-10 [Kop+14] and 6-20 amino acids [Ahn+13]

as the most commonly observed. Data obtained from the simulations can also be compared with Egger *et al.*, [Egg+16] who found 1% of free amino acids after two hours of pepsin digestion of skim milk. In simulations for **Model II** at DH = 9.18%, the quantity of free amino acids produced is around 0.6% of the total number of amino-acid residues, which therefore seems to be in the right order of magnitude.

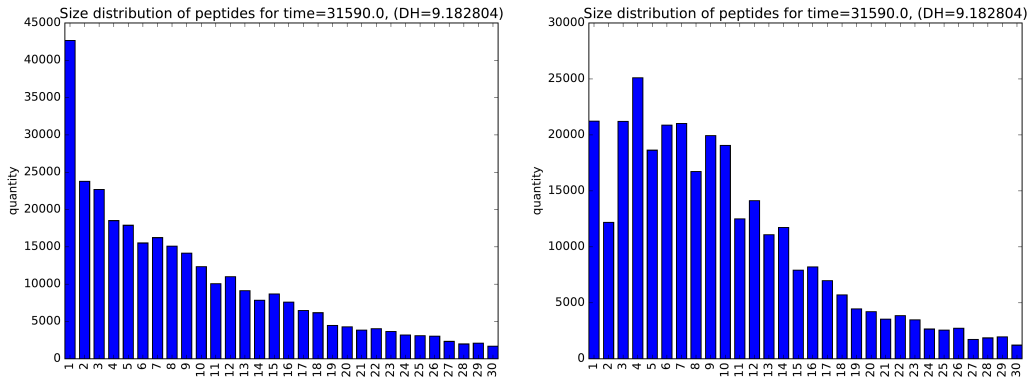


Figure 4.18: Size distribution of simulated peptides for DH = 9.18%. Results of **Model I** (left) predict higher quantities of smaller peptides, compared to **Model II** (right).

Monitored peptides and their relative abundance ranking obtained by simulations of **Model I** and **Model II** are reported in Table 4.4.

To better illustrate the presence of monitored peptides within the results of the simulations, comparative histograms are presented in Figure 4.19. It is interesting to notice that a great number of monitored peptides, highlighted in blue, are effectively generated with both models. Results also show that the predictions of **Model II** are much better than those of **Model I** because the ranking of monitored peptides according to the simulated quantities were remarkably improved despite a reduced variety of individual peptides. This highlights the importance of considering the effects of neighbor amino-acids and peptide termini to estimate cleavage probabilities, and provides more empirical validation for the heuristic approaches used in the computations.

The same data (Figure 4.19) can be analyzed from another point of view, by calculating how many experimentally detected (i.e. monitored) peptides are found among the most common peptides simulated by the models. These calculated percentages are shown in Figure 4.20. Interestingly, 50% of the monitored (i.e., 44 / 87) fall within the 299 most abundant peptides simulated by **Model I**, but within the 120 most abundant simulated by **Model II**. Again, this empirically corroborates the benefit of heuristically taking into account the presence of specific amino acids or termini in the P2-P4 and P2'-P4' positions. Taking into account the 25 most abundant peptides predicted by **Model II**, 14 of them were experimentally observed and monitored, and among the 11 unobserved peptides, 7 of them display a disulfide bond in position P3, P2, P2' or P3' of the bonds to cleave to generate them. This illustrates that the model could be improved with a more accurate evaluation of the effects of the proximities of post-translational modifications on the cleavage probabilities. Nevertheless, 10 of the 87 experimentally monitored peptides were produced despite the same configuration, meaning that the model would have missed them if cleavage probabilities were set to zero when a disulfide bond is in position P3, P2, P2' or P3'. Figure 4.20 also exposes that six of the monitored peptides are not simulated by either model. Cross-referencing the list of monitored-but-not-simulated peptides

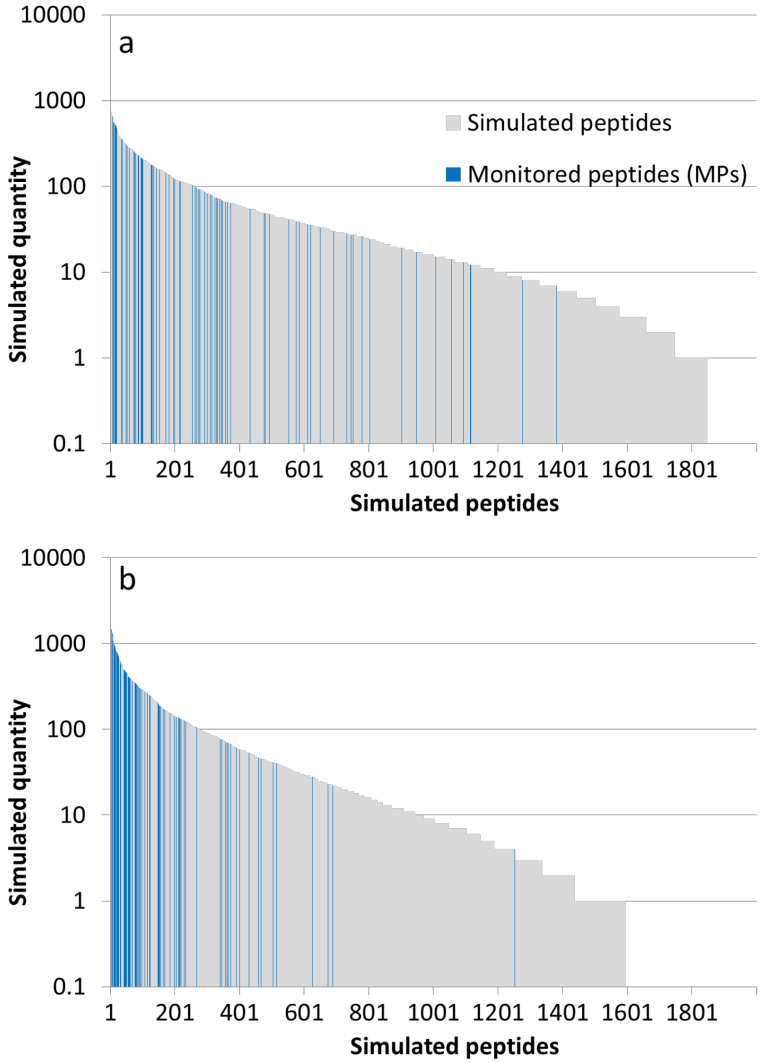


Figure 4.19: Histogram of the simulated peptides produced by **Model I (a)** and **Model II (b)** at DH = 9.18%, ordered by descending relative abundance. MPs detected during the experiments are highlighted in blue. As MPs are also the most abundant peptides produced, a good model fit will place these towards the left of the graph to match the peptides predicted to be most abundant. As a reference for the reader, since 10 simulations of 500 intial copies of the protein (5,000 intial copies) were performed, a simulated quantity of 1,000 corresponds to a molar percentage of 20, *i.e.* that 20 copies of the peptide considered were formed per 100 initial protein substrates.

Table 4.4: List of the 87 monitored peptides (MP), in alphabetical order. Their position in bovine lactoferrin is reported in column **Pos.**. Column **Rank I** and **Rank II** show the relative abundance ranking for each peptide, in simulations of **Model I** and **Model II**, respectively.

Sequence	Pos.	Rank I	Rank II
AEIYGTKESPQTHY	98-111	346	60
AKLGGRPTYE	668-677	277	48
AKLGGRPTYEE	668-678	15	33
ARSVDGKEDL	276-285	620	185
AVAKF	167-171	315	97
AVVARSVDGKEDL	273-285	-	-
DALNLDGGY	409-417	1008	356
DGGMVF	79-84	58	87
DLIWKL	284-289	300	215
DRTAGWNIPMGL	481-492	-	-
EAGRDPYKLRPVA	85-97	254	24
EAGRDPYKLRPVAA	85-98	575	84
EAGRDPYKLRPVAAE	85-99	99	57
EIYGTKESPQTHY	99-111	479	124
ENLPEKADRDQ	235-245	34	150
ENLPEKADRDQY	235-246	267	76
ENLPEKADRDQYE	235-247	338	152
ENLPEKADRDQYEL	235-248	172	47
EPLQGAVAKF	162-171	18	151
ESLEPLQG	159-166	-	-
ESLEPLQGA	159-167	-	-
ESLEPLQGAVAKF	159-171	-	-
EVKARYTRV	356-364	1094	459
EYLGTE	678-683	333	58
EYLGTEY	678-684	1056	340
FEAGRDPYKLRPVA	84-97	1115	267
FENLPEKADRDQYEL	234-248	649	674
FGSPPGQRDLL	308-318	10	31
FGSPPGQRDLLF	308-319	128	91
FKDSALGF	319-326	12	11
FKSETKNLL	651-659	20	92
FQLDQL	123-128	214	116
FQLFGSPPGQRDLL	305-318	1116	234
FVKETTVF	227-234	74	19
IAEKKADA	68-75	97	9
IAEKKADAVTL	68-78	96	216
IAEKKADAVTLDGGM	68-82	153	214
IYGTKESPQTHY	100-111	59	69
KGEADALNL	405-413	181	229
KGEADALNLDGGY	405-417	553	504
KKADAVTLDGGM	71-82	262	390
KKADAVTLDGGMVF	71-84	751	401
KNLRETAE	348-355	329	81
KSETKNLL	652-659	16	26
KYYGYTGA	541-548	274	49
LDGGMVF	78-84	746	205
LFGSPPGQRDLL	307-318	780	625
LKNLRETAE	347-355	948	345
LNLDGGY	411-417	552	169
LRIPSKVDSAL	327-337	7	3
LTTLKNLRE	344-352	1277	429
LTTLKNLRETAE	344-355	804	147
LTTLKNLRETAEE	344-356	324	80
NLDGGYIY	412-419	733	689
QLFGSPPGQRDLL	306-318	196	365
RPYLSWTESLEPLQG	152-166	1382	1252
RSVDGKEDL	277-285	200	44
RSVDGKEDLIWKL	277-289	432	466
RTAGWNIPMGL	482-492	268	78
SFQLFGSPPGQRDLL	304-318	-	-
SLEPLQGAVAKF	160-171	610	372
SWTESLEPLQG	156-166	903	515
TESLEPLQGAVAKF	158-171	88	16
VFEAGRDPYKLRPVA	83-97	309	77
VFEAGRDPYKLRPVAA	83-98	692	164
VFEAGRDPYKLRPVAAE	83-99	135	156
VKETTVF	228-234	19	4
VLKGEA	403-408	493	56
VLKGEADAL	403-411	78	45
VLKGEADALNL	403-413	142	107
VLKGEADALNLDGGY	403-417	477	219
VLLHQQAL	629-636	126	63
VTLDGGM	76-82	52	22
VTLDGGMVF	76-84	218	21
VVARSVDGKEDL	274-285	48	20
VVKKGSNF	116-123	73	13
VVKKGSNFQLDQL	116-128	373	362
WAKNLNRED	579-587	217	210
WAKNLNREDF	579-588	291	23
WAKNLNREDFRL	579-590	343	122
WIIPMG	144-149	585	43
WIIPMGIL	144-151	129	121
WNIPMGL	486-492	101	6
YLAVA	452-456	357	51
YLGSRY	338-343	38	14
YLGSRYLTT	338-346	363	198
YLGSRYLTTL	338-347	86	79

with the probability tables used as a base for the models, the reason for their absence becomes clear: all of these monitored peptides are in fact produced by cleavages that are not observed by Hamuro *et al.*, [Ham+08] and thus the derived probabilities are set to 0. Even heuristic alterations included in **Model II** cannot change an initial probability of 0. The complete list of the missing monitored peptides and the relative cutting points that generate them are reported in Table 4.5.

Table 4.5: Missing MPs, not produced by the simulations. Cross referencing their position with frequency tables reported in Figure 4.1, it is noticeable how cuts P1-P1' with the corresponding amino acids are not observed.

Sequence	Position	Cut	Probability
AVVARSVDGKEDL	273-285	H-A	$\mathscr{P}(cleavage \vert P1 = H, P1' = A) = 0.0$
DRTAGWNIPMGL	481-492	V-D	$\mathscr{P}(cleavage \vert P1 = V, P1' = D) = 0.0$
ESLEPLQG	159-166	T-E	$\mathscr{P}(cleavage \vert P1 = T, P1' = E) = 0.0$
ESLEPLQGA	159-167	T-E	$\mathscr{P}(cleavage \vert P1 = T, P1' = E) = 0.0$
ESLEPLQGAVAKF	159-171	T-E	$\mathscr{P}(cleavage \vert P1 = T, P1' = E) = 0.0$
SFQLFGSPPGQRDLL	304-318	K-S	$\mathscr{P}(cleavage \vert P1 = K, P1' = S) = 0.0$

A final assessment of the models' behavior can be performed through a comparative analysis of the heat map computed from the original data (taken from Grosvenor *et al.* [GHD14]) and heat maps obtained from simulated data, see Figure 4.21. It is worth reminding that the experimental data allow for relative quantifications of a peptide as a function of time or DH, but not between two different peptides. Thus, the corresponding heat map is normalized by the maximum of a column (corresponding to one amino-acid of the protein sequence). This contrasts with the heat map calculated from simulated results for which absolute quantities are predicted. Therefore, it is not

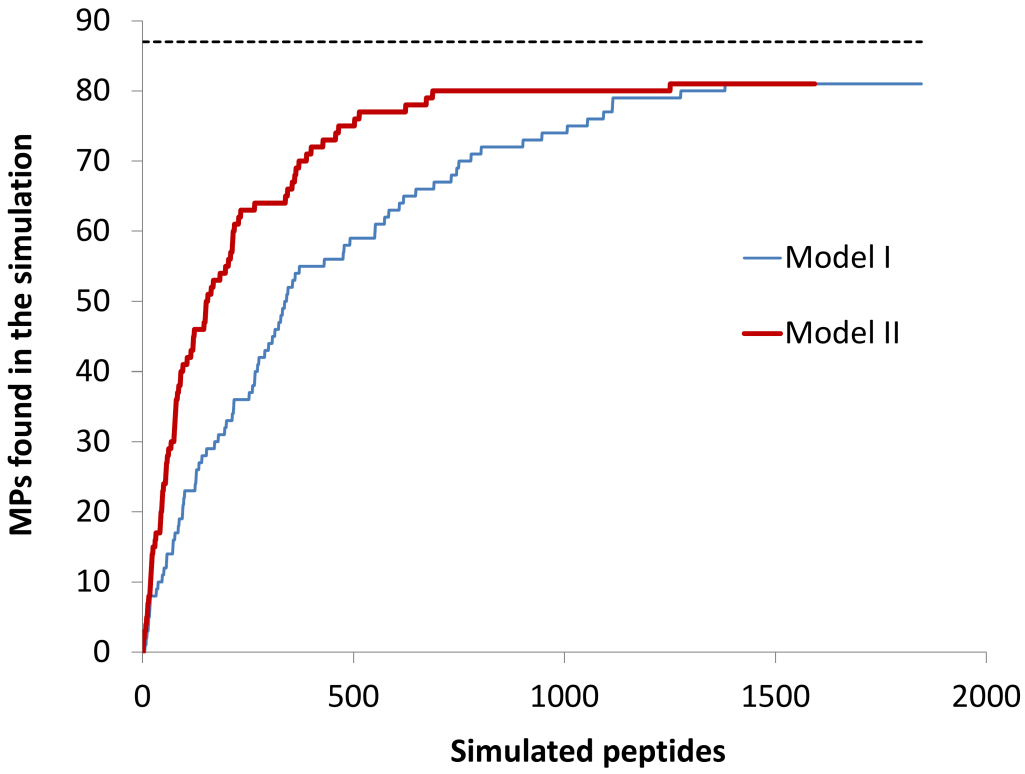


Figure 4.20: Cumulative number of MPs simulated by **Model I** (blue thin line) and **Model II** (red thick line), out of the 87 considered (dotted line) at DH = 9.18%. Both models fail to produce 6 MPs that are found experimentally.

possible to compare grey-levels horizontally in the experimental map nor between simulated and experimental maps for a given AA residue. Figure 4.21 nevertheless unambiguously demonstrates that both the experimental and simulated maps show gaps in the same locations. In other words, the most abundant simulated regions are indeed the ones where peptides are experimentally detected and monitored.

Although heat map rendition showed good correlations, the quantitative predictions of a given monitored peptide as a function of digestion time were less well-aligned with experimental observations (data not shown). Experimentally-observed peptides were more likely to form and then decline as additional cleavage formed smaller peptides, [GHD14] whereas both models tend to predict increasing abundance up to the stop point of 10% DH. This may be linked to the much higher number of peptides predicted by the models than monitored experimentally. This is a logical consequence of the great number of non-zero probabilities in Table 4.1, hence illustrating one limit of the proposed model. In a long term perspective, such approaches may nevertheless rely on more complete and accurate information regarding cleavage specificities, and include other alterations on cleavage probabilities related to structural and/or physicochemical properties of the substrates.

4.2.8 Conclusions

The models presented in this contribution represent a novel approach to modelling pepsin digestions, diverging from and complementing standard cleavage prediction tools like PeptideCutter in that the

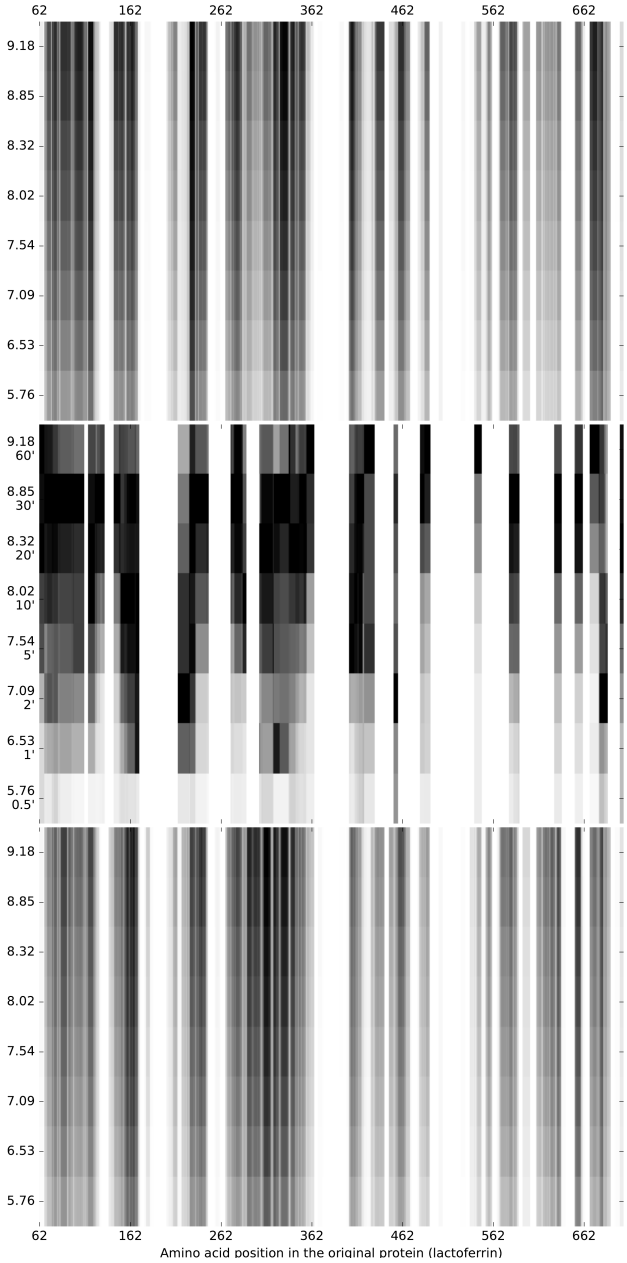


Figure 4.21: Heat maps from Model II **(a)**, original experimental data reproduced from [GHD14] with permission from the Royal Society of Chemistry, and including two disulfide peptides not included in the models **(b)**, and from Model I **(c)**.

quantitative peptide profile is predicted, rather than the peptide sequences only.

The two Monte-Carlo system models developed, utilising simpler (**Model I**) and more complex (**Model II**) cleavage probabilities, generated quantitative peptide profiles at degrees of hydrolysis up to 10%. On comparison to quantitative experimental data, **Model II**, which was modified to account for flanking residues up to four positions away from the site of cleavage (P4-P4') and altered cleavage probabilities near peptide termini, predicted a peptide profile that included nearly all (81 out of 87) of the experimentally monitored peptides, with increasing likelihood linked with experimental abundance (Figure 4.19). The six monitored peptides not predicted by either **Model I** or **Model II** were produced by cuts set to zero probability in the models, which were based on reported cleavage probabilities. [Ham+08] This signifies that improvements to models such as these will be possible with increasingly accurate information regarding cleavage specificities. This high number of predicted peptides represents satisfyingly accurate qualitation from the model.

Although the similarities between the experimental and simulated heat maps is very promissing, the quantitative predictions of a given peptide as a function of time does not match experimental observations so far. Futher developments of this model for pepsin activity will be made possible as increasingly accurate cleavage specificities become available. Other heuristic alterations could also be included within the model to account for structural features of the considered substrate, such as the accessibility of the peptide bonds, the flexibility and/or hydrophibicity of the considered chain. In the meantime, it will serve as a computationally simple approach to forecast the appearance of peptides from any protein, able to be run on an average end-user computer. A single experiment on 500 copies of lactoferrin, for example, takes less than 40 minutes to run on a Dell Latitude laptop, with an i7-2640M processor and 8 GB of RAM. The methodology can be useful, for example, in predicting the release of allergens or bioactives from well-characterised food products. Additionally, the basic principle may be adapted to any endoprotease, for which there is sufficient available data on cleavage specificities.

Conflict of interest

The authors declare no conflicts of interest.

Funding

Funded in part by Strategic Science Investment Fund (contract A21246) from AgResearch Ltd, New Zealand.

4.3 COST Action FoodMC

Methodologies and tools from Maths and Computer Science (MCS) are emerging as key contributors to modernization and optimization of processes in various disciplines: the agri-food sector, however, is not a traditional domain of application for MCS, and at the moment there is no community organized around solving the issues of this field. COST Action "Mathematical and Computer Science Methods for Food Science and Industry" (FoodMC) brings together scientists and practitioners from MCS and agri-food domains, stimulating the emergence of new research, and structuring a new community to coordinate further investigation efforts. Exploiting approaches originating at different sub-fields of MCS, from applied mathematical models to knowledge engineering, this COST Action covers two main topics: understanding and controlling agri-food processes; and eco-design of agri-food products.

4.3.1 Introduction

European Cooperation in Science and Technology (COST) is Europe's longest-running intergovernmental framework for cooperation in science and technology. Founded in 1971, COST holds a successful history of funding science and technology networks for over 40 years, offering scientists the opportunity to embark upon bottom-up, multidisciplinary cooperation across all science and technology domains.

Also known as COST Actions, these science and technology networks allow scientists to grow their ideas by sharing them with their peers. This gives impetus to their research, career and innovation. Researchers, engineers and scholars from both public and private sectors can set up their own network in any field of science and technology.

COST Actions grow throughout a funding period of 4 years. The funding covers networking activities such as meetings (e.g. travel, subsistence, local organiser support), conferences, workshops, short-term scientific missions (STSMs), training schools, publications and dissemination activities. COST does not fund research itself.

4.3.2 Challenge

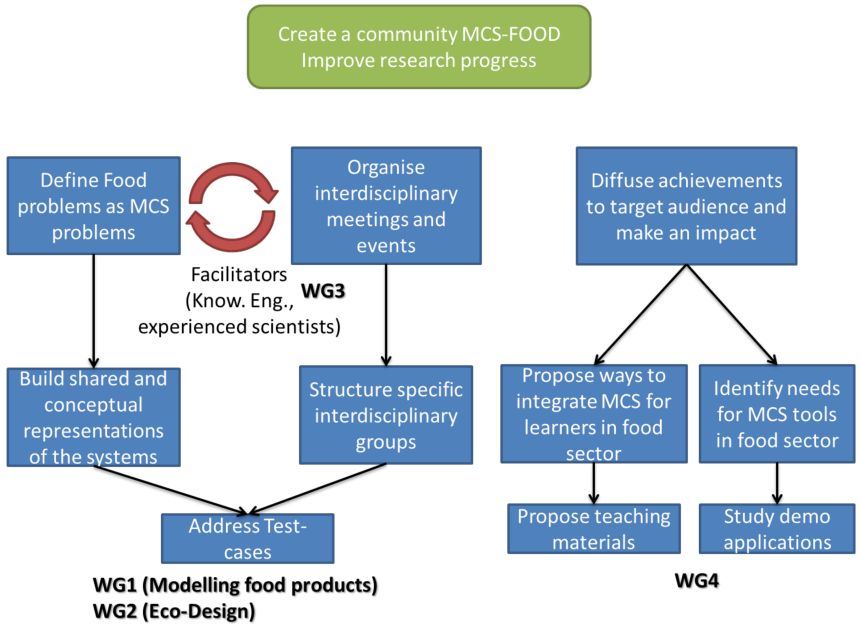


Figure 4.22: Organization of the Working Groups envisioned in the Action.

Food processing and agricultural products catering companies are one of the major employers and economic forces in the EU, representing both a central component of the agro-food system, and a crucial provider of biomaterials and biofuels. In recent years, this strategic industry has been facing unprecedented challenges, mainly concerning food security and the threat of climate change. Additionally, the production of processed agricultural products needs to adopt and comply with several new regulations, aiming at reducing waste, improving re-utilization of by-products, limiting energy consumption and lowering the overall environmental impact. These demanding objectives can only be achieved through appropriate adaptation and innovation in the food processing activity.

Disruptive innovations, however, require considerable economic efforts and the development of new skills not readily available in the agro-food domain, especially in small and medium enterprises (SMEs). A considerable number of unsustainable practices are still in place, due to the high cost of experimenting with new techniques on existing production/supply chains and validating scale-up.

There is evidence that developing Mathematical and Computer Science (MCS) models for the target processes can contribute to solving the issue [Try12], allowing even SMEs to optimize resource management and economic outputs, while guaranteeing the current levels of quality and availability of products. The agro-food industry, however, is not a traditional application domain for MCS: at the moment, there is no structured community around this issue, nor a coordinated effort to advance the state of the art; and building adequate mathematical models for specific applications is extremely knowledge and labour intensive. It is the role of academic research to initiate the development of methods, functional models, software or technologies, which will be critical to guide the evolution of the food processing industry with regards to the grand challenges of the future.

MCS researchers and practitioners can also benefit from working on agro-products industry applications, since the field provides considerable challenges to existing methodologies in MCS: uncertainty of the data, multi-scale description of the systems, coupling of models, representation of expert knowledge, etc. As the upcoming challenges for the industry grow more pressing, promoting cooperation between agro-food and applied mathematics becomes more and more urgent. The aim of this COST Action is thus to create a community of scientists and practitioners from the two different domains, stimulating the emergence of new research and ideas tackling these ambitious topics.

The development of novel mathematical and computer models, following the complex systems and knowledge engineering paradigms, has been slowly gaining support in the agro-food community over the last two decades [Per+16; Van+14]). Existing projects, however, are scattered and uncoordinated, focusing more on the solutions to specific issues than on an organized collection of demands and techniques in the field: for these reasons, major methodological breakthroughs, even stemming from applications, are still extremely rare. Although food production is a major industry in most countries, the number of publications dedicated to the treatment of food industry problems by means of innovative MCS modelling is well below that of other types of industries. Coordinating the currently divided research efforts is crucial to avoid re-discoveries and dispersion of useful data, and at the same time promoting the sharing of theoretical and experimental results. Moreover, the application domain of agro-food products is rich and multi-faceted, and research efforts so far have not been balanced over all aspects. As a result, several features of the considered challenges are not well understood, while the expertise around others should be further developed. Significant progress in the domain can be obtained by providing a roadmap with well-defined MCS problems, addressing critical issues in food processing, in particular food security and sustainability. This issue calls for the close collaboration of domain specialists with mathematicians and computer scientists.

4.3.3 Action organization

FoodMC officially started on **April 11, 2016** and included researchers from 28 different European nations (Belgium, Bosnia and Herzegovina, Bulgaria, Croatia, Cyprus, Denmark, Finland, France, Germany, Greece, Ireland, Israel, Italy, Lithuania, Luxembourg, Malta, The Netherlands, North Macedonia, Norway, Poland, Portugal, Romania, Serbia, Slovenia, Spain, Switzerland, Turkey, United Kingdom) and 4 partner countries (Canada, Morocco, Ukraine, USA).

The Action is divided into four Working Groups (WGs), organized as in Figure 4.22. Each WG is going to focus on a specific aspect of the network, ranging from exploring suitable real-world case

studies, to discussing industrials' and practitioners' needs for efficient modelling tools, to gathering and sharing information.

WG 1: Modelling food products and processes

This group is focused on MCS solutions for modelling food properties and food processes. As the domain is very large, the group identified the opportunities susceptible to lead to breakthroughs and to meet stakeholders' needs, focusing the work on the description of benchmark case-studies. This WG produced state-of-the-art reviews of food products and food process modelling, an overview of the scientific challenges and finally identified stakeholders concerns. WG 1 provided guidelines for research at both the fundamental and the applicative level. The WG members were in charge of identifying modelling approaches with a potential high impact on food sector activities, and define benchmark case-studies that will be addressed during Action workshops and STSMs. Finally, this WG promoted the use of MCS solutions in the food sector through the support and co-organization of training schools.

WG 2: Eco-design of food processes

This group described the kind of systems to be addressed by eco-design, the appropriate MCS techniques and tools to be used, it proposed illustrative/pedagogical case-studies, and defined the boundaries of this interdisciplinary research, that can potentially lead to the delivery of suitable methods and tools in the future. Mirroring the actions on WG 1, WG 2 described the state of the art for modelling in eco-design, addressing the complex network of interactions linking the agro-food activities together, which grows more intricate with system size (local, regional, international). To do so, the WG collected and integrate inputs from involved stakeholders and specialists from different disciplines, through dedicated meetings, workshops and STSM. Finally, it identified a few representative case studies that the scientific community could efficiently address.

WG 3: From scientific results to tools

The WG has the objective of promoting the development of computer applications, allowing a larger audience of users to exploit scientific results. Expert knowledge, experimental data and mathematical models have been used to answer the users' needs. WG 3 addressed the problem of the low delivery of operational tools based on food science research results. Existing applications, such as web semantic applications, knowledge-based systems, simulation tools, were adapted to the food sector. The WG identified the main needs of the users that can benefit the most from the development of such tools. A limited number of case-studies were identified and addressed during the Action, through specific workshops and STSMs, from the second year. At the end of the Action, results were disseminated to interested users through a comprehensive report, and a training school was organized with the gathered materials.

WG 4: Knowledge acquisition and diffusion

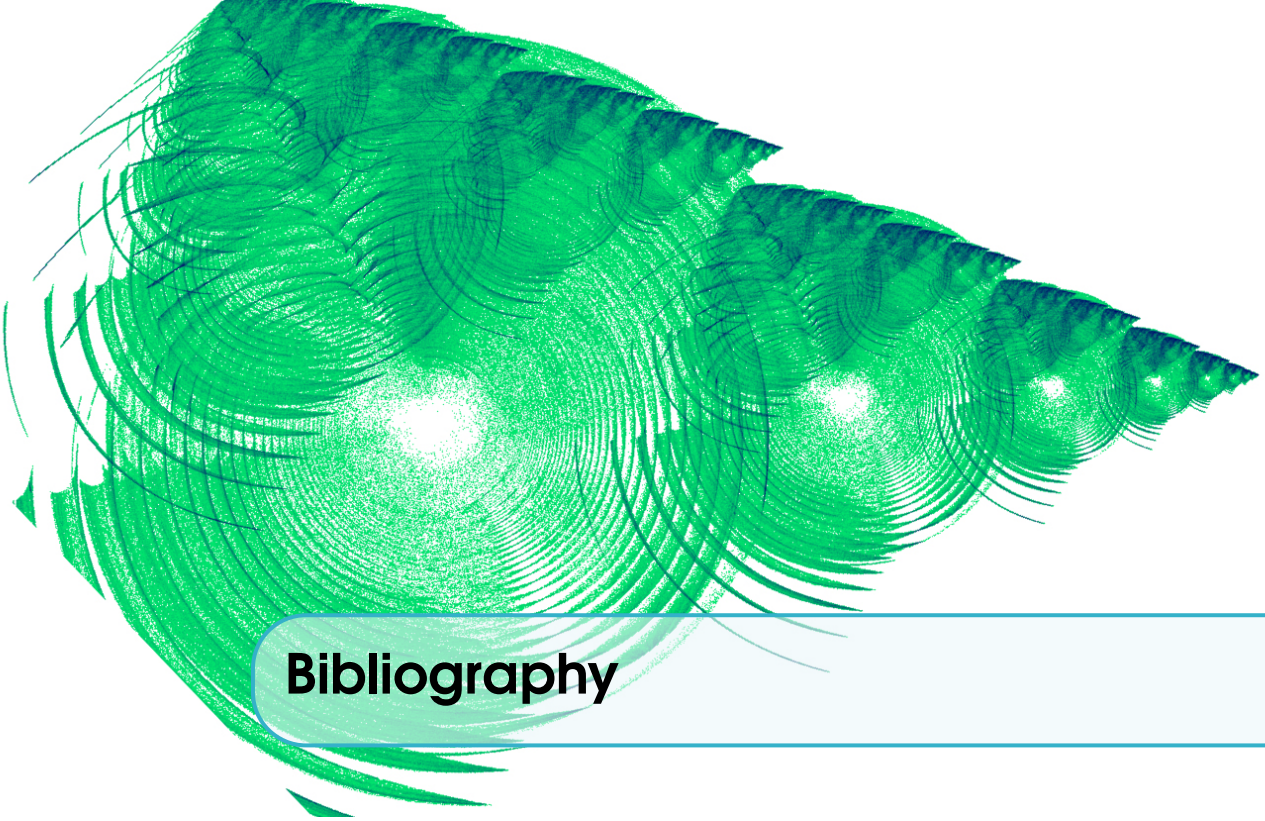
The WG's objective is to promote dissemination of the Action results, via the design and maintenance of the Action website. More than a support to convey related information and deliverables, the Action website stimulated knowledge transfer. This WG's activity also favored communication, as successful interdisciplinary research requires a mutual understanding between participants with different backgrounds. Face-to-face meetings of specialists from distant disciplines are generally insufficient to reach this point, because a typical expert relies on a great deal of tacit and implicit knowledge. Shared understanding can be promoted by formalization of tacit knowledge and participatory modelling using, for example, visualization techniques that support acquisition of knowledge. WG members

worked with the other WGs leaders to take part into activities that require strong interdisciplinary exchange, and organize the co-construction of knowledge models.

4.3.4 Outcomes

Besides yearly meetings, FoodMC sponsored 4 training schools for graduate and unergraduate students, focused on modelling in food science[8]. The Action supported in particular Ph.D. students and early career researchers by funding 31 STSMs that allowed them to gain new insights and techniques in different laboratories. FoodMC also supported researchers from inclusiveness target countries by awarding 13 conference grants, that made it possible for Ph.D. students and early career investigators to participate to conferences of their domain. Several new collaborations were started in the scope of FoodMC, and the papers co-wrote by different subset of authors are too many to be cited here. I will just mention the review and position papers that were completed under my direction by participants to COST Action [Car+21; Dje+18; Dje+19a; Dje+19b]. The Action also coordinated an application for a Marie-Curie Initial Training Network focused on food modelling, and produced a series of web seminars where different participants described their activities. More information is available on FoodMC's website, `https://www6.inrae.fr/foodmc`.

[8] `https://www.virprofood.org`



Bibliography

[Ahn+13] Joomi Ahn et al. “Accessing the reproducibility and specificity of pepsin and other aspartic proteases”. In: *Biochimica et Biophysica Acta (BBA)-Proteins and Proteomics* 1834.6 (2013), pages 1222–1229 (cited on pages 156, 157).

[Bar+14] Florence Barbé et al. “Tracking the in vivo release of bioactive peptides in the gut during digestion: mass spectrometry peptidomic characterization of effluents collected in the gut of dairy matrix fed mini-pigs”. In: *Food Research International* 63 (2014), pages 147–156 (cited on page 151).

[Bau+10] Cédric Baudrit et al. “Towards a global modelling of the Camembert-type cheese ripening process by coupling heterogeneous knowledge with dynamic Bayesian networks”. In: *Journal of Food Engineering* 98.3 (2010), pages 283–293 (cited on page 138).

[Bau+15] Cédric Baudrit et al. “A probabilistic graphical model for describing the grape berry maturity”. In: *Computers and Electronics in Agriculture* 118 (Oct. 2015), pages 124–135 (cited on page 141).

[Car+21] Otilia Carvalho et al. “Modelling Processes and Products in the Cereal Chain”. In: *Foods* 10.1 (Jan. 2021), page 82 (cited on page 167).

[Cha+17a] Thomas Chabin et al. “Interactive evolutionary modelling of living complex food systems: freeze-drying of lactic acid bacteria”. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM. 2017, pages 267–268 (cited on page 145).

[Cha+17b] Thomas Chabin et al. “LIDeOGraM : An interactive evolutionary modelling tool.” In: *Proceedings of the International Conference on Artificial Evolution (Evolution Artificielle)*. 2017 (cited on page 145).

[Cha+91] C.P. Champagne et al. “freeze-drying of lactic acid bacteria. A review”. In: *Canadian Institute of Food Science and Technology journal: Journal de l’Institut canadien de science et technologie alimentaire* (1991) (cited on page 144).

[Che+11] Jianshe Chen et al. “Development of a simple model device for in vitro gastric digestion investigation”. In: *Food & function* 2.3-4 (2011), pages 174–182 (cited on page 151).

[CBL97] Jie Cheng, David A Bell, and Weiru Liu. “An algorithm for Bayesian belief network construction from data”. In: *proceedings of AI & STAT’97*. 1997, pages 83–90 (cited on page 138).

[Chr55] L Korsgaard Christensen. “Concerning the pH optimum of peptic hydrolysis”. In: *Archives of biochemistry and biophysics* 57.1 (1955), pages 163–173 (cited on page 151).

[Con+14] UniProt Consortium et al. “UniProt: a hub for protein information”. In: *Nucleic acids research* (2014), gku989 (cited on pages 152, 155).

[Cou+12] Cécile Coulon-Leroy et al. “Prediction of vine vigor and precocity using data and knowledge-based fuzzy inference systems”. In: *Journal International des Sciences de la Vigne et du Vin* 46.3 (2012), pages 185–205 (cited on page 140).

[Cro+03] Marie-Josée Cros et al. “A biophysical dairy farm model to evaluate rotational grazing management strategies”. In: *Agronomie* 23.2 (2003), pages 105–122 (cited on page 144).

[DVG07] Z.-W. Dai, P. Vivin, and M. Génard. “Modelling the effects of leaf-to-fruit ratio on dry and fresh mass accumulation in ripening grape berries”. In: *VIII International Symposium on Modelling in Fruit Research and Orchard Management 803*. 2007, pages 283–292 (cited on page 144).

[Dai+09] Zhan Wu Dai et al. “Model-based analysis of sugar accumulation in response to source–sink ratio and water supply in grape (Vitis vinifera) berries”. In: *Functional Plant Biology* 36.6 (2009), pages 527–540 (cited on page 139).

[De 06] Kenneth A De Jong. *Evolutionary computation: a unified approach*. MIT press, 2006 (cited on page 146).

[Dje+18] Ilija Djekic et al. “Review on environmental models in the food chain - Current status and future perspectives”. In: *Journal of Cleaner Production* 176 (Mar. 2018), pages 1012–1025 (cited on page 167).

[Dje+19a] Ilija Djekic et al. “Cross-European initial survey on the use of mathematical models in food industry”. In: *Journal of Food Engineering* 261 (Nov. 2019), pages 109–116 (cited on page 167).

[Dje+19b] Ilija Djekic et al. “Scientific Challenges in Performing Life-Cycle Assessment in the Food Supply Chain”. In: *Foods* 8.8 (Aug. 2019), page 301 (cited on page 167).

[Dru99] Marek J Druzdzel. “SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: a development environment for graphical decision-theoretic models”. In: *Aaai/Iaai*. 1999, pages 902–903 (cited on page 138).

[Egg+16] Lotti Egger et al. “The harmonized INFOGEST in vitro digestion method: From knowledge to action”. In: *Food Research International* 88, Part B (2016), pages 217–225 (cited on page 158).

[FBR15] Michael Fadock, Ralph B Brown, and Andrew G Reynolds. “Visible-Near Infrared Reflectance Spectroscopy for Nondestructive Analysis of Red Winegrapes”. In: *American Journal of Enology and Viticulture* (2015), ajev–2015 (cited on page 140).

[FS10] MJ Ferrua and RP Singh. “Modeling the fluid dynamics in a human stomach to gain insight of food digestion”. In: *Journal of food science* 75.7 (2010), R151–R162 (cited on page 151).

[FXS14] MJ Ferrua, Z Xue, and RP Singh. “Dynamics of gastric contents during digestion—computational and rheological considerations”. In: *Food Structures, Digestion and Health*. Academic Press, 2014, pages 319–360 (cited on page 151).

[Fra93] Aviezri S Fraenkel. “Complexity of protein folding”. In: *Bulletin of mathematical biology* 55.6 (1993), pages 1199–1210 (cited on page 152).

[Gas+05] Elisabeth Gasteiger et al. “Protein identification and analysis tools on the ExPASy server”. In: *The Proteomics Protocols Handbook*. Edited by John M. Walker. Totowa, NJ: Humana Press, 2005, pages 571–607 (cited on page 151).

[GVS86] SF Gauthier, C Vachon, and L Savoie. “Enzymatic conditions of an in vitro method to study protein digestion”. In: *Journal of Food Science* 51.4 (1986), pages 960–964 (cited on page 151).

[GFC] M. Ghoniem, J.-D. Fekete, and P. Castagliola. “A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations”. In: *IEEE Symposium on Information Visualization*. IEEE (cited on page 136).

[GHD14] Anita J Grosvenor, Brendan J Haigh, and Jolon M Dyer. “Digestion proteomics: tracking lactoferrin truncation and peptide release during simulated gastric digestion”. In: *Food & function* 5.11 (2014), pages 2699–2705 (cited on pages 151, 155, 156, 160–162).

[Guo+15] Qing Guo et al. “Disintegration kinetics of food gels during gastric digestion and its role on gastric emptying: an in vitro analysis”. In: *Food & function* 6.3 (2015), pages 756–764 (cited on page 151).

[Ham+08] Yoshitomo Hamuro et al. “Specificity of immobilized porcine pepsin in H/D exchange compatible conditions”. In: *Rapid Communications in Mass Spectrometry* 22.7 (2008), pages 1041–1046 (cited on pages 151–154, 160, 163).

[Joh+07] Nikki Johnston et al. “Activity/stability of human pepsin: implications for reflux attributed laryngeal disease”. In: *The Laryngoscope* 117.6 (2007), pages 1036–1039 (cited on page 151).

[Kei92] Borivoj Keil. *Specificity of proteolysis*. Springer–Verlag, 1992, page 336 (cited on page 151).

[Kop+14] Katrin A Kopf-Bolanz et al. “Impact of milk processing on the generation of peptides during digestion”. In: *International Dairy Journal* 35.2 (2014), pages 130–138 (cited on page 157).

[Koz92] John R Koza. *Genetic Programming, On the Programming of Computers by Means of Natural Selection. A Bradford Book*. MIT Press, Cambridge, MA, USA, 1992 (cited on page 146).

[KPB14] Josua Krause, Adam Perer, and Enrico Bertini. “INFUSE: interactive feature selection for predictive modeling of high dimensional data”. In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pages 1614–1623 (cited on page 145).

[Kro+14] Dirk P Kroese et al. “Why the Monte Carlo method is so important today”. In: *Wiley Interdisciplinary Reviews: Computational Statistics* 6.6 (2014), pages 386–392 (cited on page 151).

[Le +14] Steven Le Feunteun et al. “Impact of the dairy matrix structure on milk protein digestion kinetics: mechanistic modelling based on mini-pig in vivo data”. In: *Food and bioprocess technology* 7.4 (2014), pages 1099–1113 (cited on page 151).

[LPT16] Evelyne Lutton, Nathalie Perrot, and Alberto Tonda. *Evolutionary Algorithms for Food Science and Technology*. Wiley, Nov. 2016 (cited on page 136).

[Min+14] M Minekus et al. “A standardised static in vitro digestion method suitable for food–an international consensus”. In: *Food & function* 5.6 (2014), pages 1113–1124 (cited on page 151).

[Min+08] Piotr Minkiewicz et al. “Food Peptidomics”. In: *Food Technology & Biotechnology* 46.1 (2008) (cited on page 151).

[Mur02] Kevin Patrick Murphy. “Dynamic bayesian networks: representation, inference and learning”. PhD thesis. University of California, Berkeley, 2002 (cited on page 137).

[NF15] Alice B Nongonierma and Richard J FitzGerald. “The scientific evidence for the role of milk protein-derived bioactive peptides in humans: A Review”. In: *Journal of Functional Foods* 17 (2015), pages 640–656 (cited on page 151).

[Pas+11] Stephanie Passot et al. “Quality degradation of lactic acid bacteria during the freeze drying process: Experimental study and mathematical modelling”. In: 2011 (cited on page 144).

[Pea14] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 2014 (cited on page 137).

[Ped+11] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *Journal of Machine Learning Research* 12.Oct (2011), pages 2825–2830 (cited on page 144).

[Per+15] Nathalie Perrot et al. “A Decision Support System Coupling Fuzzy Logic and Probabilistic Graphical Approaches for the Agri-Food Industry: Prediction of Grape Berry Maturity”. In: *PLOS ONE* 10.7 (July 2015). Edited by Zhaohong Deng, e0134373 (cited on page 140).

[Per+16] Nathalie Perrot et al. “Some remarks on computational approaches towards sustainable complex agri-food systems”. In: *Trends in Food Science & Technology* 48 (2016), pages 88–101 (cited on page 165).

[PHM77] James C Powers, A Dale Harley, and Dirck V Myers. “Subsite specificity of porcine pepsin”. In: *Acid Proteases: Structure, Function, and Biology*. Springer, 1977, pages 141–157 (cited on pages 151–155, 157).

[Raw+14] Neil D Rawlings et al. “MEROPS: the database of proteolytic enzymes, their substrates and inhibitors”. In: *Nucleic acids research* 42.D1 (2014), pages D503–D509 (cited on page 151).

[Ray+10] M. Raynal et al. “Epicure, a geographic information decision support system risk assessment of downy and powdery mildew epidemics in Bordeaux vineyards”. In: 2010 (cited on page 139).

[Rip87] Brian D Ripley. *Stochastic simulation*. John Wiley & Sons, 1987 (cited on pages 151, 152).

[SL09] Michael Schmidt and Hod Lipson. "Distilling free-form natural laws from experimental data". In: *science* 324.5923 (2009), pages 81–85 (cited on page 146).

[Sic+11] M. Sicard et al. "Expert knowledge integration to model complex food processes. Application on the camembert cheese ripening process". In: *Expert Systems with Applications* 38.9 (Sept. 2011), pages 11804–11812 (cited on pages 137, 139, 140).

[SJ60] R. G. D. Steel and H. James. *Principles and procedures of statistics: with special reference to the biological sciences*. McGraw-Hill, 1960 (cited on page 142).

[Tan63] J Tang. "Specificity of pepsin and its dependence on a possible hydrophobic binding site". In: *Nature* 199.4898 (1963), pages 1094–1095 (cited on page 151).

[Try12] Gilles Trystram. "Modelling of food and food processes". In: *Journal of Food Engineering* 110.2 (2012), pages 269–277 (cited on page 165).

[Tur+17] Cagatay Turkay et al. "Supporting theoretically-grounded model building in the social sciences through interactive visualisation". In: *Neurocomputing* (2017) (cited on page 145).

[Van+14] Harald GJ Van Mil et al. "A complex system approach to address world challenges in food and agriculture". In: *Trends in Food Science & Technology* 40.1 (2014), pages 20–32 (cited on page 165).

[Vel+15] H. Velly et al. "Cyclopropanation of unsaturated fatty acids and membrane rigidification improve the freeze-drying resistance of Lactococcus lactis subsp. lactis TOMSC161". In: *Applied microbiology and biotechnology* 99.2 (2015), pages 907–918 (cited on pages 144–146).

[Vel+14] Hélène Velly et al. "Cell growth and resistance of Lactococcus lactis subsp. lactis TOMSC161 following freezing, drying and freeze-dried storage are differentially affected by fermentation conditions". In: *Journal of applied microbiology* 117.3 (2014), pages 729–740 (cited on page 144).

[Zad65] L.A. Zadeh. "Fuzzy sets". In: *Information and Control* 8.3 (1965), pages 338–353. ISSN: 0019-9958 (cited on page 141).

5. Health Applications

The domain of health care offers a large variety of complex, practical problems that can in principle be tackled with algorithms. These problems often present complex obstacles, such as a considerable inter-sample variance and high feature dimensionality, making them particularly suited for machine learning and optimization approaches described in the previous sections. In this chapter, I describe applications I worked on, where Artificial Neural Networks (ANNs) and Evolutionary Algorithms (EAs) are used to face important challenges in the field.

In the first part (Section 5.1), I present automatic the results of a research line tackling the automatic discovery of primer sets, characteristic subsequences of RNA used to identify virus strains, with the specific application to SARS-CoV-2 detection [Lop+21a; Lop+21b]. In a second part (Section 5.2), feature selection techniques are applied to the identification of short gene expression signatures for the classification of cancer types [Lop+18; Lop+19; Lop+20].

5.1 Automatic Design of Primer Sets

As the pandemic of SARS-CoV-2 continues to affects the globe, researchers and public health teams around the world monitor the virus for acquired mutations that may lead to higher risks of developing COVID-19 or vaccine resistance. Prompt and widespread diffusion of information related to viral threats plays a critical role in research and mitigation of outbreaks [Waa+20]. This is especially true when viruses mutate rapidly under clinical, therapeutic or vaccine pressure.

Several molecular kits have been proposed and developed to diagnose SARS-CoV-2 infections. Most kits rely on the amplification of one or several genes of SARS-Cov-2 by real-time reverse transcriptase-polymerase chain reaction (qRT-PCR) [Afz20; Org+20], using *primers*. Primers are short RNA sequences, used to detect the presence of a specific virus in a host organism. Good qRT-PCR primer candidates need to possess several qualities, including having a PCR product size or amplicon between 100 - 200 bps, with a melting temperature (T_m) close to 60°C, a presence of C and G bases representing 40-60% of the sequence, and of a typical length of 18-22 nucleotides.

As new variants are identified, like B.1.1.7 from the UK, B.1.351 from South Africa, or P.1/B.1.1.28.1 from Brazil, developing primers specific to these potentially more dangerous strains

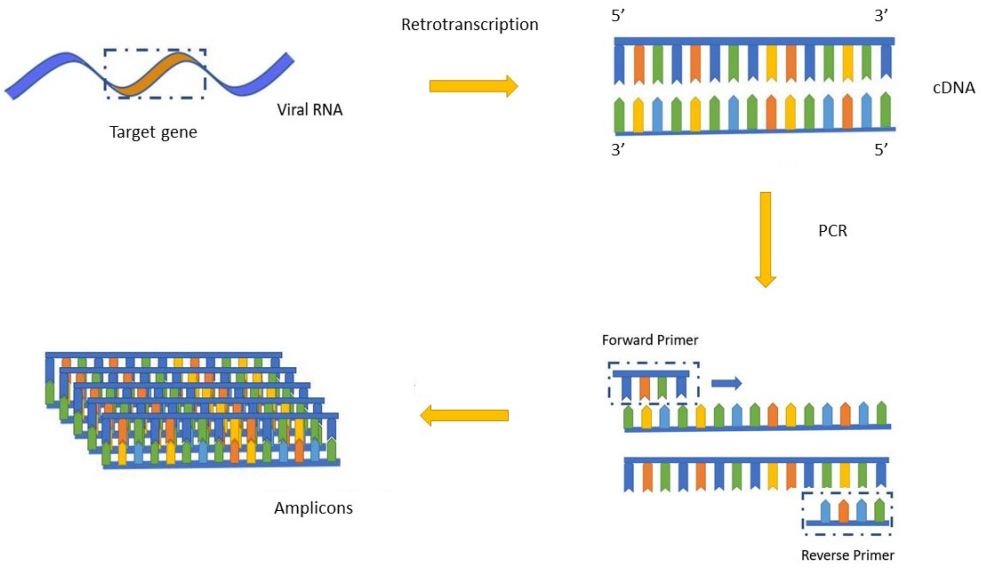


Figure 5.1: PCR amplification of a section of interest in viral RNA. A primer uniquely identifies a section of the viral genome.

becomes pivotal in locating and containing the disease. Primer design, however, is traditionally a long process, where domain experts create a canonical sequence for the target virus, analyze the virus' structure, identify promising areas that might be unique to the strain, and create candidates that are then tested first in-silico, and then in the lab. The emerging adoption of machine learning techniques in the health sector, together with the free access to an unprecedented amount of data, opened the way for semi- and fully automated solutions for primer design.

In the following sections, basic background information on primer design is provided, and two automatic approaches to primer design, with their respective results, are described.

5.1.1 Primer design

A primer or oligonucleotide is a short nucleic acid sequence that helps to start the DNA synthesis in a Polymerase Chain Reaction (PCR) assay. The presence of primers is necessary because the DNA polymerase can only attach new nucleotides to an existing strand of nucleotides to make copies of a DNA fragment[1]. Primers typically have a length of 18-22 nucleotides and are essential components of the procedure, as they determine the specificity of PCR [PBH08]. In the case of SARS-CoV-2, the viral RNA will be first transformed into single-strand complementary DNA (cDNA), and then the section of interest will be amplified. The forward and reverse primer attach in different directions of the targeted section in the cDNA sequence. If the cDNA sequence does not contain the forward and reverse primer sequences, then they will not attach and consequently won't be amplified, see Fig. 5.1 for a summary of the procedure.

Primer design is a crucial step of the process, because of its relationship with the success and quality of PCR analyses. Different factors can affect primers' efficiency, including; dimer formation

[1] https://www.nature.com/scitable/definition/primer-305/

(high self-complementary formation), stem loop interference, GC content (presence of C and G bases in the DNA sequence), high 3' stability (presence of C/G pairs at the 3' end) and extreme melting temperatures. While different computer programs have been proposed to facilitate the design process, it is still necessary to develop new algorithms to select the best PCR primers and avoid errors when using them in the laboratory [LB10; Rod+15].

The traditional approach to primer generation is to first use a canonical sample of the target variant, a summary of all samples available, and then analyze the differences with respect to the original virus. The differences, mainly mutations, are going to be possible locations of interest, and candidate primers can be designed by hand around the mutation. In the case of SARS-CoV-2, all the mutations are referenced to the canonical sequence *NC*_045512.2 [Wu+20]. In addition, several parameters need to be verified such as T_m close to 60°, %GC content close to 50%, self complementarity, and high 3' stability. Then, the designed primer has to be checked as a unique subsequence, not appearing more than once inside the same sample. This is traditionally carried out with the BLAST software, from the National Center for Biotechnology Information (NCBI) [Bor+13]. Nevertheless, if a sequence is not available in BLAST it cannot be identified using this method, as is the case for many of the new samples recently sequenced from the SARS-CoV-2 virus and its variants.

5.1.2 Convolutional neural networks

A Convolutional Neural Network (CNN) is a type of ANN featuring one or more convolutional layers. Such layers take inspiration from the human visual system [Fuk80], and are able to recognize patterns appearing in a stream of information, such as combinations of pixels in an image, or a subsequence of bps in RNA, independently from their absolute position.

Convolution is better described by an example: let's consider an input tensor (a generic n-dimensional matrix) of size 5*5, with a tensor of weights 3*3. In CNNs, a sliding window is used to represent the receptive field: using a sliding window of size 3*3 in the example would result in the convolution operation presented in Figure 5.2. The sliding window and the weights are turned into a vector, and the dot product of each is summed up. After the dot product, there is an activation function (for each neuron), typically a rectified linear unit, commonly referred to as *ReLU* [NH10], with a function $f(x) = max(x,0)$. This will result in a single value. The same procedure is repeated to the whole input tensor, resulting in a 3x3 output tensor. In practical applications, input tensors can have even higher dimensions. To further reduce the output size, a $maxpooling$ operation can be performed, where only the highest value in a sliding grid is reported to the subsequent layer. Following the example of the convolution operation, the max pooling of the 3x3 output tensor with a 2x2 sliding window, will result in a 2x2 output tensor. In a complete CNN, the resulting 4 values will be transmitted to a fully connected rectifier layer with a $softmax$ function, for classification applications. The $softmax$ function reduces a k-dimensional vector of arbitrary real values to a k-dimensional vector of real values in range $(0,1)$, that add up to 1: this can be used to associate each class with a probability.

Once the structure of a CNN is defined, its internal weights need to be optimized to fit the target problem. This operation is performed through backpropagation. In backpropagation, the initial system output is compared to the desired output, and the system is adjusted until the difference between the two is minimized, typically resorting to gradient descent optimization, using cross-entropy loss as the function to minimize. Cross-entropy loss for one-hot labels is defined as:

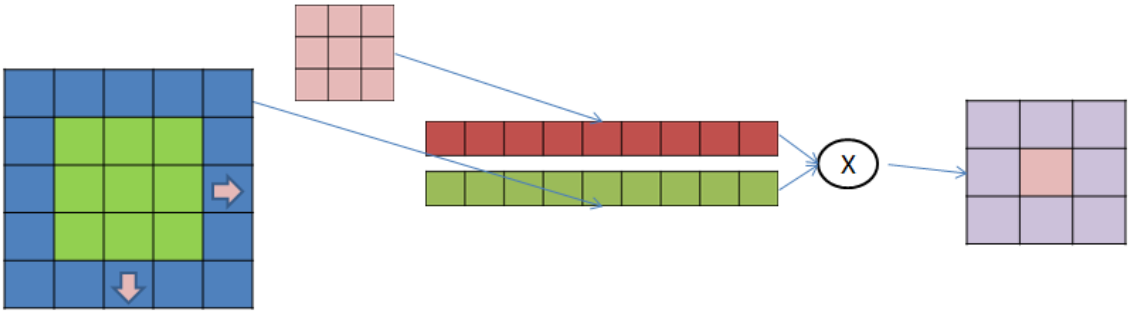


Figure 5.2: Example of convolution. A 5*5 input tensor (for example, a 5*5 pixel picture in black and white), is passed through a weight tensor of size 3*3.

$$L = \sum_{j=1}^{N} \sum_{i=1}^{M} -t_j^{(i)} \log z_j^{(i)} \tag{5.1}$$

where $t_j = (0,\ldots,0,\underbrace{1}_{k},0,\ldots,0)$ is the desired output vector and z_j the calculated output vector, if the example j belongs to the m-th class, and

$$z_j^{(i)} = \frac{e^{f_j}}{\sum_{i=1}^{M} e^{f_i}}, \tag{5.2}$$

is the softmax function, and N the number of samples.

CNN architecture

After a few trial runs with varying hyperparameters, the final architecture is selected and reported in Fig. 5.3. The CNN used during all the experiments is composed of one convolutional layer with 12 different filters or weights (each with window size 21) with maxpooling (with pool size and stride 148), a fully connected layer (196 rectified linear units with dropout probability 0.5), and a final softmax layer with 5 units, to differentiate the different classes of Coronavirus strains. The optimized used is Adaptive Momentum (ADAM) [KB14], with learning rate 10^{-5} and a batch size of 50 samples, run for 1,000 epochs.

The convolutional layer of the network, in simple terms, is analyzing subsequences of 21 base pairs that can appear in different points of the virus genome. The size selected is 21 base pairs, as designed primers for real-time PCR (RT-PCR) tests have a length of 18-22 bps normally. The pool size of the maxpooling represents the interval in which a specific 21-bps sequence can be recognized (in this case, 148 positions). Through the training process, the convolutional layer is *de-facto* learning new features to characterize the problem, directly from the data. In this specific case, the new features are 21-bps sequences that can more easily separate different virus strains. By analyzing the result of each filter in a convolutional layer, and how its output interacts with the corresponding max pooling, it is possible to detect human-readable sequences of base pairs that might provide domain experts with relevant information. It is important to notice that these sequences are not bound to specific locations of the genome; thanks to its structure, the CNN is able to detect them and recognize their importance even if their position is displaced in different samples.

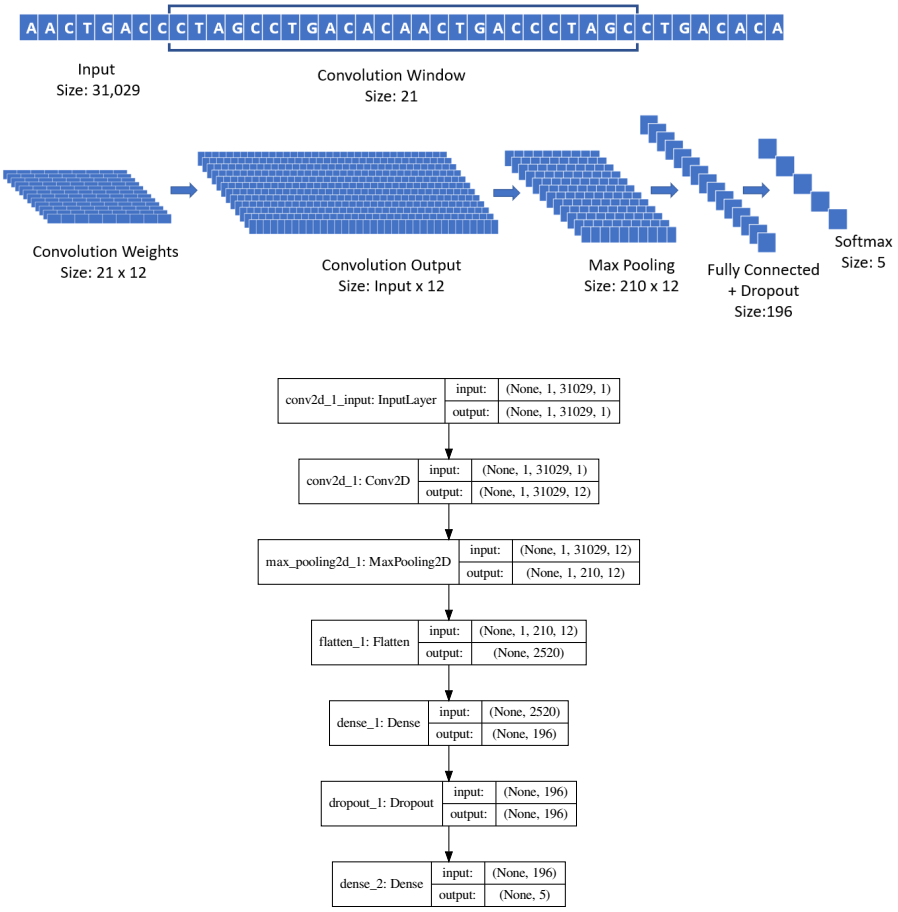


Figure 5.3: Architecture of the CNN used in the experiments for automated primer design.

Dataset

583 sequences (**.fasta* files) are downloaded from the NGDC repository [Bei13] on March 15^{th}, 2020 (Table 5.1). 30 SARS-CoV-2 sequences are left out, while the rest of the data is repeatedly split into 80% for training, 10% for validation, 10% for testing. The trained CNN described above obtained a mean accuracy of 98.73% on test data in a 10-fold cross-validation. Once the network is trained, in a first analysis, the inputs and outputs of the convolutional layer are plotted, to visually inspect for patterns. As an example, Fig. 5.4 reports the visualization of the first 1,250 bps of each of the 553 samples from the NGDC repository used in the validation process.

Each filter slides a 21-bps window over the input, and for each step produces a single value. The output of a filter is thus a sequence of values in $(0, 1)$. The output of the max pooling for each of the 12 filters is then further inspected for patterns. It is noticeable how samples belonging to different classes can be already visually distinguished. At this step, filter 0 is identified as the most promising, as it seems to focus on a few relevant points in the genome, that could correspond to meaningful cDNA sequences.

Given this data, it is now possible to identify the 21-bps sequences that obtained the highest output values in the max pooling layer of filter 0, in a section of 148 positions. This process results

Table 5.1: Organism, assigned label, and number of samples in the unique sequences for the NGDC repository (left) and for query: *gene="ORF1ab" AND host="homo sapiens" AND "complete genome"* in the NCBI repository (right). The NCBI organism naming convention [Miz07] is adopted here.

Organism	Label	Number of Samples
SARS-CoV-2	0	96
MERS-CoV	1	240
HCoV-OC43	2	132
HCoV-229E	2	22
HCoV-EMC	2	6
HCoV-4408	2	2
HCoV-NL63	3	58
HCoV-HKU1	3	17
SARS-CoV	4	7
SARS-CoV P2	4	1
SARS-CoV HKU-39849	4	1
SARS-CoV GDH-BJH01	4	1
Total Samples	-	583

Organism	Label	Number of Samples
SARS-CoV-2	0	68
MERS-CoV	1	180
HCoV-OC43	1	105
HCoV-NL63	1	29
HCoV-HKU1	1	13
HCoV-4408	1	2
HCoV-229E	1	3
HCoV-EMC	1	3
HAstV-VA1	1	1
HAstV-BF34	1	1
HMO-A	1	1
HAstV-SG	1	1
Total Samples	-	407

in 210 (31,029 divided by 148) *max pooling features*, each one identifying the 21-bps sequence that obtained the highest value from the convolutional filter, in a specific 148-position interval of the original genome: The first max pooling feature will cover positions 1-148, the second will cover position 149-296, and so on. The whole set of max pooling features is graphed for the complete data 4,410 (210*21), Fig. 5.5.

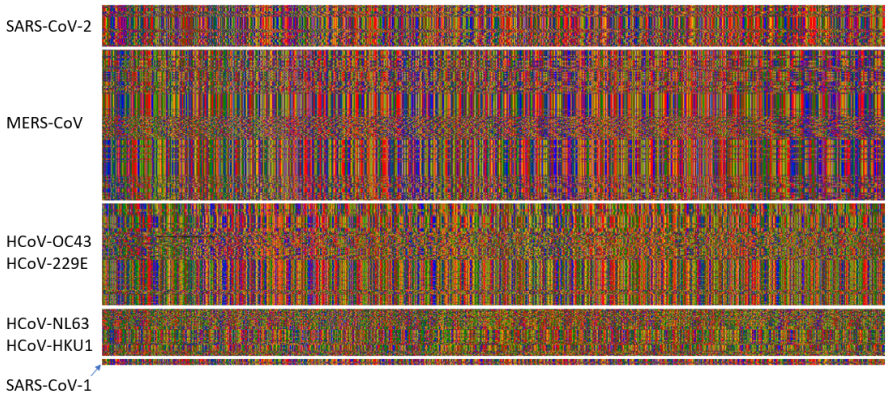


Figure 5.4: cDNA visualization for the first 1,250 bps from the input NGDC dataset, for each of the 553 samples. Each sample is represented by a horizontal line of pixels. colored pixels represent bases: G=green, C=blue, A=red, T=orange, missing=black. The data is separated by class (Table 5.1) SARS-CoV1: SARS-CoV, SARS-CoV P2, SARS-CoV HKU-39849 and SARS-CoV GDH-BJH01. For visualization purposes HCov-EMC and HCoV-4408 are not shown, given the number of examples. From a simple visual inspection, it is already possible to notice the similarity of the patterns between the classes.

Analyzing the different sequence values appearing in the max pooling feature space, a total of 3,827 unique 21-bps cDNA sequences are found, all potentially informative for identifying different virus strains. For example, sequence **AGG TAA CAA ACC AAC CAA CTT** is only found inside the SARS-CoV-2 class, in 59 out of 66 available samples. Sequence **CAC GAG TAA CTC GTC**

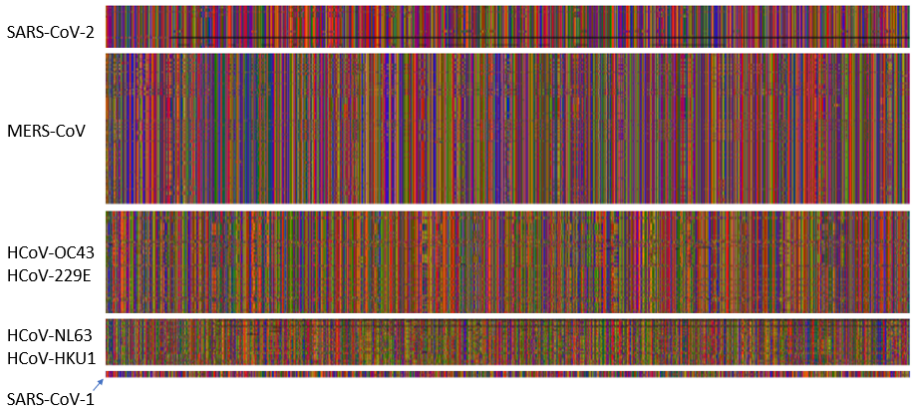


Figure 5.5: cDNA visualization for the first 105 out of 210 21-bps-long sequences selected from the input dataset. Each sample is represented by a horizontal line of pixels. Colored pixels represent bases: G=green, C=blue, A=red, T=orange, missing=black. The data is separated by class (Table 5.1) SARS-CoV1: SARS-CoV, SARS-CoV P2, SARS-CoV HKU-39849 and SARS-CoV GDH-BJH01. For visualization purposes HCov-EMC and HCoV-4408 are not shown, given the number of examples. From a simple visual inspection, it is already possible to notice the similarity of the patterns between the classes.

TAT CTT is present again only in SARS-CoV-2, in 63 out of the 66 samples.

The combination of the convolutional and max pooling layer allows the CNN to identify sequences even if they are slightly displaced in the genome (by up to 148 positions). Thus, a table of feature appearance of each of the sequences selected from the previous step is created. This results, in just a set of feature to differentiate SARS-CoV-2 from other viruses.

The experiments presented in the following subsections to validate the method have different objectives and make use of different datasets, all slowly building towards the final objective of designing primers for SARS-CoV-2. A summary of all the experiments and datasets used is shown in Fig. 5.6.

Experiment 1: Validation on the NGDC dataset

Starting from the dataset downloaded from the NGDC repository [Bei13] on March 15th 2020, repeated sequences are removed, and the procedure to translate the data into the sequence feature space is applied. The result is a frequency table of 3,827 features (21-bps sequences) with 583 samples (Table 5.1 (left)). Next, a state-of-the-art feature selection algorithm [Lop+19; Lop+20] is executed, reducing the sequences needed to identify different virus strain to the bare minimum. Remarkably, it is possible to correctly differentiate all the coronavirus (MERS-CoV, SARS-CoV-2, SARS-CoV-1, etc) samples using only 53 of the original 3,827 sequences, obtaining a 100% accuracy in a 10-fold cross-validation with a simpler and more traditional classifier, such as Logistic Regression.

Experiment 2: Validation on the NCBI dataset

A dataset is downloaded from the NCBI repository [She+01] on March 15th 2020, with the following query: *gene="ORF1ab" AND host="homo sapiens" AND "complete genome"*. The query results in 407 non-repeated sequences (Table 5.1 (right)). This dataset, named NCBI-A, presents 68 sequences belonging to SARS-CoV-2. The procedure to translate the data into the set of sequence features is

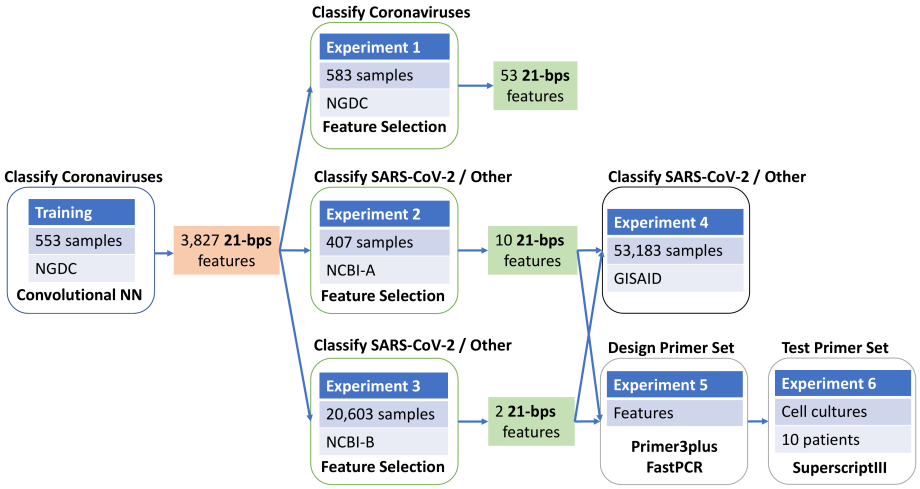


Figure 5.6: Summary of the different experiments, and corresponding datasets used.

then applied, and the same state-of-the-art feature selection algorithm of the previous experiment is applied. The result is a list of 10 different sequences (Table 5.2), with a remarkable property: just checking for their presence is enough to differentiate between SARS-CoV-2 and other viruses in the dataset, with a 100% accuracy. Each of the sequences, in fact, only appears in SARS-CoV-2 samples.

Table 5.2: Sequences that only exist in SARS-CoV-2, that help differentiate between the virus and other taxa.

TAG CAC TCT CCA AGG GTG TTC
CAT CTA CTG ATT GGA CTA GCT
AAT GAA TTA TCA AGT TAA TGG
CAC GTA GGA ATG TGG CAA CTT
TGA GCA GTG CTG ACT CAA CTC
CAA CTT TTA ACG TAC CAA TGG
CTA AAG CAT ACA ATG TAA CAC
GAT GGT CAA GTA GAC TTA TTT
TGC CAC TTG GCT ATG TAA CAC
TAT TAG TGA TAT GTA CGA CCC

Experiment 3: Further validation on the NCBI dataset

On March 17th 2020, the NCBI repository [She+01] is queried with: *"virus" AND host="homo sapiens" AND "complete genome"*, restricting the size from 1,000 to 35,000 bps. The query returns 20,603 samples, of which only 32 belong to SARS-CoV-2, and 20,571 are from other taxa, including Hepatitis B, Dengue, Human immunodeficiency, Human orthopneumovirus, Enterovirus A, Hepacivirus C, Chikungunya virus, Zaire ebolavirus, Human respirovirus 3, Orthohepevirus

A, Norovirus GII, Hepatitis delta virus, Mumps rubulavirus, Enterovirus D, Zika virus, Measles morbillivirus, Enterovirus C, Human T-cell leukemia virus type I, Yellow fever virus, Adeno-associated virus, rhinovirus (A, B and C), for a total of more than 900 viruses. This dataset is named NCBI-B. Then,the procedure to translate the data into the sequence feature space is applied, and the feature selection algorithm is executed. This results in 2 sequences of 21 bps: just by checking for their presence, it is possible to separate SARS-CoV-2 from the rest of the samples with a 100% accuracy. The sequences are: **AAT AGA AGA ATT ATT CTA TTC** and **CGA TAA CAA CTT CTG TGG CCC**.

Experiment 4: Validation on the GISAID dataset

The Global Initiative for Sharing All Influence Data (GISAID) repository [SM17] on August 10th, 2020, had 53,183 sequences available for SARS-CoV-2, from different countries. From those, 52,645 have as $< 1\%$ Ns (low percentage of missing values), high coverage and *host="homo sapiens"*. Starting from the 21-bps sequences obtained from experiments 2 and 3, a frequency table is computed, to verify whether the sequences also appear in other datasets and could then be used for detection. The frequency of appearance of the target sequences among the samples in the GISAID dataset is reported in Table 5.3, second column. In addition, 26 sequences from other hosts (*manis javanica*, *rhinolophus affinis*, *canine* and *felis catus*) are also obtained from the GISAID repository, to make a comparison with the sequences from experiment 2 and 3.

Table 5.3: Percentage of appearance for each of the 12 discovered 21-bps sequences across the different datasets, and comparison to similar viruses in nature and other hosts.

Source	GISAID	NCBI	NCBI	NGDC	NGDC	GISAID	GISAID	GISAID	GISAID
Virus	SARS-CoV-2	Other Taxa	SARS-CoV-2	Other Taxa	SARS-CoV-2	Betacoronavirus	Betacoronavirus	Betacoronavirus	Betacoronavirus
Host	Homo Sapiens	Homo Sapiens	Homo Sapiens	Homo Sapiens	Homo Sapiens	Manis javanica	Rhinolophus affinis	Canine	Felis Catus
# Samples	52,645	20,572	32	487	96	17	1	2	6
CAC GTA GGA ATG TGG CAA CTT	99.84%	0.00%	100.00%	0.00%	97.92%	0.00%	100.00%	100.00%	50.00%
TAT TAG TGA TAT GTA CGA CCC	99.73%	0.00%	100.00%	0.00%	97.92%	0.00%	0.00%	100.00%	50.00%
AAT GAA TTA TCA AGT TAA TGG	99.94%	0.00%	100.00%	0.00%	96.88%	76.47%	0.00%	100.00%	66.67%
AAT AGA AGA ATT ATT CTA TTC	99.73%	0.00%	100.00%	0.00%	96.88%	0.00%	100.00%	100.00%	66.67%
CAA CTT TTA ACG TAC CAA TGG	99.55%	0.00%	100.00%	0.00%	97.92%	0.00%	0.00%	100.00%	50.00%
CTA AAG CAT ACA ATG TAA CAC	99.76%	0.00%	100.00%	0.00%	100.00%	0.00%	0.00%	100.00%	66.67%
TAG CAC TCT CCA AGG GTG TTC	99.57%	0.00%	100.00%	0.00%	97.92%	0.00%	0.00%	100.00%	66.67%
CGA TAA CAA CTT CTG TGG CCC	99.06%	0.00%	100.00%	0.00%	97.92%	0.00%	100.00%	50.00%	50.00%
TGC CAC TTG GCT ATG TAA CAC	99.90%	0.00%	100.00%	0.00%	97.92%	0.00%	100.00%	100.00%	66.67%
CAT CTA CTG ATT GGA CTA GCT	99.79%	0.00%	100.00%	0.00%	97.92%	0.00%	100.00%	100.00%	50.00%
TGA GCA GTG CTG ACT CAA CTC	99.56%	0.00%	100.00%	0.00%	98.96%	0.00%	0.00%	100.00%	66.67%
GAT GGT CAA GTA GAC TTA TTT	99.69%	0.00%	100.00%	0.00%	96.88%	0.00%	0.00%	100.00%	66.67%

The results remarkably outperform earlier publications using machine learning for identifying SARS-CoV-2 (see for example [Ran+20]), with the added benefit of producing human-readable results instead of a plain black box classifier. Not to mention, the 21-bps sequences identified by the experiments can straightforwardly be used as primers.

Experiment 5: Design of the candidate primer set.

After the analysis carried out on the deep learning model, a test with Primer3plus [Unt+07] is performed, to see which of the sequences could be used as a forward primer, using sample NCBI NC045512.2 as the reference SARS-CoV-2 sequence. Primer3plus is a software able to predict the suitability of candidate primers. Sequence **TAG CAC TCT CCA AGG GTG TTC** is predicted to be suitable, showing a frequency of appearance of 99.57% in viral genomes available from different countries in GISAID [SM17] and 100.0% on the NCBI-A and NCBI-B [She+01] datasets. Using the reference SARS-CoV-2 sequence, this discovered sequence is located between nucleotides 25,604 and 25,624 in the ORF3a gene. In SARS-CoV, this gene encodes a protein of 274 aa, that is related with necrotic cell death [Shi+19], chemokine production like interleukin 8 (IL-8) and RANTES/CCL5, NFκB activation resulting in an inflammatory response [Kan+06] and may play an important role in the virus' life cycle [Pad+07]. A specific primer set for detection of SARS-

CoV-2 is designed using Primer3plus: **TAG CAC TCT CCA AGG GTG TTC** is used as forward primer and **GCA AAG CCA AAG CCT CAT TA** as reverse primer, obtaining an amplicon size of 179 bps. An *in-silico PCR* test using FastPCR 6.7 [KLS+09] with default parameters is then run, using NC045512.2 as a reference SARS-CoV-2 sequence. This yields positive results, with a melting temperature $Tm = 56.2°C$ for the forward primer, $Tm = 53.1°C$ for the reverse primer and $Ta = 58°C$.

In order to compare the primer automatically designed with the proposed approach, the frequency of appearance of different primers sets' sequences used in SARS-CoV-2 RT-qPCR tests developed by WHO referral laboratories is computed on the 52,645 sequences from the GISAID repository, and the 583 samples of different coronaviruses from the NGDC dataset from experiment 1. The used primers set are developed by University of Hong Kong (HKU-N); Charite, Berlin, Germany (Charite-E); US-CDC, United States (US-CDC-N1, US-CDC-N2, US-CDC-N3) and China CDC, China (China-CDC-ORF1ab, China-CDC-N) (Table 5.4). These primers are the ones more commonly used, as stated in the GISAID status update of August 11, 2020 [2].

From the results in Table 5.4, it is visible how all sequences have a frequency of appearance of $> 99\%$, with the exception of CHINA-CDC-N-F with a 68.52%. This is consistent with the percentage of genomes with mutation in the primer region in the GISAID update summary of August 11, 2020. In the in-silico analysis of specificity , all the primers sets' sequences are compared with samples from the NCBI-B and NGDC datasets: the results show that HKU-N-F, HKU-N-R, Charite-E-F, Charite-E-R and US-CDC-N2-F are not specific to SARS-CoV-2, as these primers bind to SARS-CoV-1 too. The rest of the sequences, including the automatically designed primers, only appear in SARS-CoV-2. Thus, in summary from 8 different primer sets, 3 of them are not specific to SARS-CoV-2, and from the remaining 5, considering frequency of appearance only, the design obtained with the proposed approach appears to be the third best option, when evaluating the worst frequency between forward and reverse primer.

Table 5.4: Frequency comparison for different sequences in primer sets suggested at the GISAID repository. The primers uncovered are marked with UtrechtU, as they were the result of a work in collaboration with Utrecht University.

Primer	Sequence	Frequency
HKU-N-F	5'-TAA TCA GAC AAG GAA CTG ATT A-3'	99.56%
HKU-N-R	5'-CGA AGG TGT GAC TTC CAT G-3'	99.58%
Charite-E-F	5'-ACA GGT ACG TTA ATA GTT AAT AGC GT-3'	99.90%
Charite-E-R	5'-ATA TTG CAG CAG TAC GCA CAC A-3'	99.90%
US-CDC-N1-F	5'-GAC CCC AAA ATC AGC GAA AT-3'	99.71%
US-CDC-N1-R	5'-TCT GGT TAC TGC CAG TTG AAT CTG-3'	99.57%
US-CDC-N2-F	5'-TTA CAA ACA TTG GCC GCA AA-3'	99.43%
US-CDC-N2-R	5'-GCG CGA CAT TCC GAA GAA-3'	99.74%
US-CDC-N3-F	5'-GGG AGC CTT GAA TAC ACC AAA A-3'	99.09%
US-CDC-N3-R	5'-TGT AGC ACG ATT GCA GCA TTG-3'	99.72%
CHINA-CDC-ORF1ab-F	5'-CCC TGT GGG TTT TAC ACT TAA-3'	99.90%
CHINA-CDC-ORF1ab-R	5'-ACG ATT GTG CAT CAG CTG A-3'	99.59%
CHINA-CDC-N-F	5'-GGG GAA CTT CTC CTG CTA GAA T-3'	68.52%
CHINA-CDC-N-R	5'-CAG ACA TTT TGC TCT CAA GCT G-3'	99.20%
UtrechtU-F	5'-TAG CAC TCT CCA AGG GTG TTC-3'	99.57%
UtrechtU-R	5'-GCA AAG CCA AAG CCT CAT TA-3'	99.48%

[2] `https://www.gisaid.org/hcov-19-analysis-update/`

Experiment 6: Validation of the candidate primer set in biological samples
To validate the primer obtained through an automatic in-silico procedure, a conventional PCR is performed on cDNA obtained from RNA from SARS-CoV-2 and other human coronaviruses. In addition, RNAs from nasopharyngeal swabs from six patients previously diagnosed with SARS-CoV-2 infection and four patients negative for SARS-CoV-2 by routine diagnostic method [Cor+20] are analyzed with the same conventional PCR. The RNA was converted into cDNA using SuperscriptIII (Thermo-Fisher Scientific, USA) and random hexamers. Subsequently, conventional PCR was performed on the cDNA using HotStar Taq DNA polymerase (Qiagen, The Netherlands) with 400nM forward primer (5'-**AG CAC TCT CCA AGG GTG TTC**-3') and 400nM reverse primer (5'-**GCA AAG CCA AAG CCT CAT TA**- 3') and the following cycling conditions: 15 min at 95°C, followed by 40 cycles of 1 min. at 95°C , 1 min. at 5 °CC and 1 min. at 72°C. The PCR products were visualized by electrophoresis. The same RNA was used in a diagnostics reference assay by Corman et al. [Cor+20] and the Cycle threshold values form this reference assay were used for estimating sensitivity.

Different dilutions of SARS-CoV-2 RNA are detected with similar sensitivity compared to the diagnostic reference assay, see Fig. 5.7 lanes 1-8. The candidate primer set created with the proposed approach exclusively detects SARS-CoV-2 and does not amplify RNA from other human coronaviruses, as shown in Fig. 5.7, lanes 9-14. The candidate primer set is able to detect SARS-CoV-2 RNA in patient samples previously found positive for SARS-CoV-2, but not in patients previously found negative, as displayed in Fig. 5.7, lanes 15-24. Although further validation will be required to develop this candidate primer set into a diagnostic assay, the results clearly demonstrate the power of the proposed method to select potential sequences for further validation.

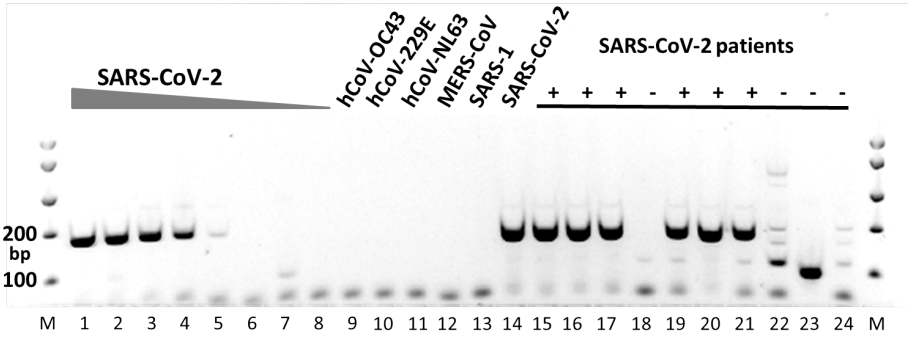


Figure 5.7: Laboratory validation of the candidate primer set by conventional PCR. MM, molecular weight marker; Lanes 1-8, 10-fold dilutions of SARS-CoV-2 RNA (corresponding to Ct values 26 to 39 in the diagnostic reference assay); Lanes 9-14, RNA from different human coronaviruses (hCoV-OC43, hCoV-229E, hCoV-NL63, MERS-CoV, SARS-1, SARS-CoV-2 respectively); Lanes 15, 16, 17, 19, 20, 21, patient samples previously found positive for SARS-CoV-2; Lanes 18, 22, 23, 24, patient samples previously found negative for SARS-CoV-2.

5.1.3 Evolutionary algorithms

While the CNN-based approach presented in section 5.1.2 proved to be effective, such technique presents a few important limitations: primers have a series of desired features, such as melting

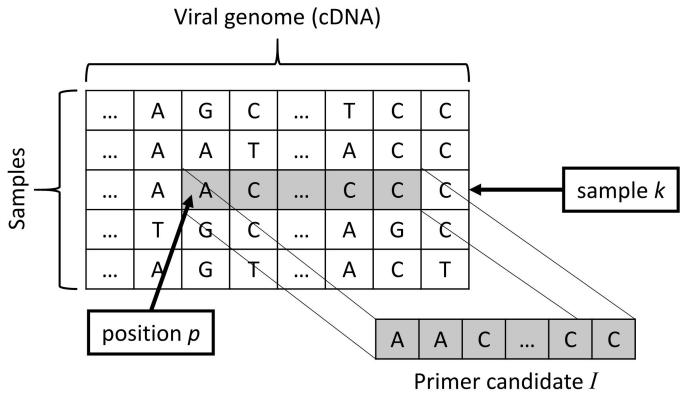


Figure 5.8: Subsequence I, representing a primer candidate, is uniquely identified by two integer values: index of the sample k in the training set, and position inside its genome p.

temperature T_m in a given range, presence of certain base pairs in specific positions, etc. that cannot be easily expressed as constraints in the procedure. A stochastic optimization algorithm, such as an EA, on the other hand, could include all this information in the fitness function. In the following sections, an EA is tested on the task of finding suitable primers for a SARS-CoV-2 variant.

Individual representation

An individual in the proposed approach is a primer candidate, thus a sub-sequence of a given virus sample. Assuming N samples of the target virus are available, and each sample is a sequence of L bps, an individual will be described by two integers, sample number k in the training dataset, and position number p inside the sample. The candidate primer will then be represented by the 21 bps in positions $\{p, p+1, ..., p+20\}$ of sample s_k, see Figure 5.8. While primers can be of any length between 18 and 22 bps, size 21 is selected to compare the final results with the primers produced by DL techniques [Lop+21a; Lop+21b], that are of length 21.

Population initialization

The population is initialized with random individuals. More specifically, the i-th individual will be characterized by two integers, k_i drawn with uniform probability in $(1, N)$ and p_i drawn with uniform probability from $(1, L-21)$.

Fitness function

The fitness function is a weighted sum, attempting to take into account all the criteria for a good primer candidate. The first term of the sum, F, is evaluating the presence of the sequence selected as candidate primer I inside training samples labeled with the target virus strain, and its absence from samples with a different label. In formal terms:

$$\mathscr{P}(I) = \sum_{i=0}^{T} P(I, s_i) \tag{5.3}$$

where T is the number of samples in the training set, s_i is the i-th sample in the training set, and function P is defined as:

$$P(I,s_i) = \begin{cases} 0, & \text{if } I \text{ is found inside } s_i \text{ and } L(s_i) == L(s_k) \\ 1, & \text{otherwise.} \end{cases} \tag{5.4}$$

where $L(s)$ returns the class label of sample s. In other words, $P(I,s_i)$ equals 1 if sequence I is found inside a sample with the same class label as sample s_k, the origin of sequence I. So, if the candidate primer I is found inside a sample that does not belong to the target class, or is not found in a sample that belongs to the target class, the solution is penalized.

The second term of the weighted sum takes into account the GC content of the candidate primer, or in other words, the presence of bases G and C:

$$\mathscr{C}(I) = 0.5 - \sum_{i=0}^{21} \frac{C(I(i))}{21} \quad \text{where } C(b) = \begin{cases} 1, & \text{if base } b \text{ is C or G} \\ 0, & \text{otherwise.} \end{cases} \tag{5.5}$$

where $I(i)$ represents the base in position i inside sequence I. The following element of the weighted sum is $\mathscr{N}$, defined as:

$$\mathscr{N}(\mathscr{I}) = \sum_{i=0}^{21} N(I(i)) \quad \text{where} \quad N(b) = \begin{cases} 1, & \text{if base } b \text{ is N} \\ 0, & \text{otherwise.} \end{cases} \tag{5.6}$$

that takes into account the presence of N symbols in the sequence, indicating an error in the read. The ideal primer candidate should only contain A, C, G, or T values.

The final term tackles the requirement of having a melting temperature T_m centered around 60°. Specialized literature [Unt+07] provides an equation to compute T_m for a sequence I:

$$T_m(I) = 81.5 + 16.6 * log_{10}([Na+]) + 41 * \mathscr{C}(I) - 600/l(I) \tag{5.7}$$

where $\mathscr{C}(I)$ is the content of C and G bases in sequence I, as described in Equation 5.5, $[Na+]$ is the molar sodium concentration, and $l(I)$ is the length of sequence I, in bps. The value of $[Na+] = 0.2$ is used, as described in [Unt+07], while $l(I) = 21$ by design. The term taking into account T_m will then be:

$$\mathscr{T}(I) = |60 - T_m(I)| \tag{5.8}$$

To summarize, the final weighted sum will be:

$$\mathscr{F}(I) = w_p \cdot \mathscr{P}(I) + w_c \cdot \mathscr{C}(I) + w_n \cdot \mathscr{N}(I) + w_t \cdot \mathscr{T}(I) \tag{5.9}$$

with w_p, w_c, w_n, w_t representing the weights associated to each term. Function $\mathscr{F}$ is to be minimized.

Selection

Individual selection for reproduction is performed through a tournament selection of size τ, where τ individuals are randomly drawn from the population with uniform probability, their fitness is compared, and the individual with the best fitness is ultimately chosen.

Evolutionary operators

Mutations operators can act on the two integer values k, representing the index of a sample in the training set, and p, representing a position inside sample k's genome. Mutations acting on p draw from a multinomial distribution where the probabilities are calculated from a normal distribution, following the idea that small displacements of the beginning of a primer left or right in the genome might provide small changes in the fitness function. Mutations targeting k, on the other hand, draw uniformly from all available sample indexes: most genomes of a virus, resulting from a sequencing process, will be almost aligned with each other, with small differences resulting from additions or deletions in the cDNA; for this reason, it is conceivable that the principle of locality will be preserved when creating a primer candidate from position p of two different samples k and k'. The second evolutionary operator adopted in the experiments is a classic one-point crossover.

Replacement

Individual replacement is carried out through a (μ, λ) scheme, where the entire population is replaced at each generation.

5.1.4 Experimental evaluation of the evolutionary algorithm

The proposed approach is validated on real-world data, using samples collected from the GISAID repository. 10,712 SARS-CoV-2 sequences are downloaded on December 23rd, 2020. After removing repeated sequences, a total of 2,104 sequences labeled as B.1.1.7, and 6,819 sequences from other variants are left, for a total of 8,923 samples. B.1.1.7 variants are assigned class label 1, while all the remaining samples are labeled as class 0. Then, the data are divided into 8,030 samples for training and 893 for testing.

The algorithm described in section 5.1.3 has been implemented in C#, and is available, along with all the data, in a GitHub open repository[3]. After a few preliminary trials to tune the EA's parameters, population size is set to $\mu = 100$, offspring size to $\lambda = 100$, and a termination condition based on a maximum number of generations, 20, probability of mutation to $p_m = 0.15$ and probability of crossover to $p_c = 0.85$, tournament selection size $\tau = 2$, and a (μ, λ) scheme, where the whole population is replaced at every generation. For the fitness function, the values for the weights are set as $w_p = 1.0, w_c = 100.0, w_n = 1000.0, w_t = 1.0$, to provide the same numerical importance to different parts of the evaluation, with the exception of $\mathcal{N}(I)$, as primers containing N symbols are unlikely to be acceptable. In principle, the fitness function could also be structured as a lexicographic comparison, first evaluating the presence of Ns in the sequence, and then proceeding with the rest of the evaluation. Using a combination of lexicographic acceptability and multi-objective evaluations could also be beneficial. These possibilities will be analyzed in future works.

In order to assess the stability of the results, the proposed approach is run 20 times, with each single run lasting around 62 minutes with 5 threads on a 64-bit Windows 10 laptop with Intel Xeon E-2186M. Thread parallelization is exploited to evaluate several individuals at the same time. The best individual of each run is then stored, to perform a later comparison with primers designed through other methods, discussed in Section 5.1.5.

The 20 runs of the proposed approach return different candidate primers, as shown in Table 5.5. Each of the resulting solutions is then simulated in the canonical version of B.1.1.7 in the GISAID sample *EPI_ISL_*601443 using Primer3Plus [Unt+07], to compute the reverse primer and perform a final assessment of its suitability as a primer. This last process is more computationally expensive

[3] https://github.com/albertotonda/ea-primers

than a simple evaluation of a fitness function, and it can last up to 20 minutes (Primer3Plus) depending of the software used to crosscheck.

From the 20 best individuals obtained, only 6 solutions can be used as forward primers. Further analysis on the sequences shows that the selected primers are in the region of 2 non-synonymous mutations (S982A, A570D) and 2 synonymous mutations (C15279T, C16176T), with the best candidate being the one based on mutation A570D (Table 5.6).

The results are then further validated on 487 samples of other coronaviruses, obtained from the National Genomics Data Center (NGDC) [Bei13]. All the candidate primers only target B.1.1.7 SARS-CoV-2, with no appearance in any other coronavirus sample. Further validation on 20,571 samples belonging to other non-corona viruses from the National Center for Biotechnology Information (NCBI) [She+01], shows no appearance of the sequence in any other virus sample, providing further support for specificity of the obtained best candidate primers.

Table 5.5: Candidate primers found during 20 runs of the proposed approach, and Primer3Plus output simulated on the canonical reference sequence *EPI_ISL_*601443.

Sequence	**Result**
GCACGTCTTGACAAAGTTGAG	GAGGTGCTGACTGAGGGAAG
TGGCAGAGACATTGATGACAC	Left primer is unacceptable: High end self complementarity
CAGAGACATTGATGACACTAC	Left primer is unacceptable: Tm too low
GGCAGAGACATTGATGACACT	Left primer is unacceptable: High end self complementarity
CCTCAAGGTATTGGGAACCTG	CATCACAACCTGGAGCATTG
TTGGCAGAGACATTGATGACA	AGCAACAGGGACTTCTGTGC
ACCTCAAGGTATTGGGAACCT	CATCACAACCTGGAGCATTG
CACACAACACATTTGTGTCTG	Left primer is unacceptable: Tm too low/High end self complementarity
TTCAGTGCATCGATATCGGTA	Left primer is unacceptable: High end self complementarity
CTCAGACTAATTCTCATCGGC	Left primer is unacceptable: Tm too low/High 3' stability
TCAGACTAATTCTCATCGGCG	Left primer is unacceptable: High 3' stability
TCAGACTAATTCTCATCGGCG	Left primer is unacceptable: High 3' stability
GGCAGAGACATTGATGACACT	Left primer is unacceptable: High end self complementarity
CAGACTAATTCTCATCGGCGG	Left primer is unacceptable: High 3' stability
GTGATGTAGAAAACCCTCATC	Left primer is unacceptable: Tm too low
GTGATGTAGAAAACCCTCATC	Left primer is unacceptable: Tm too low
CTCAGACTAATTCTCATCGGC	Left primer is unacceptable: Tm too low/High 3' stability
CTCAGACTAATTCTCATCGGC	Left primer is unacceptable: Tm too low/High 3' stability
CTCATCTTATGGGTTGGGATT	GCCACACATGACCATTTCAC
CCTTGCACGTCTTGACAAAGT	GAGGTGCTGACTGAGGGAAG

Table 5.6: Frequency of appearance in the training and test set, for each of the six candidate primers validated by Primer3Plus.

Candidate Primer	**Frequency in test set**	**Frequency in training set**	**Mutation**
GCACGTCTTGACAAAGTTGAG	0.9922	0.9893	T24506G (S982A)
CCTCAAGGTATTGGGAACCTG	0.9922	0.9889	C16176T
TTGGCAGAGACATTGATGACA	0.9922	0.9895	C23271A (A570D)
ACCTCAAGGTATTGGGAACCT	0.9922	0.9888	C16176T
CTCATCTTATGGGTTGGGATT	0.9910	0.9893	C15279T
CCTTGCACGTCTTGACAAAGT	0.9922	0.9890	T24506G (S982A)

5.1.5 Discussion

While the candidate primers identified by the proposed approach can be considered of high quality, it is worth it to compare them to other primer sets designed by conventional bioinformatics tools. A typical primer set would be represented by a sequence around a specific mutation, believed to be

exclusive to the target virus variant.

In order to perform this comparison, 21-bps sequences are generated around mutations N501Y, A570D, D614G, P681H, T716I, S982A, D1118H, with 10 bps before and after the position of the mutation, using the canonical sequence of the B.1.1.7 variant: for example, mutation N501Y (A23063T) will correspond to sequence **CCAACCCACT T ATGGTGTTGG**. The frequency of appearance of these human-designed primers is then tested against the best primers found by the EA during the experimental evaluation. The results are reported in Table 5.7.

Table 5.7: Frequency of appearance of the most significant mutations and the forward primer in the 8,293 sequences from the GISAID dataset.

Sequence	**B.1.1.7** **# samples (%)**	**Other Variants** **# samples (%)**
N501Y	1,985 (94.34%)	14 (0.02%)
A570D	2,013 (95.67%)	1 (< 0.01%)
D614G	2,096 (99.62%)	5,384 (78.96%)
P681H	2,014 (95.72%)	1 (< 0.01%)
T716I	2,005 (95.29%)	1 (< 0.01%)
S982A	2,008 (95.43%)	0 (0%)
D1118H	2,011 (95.57%)	0 (0%)
GCACGTCTTGACAAAGTTGAG	2,011 (95.57%)	0 (0%)
CCTCAAGGTATTGGGAACCTG	2,008 (95.43%)	0 (0%)
TTGGCAGAGACATTGATGACA	2,014 (95.72%)	1 (< 0.01%)
ACCTCAAGGTATTGGGAACCT	2,007 (95.38%)	0 (0%)
CTCATCTTATGGGTTGGGATT	2,012 (95.62%)	1 (<0.01%)
CCTTGCACGTCTTGACAAAGT	2,009 (95.48%)	0 (0%)
Total Samples	2,104	6,137

The 6 primer sets generated for the B.1.1.7 variant show almost the same frequency of appearance and similar specificity. Nevertheless, the sequence using mutation A570D is slightly better in frequency of appearance in the training set. The generated forward primers for the B.1.1.7 variant appear in 2,010 of 2,104 sequences, with an average frequency of 95.54%. A further analysis shows that only 2,014 of the sequences labeled as B.1.1.7 present 5 or more of the 7 studied mutations, which could point to an error in the annotation of the variant inside the GISAID dataset, or several extra mutations in the generated 21-bps sequences. If only sequences that show 5 or more of the mutations are considered as proper B.1.1.7 variant samples, the average of the generated primers correctly identifies 2,095 out of 2,104 samples, for a final 99.57% frequency of appearance.

While the primers generated by the proposed approach have a performance similar to primers obtained by conventional PCR design, it is extremely important to remark that this primers can only exist after the canonical sequence of a virus variant is defined. The canonical sequence represents the reference genome of a virus, and a considerable amount of effort from experts is necessary to reach a consensus on this sequence. The approach, like other ML techniques for primer design [Lop+21b] has the advantage of automatically finding suitable primers in a matter of hours; but differently from similar ML approaches, it is much faster (about an order of magnitude faster on comparable hardware, from the experiments reported in the previously cited publications) and can handle a wider variety of constraints describing the desired features of a candidate primer.

As of February 5th, other SARS-CoV-2 variants of concern have been identified and are on the rise across the globe, such as the one originating from Brazil (P.1) [DP21; Far21] and the one generated in South Africa (B.1.351) [Had+18; 20; Teg+20]. These new SARS-CoV-2 variants also carry the N501Y and D614G mutations, similarly to the B.1.1.7 variant. Thus, it is important to verify that the primers generated in this work are able to differentiate between the variants.

From the GISAID repository, all available 326 sequences of the B.1.351 variant are downloaded

on January 7, 2021 and all 28 non-repeated sequences of the P.1 variant are obtained on January 19, 2021. Next, the frequency of appearance of the primers in samples from other variants is verified. From Table 5.8, it is evident that while the primers found with the proposed approach are exclusive to the B.1.1.7 variant, other sequences built around mutations (such as the one built around N501Y) are often also found in different variants. thus negatively impacting their specificity.

Table 5.8: Frequency of appearance of the characteristic mutations for the UK (B.1.1.7), South African (B.1.351), and Brazilian (P.1) variants in B.1.1.7 sequences (2,104), B.1.351 sequences (337), P.1 sequences (28) and sequences of other variants (6,808).

Sequence	**Other**	**B.1.1.7**	**B.1351**	**P.1**
T1001I	0.01%	95.53%	0.00%	0.00%
A1708D	0.00%	91.83%	0.00%	0.00%
I2230T	0.01%	94.39%	0.00%	0.00%
N501Y	0.04%	94.34%	99.70%	82.14%
A570D	0.01%	95.67%	0.00%	0.00%
P681H	0.01%	95.72%	0.30%	0.00%
T716I	0.01%	95.29%	0.00%	0.00%
S982A	0.00%	95.44%	0.00%	0.00%
D1118H	0.00%	95.58%	0.00%	0.00%
Q27stop	0.04%	90.64%	0.30%	0.00%
R52I	0.00%	90.64%	0.00%	0.00%
Y73C	0.00%	90.78%	0.00%	0.00%
S235F	0.03%	95.58%	0.00%	0.00%
GCACGTCTTGACAAAGTTGAG	0.00%	95.58%	0.00%	0.00%
CCTCAAGGTATTGGGAACCTG	0.00%	95.44%	0.00%	0.00%
TTGGCAGAGACATTGATGACA	0.01%	95.72%	0.00%	0.00%
ACCTCAAGGTATTGGGAACCT	0.00%	95.39%	0.00%	0.00%
CTCATCTTATGGGTTGGGATT	0.03%	95.63%	0.00%	0.00%
CCTTGCACGTCTTGACAAAGT	0.00%	95.48%	0.00%	0.00%

Finally, from a comparison between the frequency of appearance of reverse primers in UK samples (Table 5.9) and the frequency of appearance in other variants (Table 5.8), it is possible to conclude that the best candidates for a primer set are sequences **GCA CGT CTT GAC AAA GTT GAG** as forward primer, based on mutation S982A, and **GAG GTG CTG ACT GAG GGA AG** as reverse primer. While the results are encouraging, it is important to remember that an *in-silico* validation is not enough to provide a final answer. The next step in the process will be to test the primers in the lab.

Table 5.9: Frequency of appearance of the forward and reverse primers found by the EA algorithm.

Forward Primer	Frequency	Reverse Primer	Frequency	Average
GCACGTCTTGACAAAGTTGAG	95.58%	GAGGTGCTGACTGAGGGAAG	99.71%	97.65%
CCTCAAGGTATTGGGAACCTG	95.44%	CATCACAACCTGGAGCATTG	98.62%	97.03%
TTGGCAGAGACATTGATGACA	95.72%	AGCAACAGGGACTTCTGTGC	99.62%	97.67%
ACCTCAAGGTATTGGGAACCT	95.39%	CATCACAACCTGGAGCATTG	98.62%	97.01%
CTCATCTTATGGGTTGGGATT	95.63%	GCCACACATGACCATTTCAC	99.81%	97.72%
CCTTGCACGTCTTGACAAAGT	95.48%	GAGGTGCTGACTGAGGGAAG	99.71%	97.60%

5.1.6 Limitations

While both methods for the automatic discovery of primers proved to be effective, there is one important limitation that must be addressed. Framing the primer discovery as an optimization problem, at a first glance the search space of all possible primers seems vast. Taking for example a 21-bps fixed-length sequence, and 2,000 samples that have around 30,000 bps each, the total

number of candidate primers is $2 \cdot 10^3 \cdot (3 \cdot 10^4 - 21) \simeq 6 \cdot 10^7$: computing statistics such as melting temperature T_m and frequency of appearance in other samples for each candidate would probably take months, even with parallel evaluations.

Nevertheless, I recently discovered a fundamental mistake in the reasoning that motivated the use of the previously presented techniques. While it is true that the potential number of primer candidates is large, most of them are actually identical, as the genomic sequences of different samples of the same virus strain are usually extremely close to each other, give or take a few mutations, deletions, or additions. Performing a preliminary inventory of all possible candidates with the support of a hash table to identify unique sequences, reduces the number of candidates to be analyzed from 10^7 to 10^5, a quantity that can be treated in a few days of computational effort on a multi-core server.

This consideration appears to cast a shadow on the use of CNNs and EAs for these problems: after all, if the search space can be treated with an exhaustive analysis, there is not need to use such complex computational intelligence techniques. While this might be true for the specific application to SARS-CoV-2, it becomes immediately apparent that an exhaustive approach is unfeasible for larger viruses (comprising 1M bps) or bacteria (easily numbering more than 1.5M bps and presenting further challenges, such as difficulties in aligning reads to a reference and the presence of mobile elements). Not to mention, some of the techniques alternative to qT-PCR, like Loop-mediated isothermal amplification (LAMP) require more than two primer sequences, each one with specific size and distance from the others,

In conclusion, even if the techniques developed for this specific challenge might be superseded by more traditional approaches, the experimental results open exiting perspectives for the use on computational intelligence on more complex genomic problems.

5.2 Signatures for Cancer Classification

MicroRNAs (miRNA) are small noncoding RNA molecules that can be detected in bodily fluids without the need for major invasive procedures on patients. miRNAs have shown great promise as biomarkers for tumors, to both assess their presence and predict their type and subtype. Recently, thanks to the availability of miRNAs datasets, machine learning techniques have been successfully applied to tumor classification. Results, however, are difficult to assess and interpret by medical experts because the algorithms exploit information from thousands of miRNAs.

In this section, I describe a research line for a novel methodology that aims at reducing the necessary information to the smallest possible set of circulating miRNAs, exploiting feature selection techniques. The dimensionality reduction achieved reflects a very important first step in a potential, clinically actionable, miRNA-based precision medicine pipeline.

5.2.1 Biomarkers for cancer

Cancer is difficult to diagnose and classify at early stages, and is one of the top leading causes of death worldwide [Fer+15]. Therefore, several attempts have been made to identify possible biomarkers for cancer detection. MicroRNAs (miRNAs) represent a class of small noncoding RNA molecules, with a critical role in the post-transcriptional regulation of gene expression. miRNAs also act on several cellular processes, such as cell differentiation, cell cycle progression, and apoptosis. Additionally, in tumors, some miRNAs can function as oncogenes, while others suppress tumors [TOB11].

Succeeding the earliest evidence of miRNA involvement in human cancer by Croce et al. [Cal+02], various studies have demonstrated that miRNA expressions are deregulated in human cancer through a variety of mechanisms [PC16]. Since ectopic modulation of specific miRNAs compromise the

hallmarks of cancer, several efforts have been spent to generate scaffold-mediated miRNA-based delivery systems trying to demonstrate the potential of miRNA-mediated therapies.

In comparison to invasive methods currently used for cancer diagnosis, there is an ongoing debate on the use of circulating miRNAs as possible biomarkers due to the fact that they can be detected directly from biological fluids, such as blood, urine, saliva and pleural fluid [SP11]. miRNAs possess other qualities of good candidate biomarkers such as: a) they are useful for the identification of cancer types, b) their availability of high-quality measurement techniques for miRNAs and c) they present good conservation between practical and preclinical models [He+15].

Several studies have shown the properties of miRNAs as oncogenes and tumor suppressors genes [CLG13; Fab13; FVN10]. Since then, techniques such as microarray (Affymetrix, Agilent) and sequencing techniques (Illumina), have been proposed for their identification [LLC12]. In the context of increasing availability of data, it is of utmost practical importance to build databases of miRNA expressions data for cancer research [Akh+15; BZC12; KG10] and to extract features that could be used as cancer biomarkers [BT09; Cor+11; IC12]. For example, the expression levels of miRNA *hsa-miR-21* change for different cancer types such as: squamous cell lung carcinoma [Gao+11], astrocytoma [Zhi+10], breast cancer [Yan+08], and gastric cancer [Wan+15]. Following this idea, the scientific community is currently looking for miRNA signatures (a subset of miRNAs), representing the minimal number of miRNAs to be measured for discriminating between different stages and types of cancer.

Thousands of miRNAs have been identified, and currently miRBase (v22.1) contains 1,917 stem-loop sequences, and 2,657 mature sequences for human microRNA [KG10]. Although a classification of cancer tumor type is possible using isomirs [Tel+17], not all of the miRNAs listed are available in every study, and only a few of them have been shown to work as circulating biomarkers [He+15]. Obtaining a minimal list of miRNAs able to correctly classify tumors is of utmost practical importance, because it would reduce the measurements needed and improve the likelihood of validation across multiple studies.

Several approaches in the literature propose the use of machine learning techniques for feature selection involving miRNAs. For example, feature selection for identifying miRNA targets [YAK16], for prediction of specific biomarkers for tumor origin [Tan+17] and to learn subset of features for tumor classification [PPR17]. In this study, the objective was to use feature selection and to uncover a small miRNAs signature with the aim to correctly classify cancer tumor types, and distinguish between normal and tumor tissue reducing the necessary features by an order of magnitude.

5.2.2 Ensemble feature selection

As the objective is to discover and validate a reduced list of miRNAs to be used as a signature for tumor classification, features that could optimally assist in distinguishing between different cancer types and tumor tissue need to be selected. In this sense, popular approaches used for feature selection range from univariate statistical considerations, to iterated runs of the same classifier with a progressively reduced number of features in order to assess the contribution of the features to the overall result. As the considered problem is particularly complex, relying upon simple statistical analyses might not suffice. Furthermore, features extracted using an iterative method on one classifier are likely to work well only for that specific classifier. Following the idea behind *ensemble feature selection* [Abe+09; SAV08; Sei+17], the use of multiple algorithms is proposed, to obtain a more robust and general predictive performance. An ensemble approach has the advantage of obtaining features that will be effective across several classifiers, with a better likelihood of being more

representative of the data, and not just of the inner workings of a single classifier.

For this purpose, a set of classifiers is trained to extract a sorted list of the most relevant features from each. Intuitively, as a feature considered important by the majority of classifiers in the set is also likely to be relevant for our objective, then information from all classifiers is compiled to find the most common relevant features. Starting from a comparison of 22 different state-of-the-art classifiers on the considered dataset, presented in [Rin+18], a subset of those classifiers was selected considering both; high accuracy and a way to extract the relative importance of the features from the trained classifier. After preliminary tests to set algorithms' hyperparameters, 8 classifiers were chosen, all featuring an average accuracy higher than 90% on a 10-fold cross-validation: `Bagging` [Bre99], `Gradient Boosting` [Fri01], `Logistic Regression` [Cox58], `Passive Aggressive` [Cra+06], `Random Forest` [Bre01], `Ridge` [Tik43], `SGD` (Stochastic Gradient Descent on linear models) [Zha04], `SVC` (Support Vector Machines Classifier with a linear kernel) [Hea+98]. All considered classifiers are implemented in the `scikit-learn` Python toolbox.

Overall, the selected classifiers fall into two broad typologies: those exploiting ensembles of classification trees [Bre+84] (`Bagging`, `Gradient Boosting`, `Random Forest`), and those optimizing the coefficients of linear models to separate classes (`Logistic Regression`, `Passive Aggressive`, `Ridge`, `SGD`, `SVC`). Depending on classifier typology, there are two different ways of extracting relative feature importance. For classifiers based on classification trees, the features used in the splits are counted and sorted by frequency, from the most to the least common. For classifiers based on linear models, the values of the coefficients associated to each feature can be used as a proxy of their relative importance, sorting coefficients from the largest to the smallest in absolute value. As the two feature extraction methods return heterogeneous numeric values, only the relative sorting of features provided by each classifier was considered. Furthermore, it is decided to extract the top 100 most relevant features as a reduction of about an order of magnitude, so each feature f is assigned a simple score $s_f = N_t/N_c$, where N_t is the number of times that specific feature appears among the top 100 of a specific classifier instance, while N_c is the total number of classifiers instances used; for instance, a feature appearing among the 100 most relevant in 73% of the classifiers used would obtain a score $s_f = 0.73$. 100 features are selected, because in this way the dataset is compressed at by least 90%, from 1,046 to 100 features. In order to increase the generality of the results, each selected classifier was run 10 times, using a 10-fold stratified cross-validation, so that each fold preserves the percentage of samples of each class in the original dataset. Thus, $N_c = 80$ (8 types of classifiers, run 10 times each). The complete procedure is summarized by Algorithm 2. Different approaches to the aggregation of heterogeneous feature importance from various sources are also possible (see for example [Abe+09; SAV08; Sei+17]), such as assigning to each feature a weight proportional to its relative importance. However, most alternatives would require adding and tuning extra parameters, so it is decided to opt for a simpler approach.

5.2.3 Experiment 1

An ensemble feature selection method is applied to a subset of The Cancer Genome Atlas dataset (TCGA) [Wei+13], containing 8,023 cases, with 28 different types of cancer, and 1,046 different stem-loop miRNA expressions (miRBase V16 [4], summarized in Table 5.10). Typically, classifiers trained on a dataset do not use the whole set of available features to separate classes, but only a subset which could be ordered by relative importance, with a different meaning given to the list by the specific technique, pushing for simpler models. Using 8 state-of-the-art classifiers implemented in

[4] ftp://mirbase.org/pub/mirbase/16/

Algorithm 2: Ensemble feature selection.

Normalize dataset on each of the F features, Divide dataset in N folds, Select K classifiers, Choose the number of features in the signature S;
for *each fold n of N* **do**
    for *each classifier k of K* **do**
        Train classifier k_n on all folds minus n, using all features;
        Test classifier k_n on fold n;
        Obtain sorted list l_{kn} of features from k_n;
        Assign weight w_{fnk} to each f of the F features;
        for *each feature f of F* **do**
            if *f is among the top S features in* l_{kn} **then**
                $w_{fnk} = 1$
            else
                $w_{fnk} = 0$
$N_c = N \cdot K$;
for *each miRNA feature f* **do**
    $N_t = \sum_n^N \sum_k^K w_{fnk}$;
    $s_f = N_t / N_c$;
Select S-feature signature, from features with highest s_f;
for *each fold n of N* **do**
    for *each classifier k of K* **do**
        Train classifier k_n on all folds minus n, using signature;
        Test classifier k_n on fold n;
Compare performance of classifiers using all features and signature;

the `scikit-learn` toolbox [Ped+11], the most relevant miRNAs are extracted to be used as features for cancer classification. The top k features in the list are then evaluated as a potential reduced signature for classification. In this work, after preliminary tests, $k = 100$ is selected to reduce the original features by an order of magnitude. Because other feature selection methods require the user to specify a desired number of features, this also allows for a fair and meaningful comparison with these methods.

The obtained 100-miRNA signature is first tested to classify the initial TCGA dataset, and later applied on 14 Gene Expression Omnibus (GEO) datasets obtained with different platforms (Affymetrix Multispecies Array miRNA-1, miRNA-2 and miRNA-3, Illumina 2000, and Agilent-021827 Human miRNA Microarray V3), for different cancer tumor types (Prostate, Liver, Breast, Esophageal, Head and Neck Squamous and Lung). A summary of this validation is presented in Fig. 5.9. Furthermore, the proposed methodology is compared to popular feature selection methods in bioinformatics, such as Univariate Feature Selection, Recursive Feature Elimination, Genetic Algorithms, Least Absolute Shrinkage and Selection Operator, Random Selection, Elastic Net and Ensemble Feature Selection with Complete Linear Aggregation. Next, the same signature is used to try to distinguish molecular subtypes in breast cancer, both for the TCGA dataset and a set of GEO datasets.

Finally, the 100 miRNAs included in the signature are evaluated through a meta-analysis based on the medical literature. Because this meta-analysis reveals known relationships between features selected by our approach, relative to the type of cancer considered, it has the potential to yield insight into the biological processes and relationships combinedly affecting miRNAs and cancer.

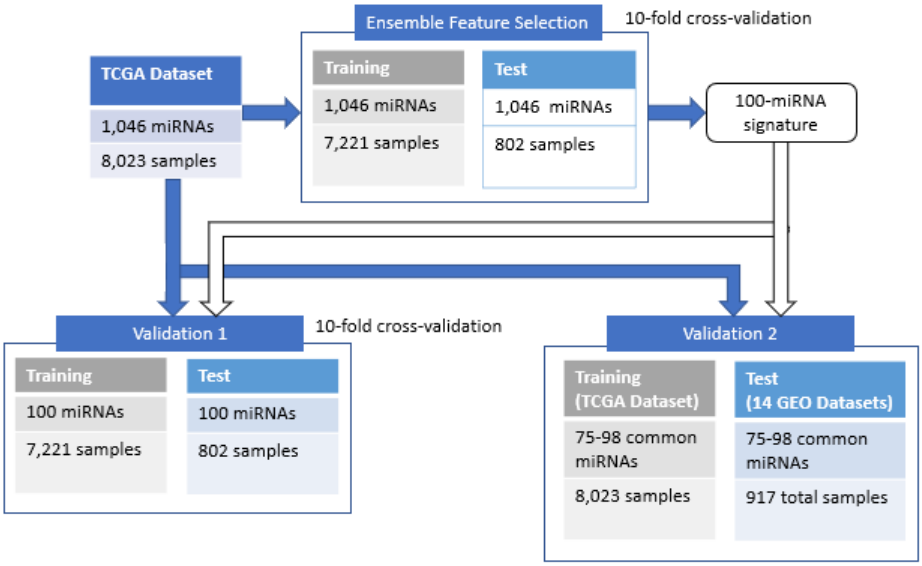


Figure 5.9: Summary of the different datasets and their use in the experiments.

TCGA Dataset

The data used in the experiments comes from the TCGA Data Portal[5], downloaded on September 1, 2016. The used data is miRNA-SEQ files (**.mirna.quantification.txt*) a total of 1,046 miRNA expression features for each sample in format mirbase V16 for stem-loop sequences [6]. The read per million (RPM) values in the file are considered, and all of the samples where the item does not meet the study protocol as stated in the *file annotations* are removed. In summary, the dataset used in the following experiments includes 28 types of tumors, 1,046 miRNA features, and 8,023 patient samples. Information on the dataset is summarized in Table 5.10. The data is normalized by removing the mean and scaling to unit variance: it is important to notice that the normalization is learned on the training set, and applied to the test set, so that knowledge of the whole dataset did not bias the performance on the test set. In addition, a second dataset that differentiates between normal tissue (NT) and tumor tissue (TT) is created, consisting of 8,657 samples; 8,023 TT and 634 NT.

Table 5.10: Summary of the TCGA dataset used in the study.

Tumor Type	Acronym	Tumor Tissue	Normal Tissue	Class
Adrenocortical carcinoma	ACC	80	0	0
Bladder Urothelial Carcinoma	BLCA	411	19	1
Breast invasive carcinoma	BRCA	777	87	2
Cervical squamous cell carcinoma	CESC	306	3	3
Cholangiocarcinoma	CHOL	36	9	4
Lymphoid Neoplasm Diffuse Large B-cell Lymphoma	DLBC	47	0	5
Esophageal carcinoma	ESCA	187	13	6
Head and Neck squamous cell carcinoma	HNSC	487	44	7
Kidney Chromophobe	KICH	66	25	8
Kidney renal clear cell carcinoma	KIRC	260	71	9
Kidney renal papillary cell carcinoma	KIRP	291	34	10
Lower Grade Glioma	LGG	528	0	11
Liver hepatocellular carcinoma	LIHC	374	50	12
Lung adenocarcinoma	LUAD	458	46	13
Lung squamous cell carcinoma	LUSC	341	45	14
Mesothelioma	MESO	86	0	15
Pancreatic adenocarcinoma	PAAD	154	4	16
Pheochromocytoma and Paraganglioma	PCPG	184	3	17
Prostate adenocarcinoma	PRAD	494	52	18
Sarcoma	SARC	260	0	19
Skin Cutaneous Melanoma	SKCM	450	2	20
Stomach adenocarcinoma	STAD	399	45	21
Testicular Germ Cell Tumors	TGCT	156	0	22
Thyroid carcinoma	THCA	513	59	23
Thymoma	THYM	124	2	24
Uterine Corpus Endometrial Carcinoma	UCEC	417	21	25
Uterine Carcinosarcoma	UCS	57	0	26
Uveal Melanoma	UVM	80	0	27
Total		**8,023**	**634**	

[5] https://tcga-data.nci.nih.gov/docs/publications/tcga/

[6] ftp://mirbase.org/pub/mirbase/16/genomes/hsa.gff

Geo Datasets

To validate the results, 14 datasets from the GEO repository [7] are used, from 5 different platforms. Two types of miRNA discovery technologies are used in the datasets: microarrays and sequencing. miRNAs expression levels are platform and technology dependent [BAB14; Del+13; Les+13]. Therefore, for each dataset is important to consider whether the information is in stem-loop or mature sequence and then calculate the contributions to make a direct comparison.

In the TCGA dataset, stem-loop sequences were directly measured in raw read counts. When reading a mature sequence, the protocol that was followed assigns a read count to it, and then randomly assigns a read count to one of the stem-loop sequences that share the same mature sequence [Chu+15].

GPL8786, GPL10850

Affymetrix Multispecies miRNA-1 Array (GPL8786) and Agilent-021827 Human miRNA Microarray V3 (GPL10850) cannot read stem-loop sequences, so the corresponding GEO datasets only show information for mature sequences. Thus, in order to perform a fair comparison, the raw read count for stem-loop sequences is defined as a linear function of the read counts of the mature sequences. If the read counts of a specific stem-loop sequence is termed X_i, for *hsa-mir-10b* the result will be:

$$X_{hsa-mir-10b} = a_0 \cdot X_{hsa-miR-10b} + a_1 \cdot X_{hsa-miR-10b*} \tag{5.10}$$

Where a_0 and a_1 are two coefficients to be set. The mapping between the values of two different platforms $P1$ and $P2$ can then be written as:

$$X^{P1}_{hsa-mir-10b} = a_2 \cdot X^{P2}_{hsa-mir-10b} \tag{5.11}$$

To reduce the problem, only relationships between a stem-loop sequence and its most common corresponding mature sequence e.g hsa-mir-10b to hsa-miR-10b, are considered, disregarding hsa-miR-10b*. From Eq. 5.10 and 5.11:

$$\begin{aligned}
X^{P1}_{hsa-mir-10b} &= a_2 \cdot X^{P2}_{hsa-mir-10b} \\
X^{P1}_{hsa-mir-10b} &= a_2 \cdot (a_0 \cdot X^{P2}_{hsa-miR-10b} + a_1 \cdot X^{P2}_{hsa-miR-10b*}) \\
X^{P1}_{hsa-mir-10b} &= a_2 \cdot a_0 \cdot X^{P2}_{hsa-miR-10b} \\
X^{P1}_{hsa-mir-10b} &= a^P_{hsa-miR-10b} \cdot X^{P2}_{hsa-miR-10b}
\end{aligned} \tag{5.12}$$

where a^P_i becomes the only coefficient to be found, and it represents the transformation between platforms for that specific sequence. A different linear function will be found for each pair of platforms, as it is assumed that each machine will have unique properties.

For GPL8786 GEO datasets, the linear gene expression values given by the function `rmasummary` from the Matlab bioinformatics toolbox are considered, which is a normalized robust multi-array average procedure, as a z-score [Che+03; Iri+03]. The equation of a z-score is:

[7] https://www.ncbi.nlm.nih.gov/gds

$$Z = \frac{(X - \mu)}{\sigma} \tag{5.13}$$

where X is the value of a feature; μ and σ are the average and the standard deviation for a feature. Next, by considering the linear expression values as z-scores, the GEO datasets are mapped to corresponding intensities in the TCGA dataset space, by solving for X:

$$X_i = (Z_i \cdot (\sigma_i^{TCGA}) + \mu_i^{TCGA}) \cdot a_i^P \tag{5.14}$$

where X_i is the intensity of miRNA i in the TCGA dataset space, Z_i is the linear gene expression value given by the scaled `rmasummary` summary function, μ_i^{TCGA} and σ_i^{TCGA} are the average value and the standard deviation for miRNA i, both computed on the original TCGA dataset, and a_i^P is a scale value, dependent on the platform. The value a_i^P is computed using a subset of all the GEO datasets from the same platform, by minimizing the error between actual class and predicted class, using a model trained in the TCGA dataset with Root Mean Squared Error (RMSE).

$$RMSE = \sqrt{\frac{\sum_{s=1}^{S} Predicted_s(TCGA, a^P) - Actual_s(TCGA)}{S}} \tag{5.15}$$

where S is the total number of samples in the dataset, and a^P is a vector containing the values of a_i^P for each feature i. A state-of-the-art numerical optimizer [HMK03] is applied to this task, to find the 98 parameters represented by a^P.

For GPL10850 the MatLab function `agferead` from the Bioinformatics Toolbox is appleid, and the value of `gTotalGeneSignal` is adopted for each of the probes to calculate the contributions and a_i^P as for GPL8786.

GPL14613, GPL16384

Affymetrix Multispecies miRNA-2 Array (GPL14613) and Affymetrix Multispecies miRNA-3 Array (GPL16384) measure the stem-loop sequences directly, and denote them by *hp_hsa*. The linear relationship between the TCGA dataset and the corresponding subset of GEO datasets is thus represented by Eq. 5.11, and the a_i^P parameters to be found are reduced to the a_{2i}

As remarked by Telonis et al [Tel+17], for these datasets, not all the types of cancer are available, or present the necessary quality standards. Thus, the analysis is limited to 6 different types of cancer; Prostate, Liver, Breast, Esophageal, Head and Neck Squamous Cell and Lung. For the sequencing data, extra mapping is not necessary besides the sample normalization (platform GPL11154), and only stem-loop sequences are used.

Using this procedure, it is possible to map the GEO repository measurements into the TCGA dataset space as seen in Fig. 5.10. Other examples are shown in Fig. 5.11, where plots were created using the first two dimensions of a Principal Component Analysis (PCA) computed on the TCGA dataset and applied to the GEO datasets, to provide a comparison between the cancer type in each GEO and the corresponding class in TCGA. Remarkably, samples from GEO datasets are often considerably close to samples of the corresponding class in TCGA. During validation, the common features between each GEO dataset and the 100-miRNA signature obtained using the ensemble approach are selected. The accuracy of the classification algorithms was then evaluated by training them on the TCGA dataset and testing them on each GEO dataset. A summary of the experiments is presented in Fig. 5.9.

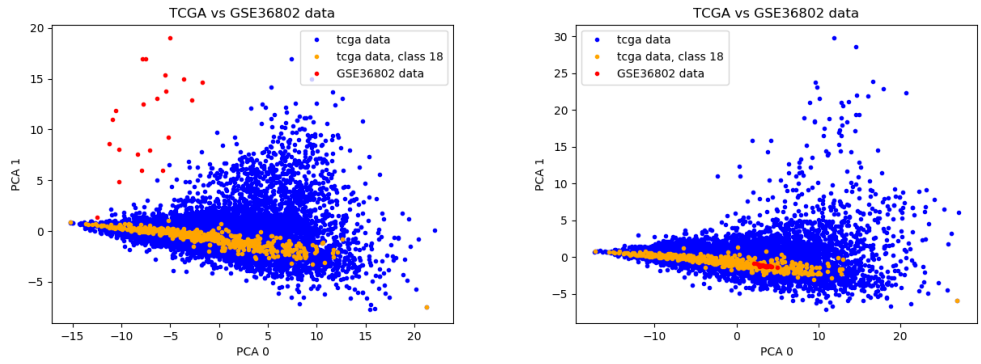


Figure 5.10: Example of mapping GSE microarray data into TCGA space (GSE36802).

Feature selection and validation on the TCGA dataset

Table 5.11 compares the classification accuracy on a 10-fold cross-validation for each classifier, using the full 1,046 features, and then employing the reduced 100-miRNA signature. It is interesting to notice how the accuracy is, for most cases, unchanged, providing empirical evidence that a 100-miRNA signature is enough to obtain good classification results, with a small statistically significant (T-test, $p < 0.05$) difference of 1.4%.

Table 5.11: Accuracy of classifiers used in the experiments on the TCGA dataset. In the case a classifier is not using standard values for its hyperparameters, the relevant variations are summarized in the corresponding column.

Classifier	Accuracy (10-fold CV)					
	1,046 Features		100 Features		Hyper Parameters	Feature Selection Method
	avg	std	avg	std		
Gradient Boosting	0.9398	0.0076	0.9359	0.0086	300 predictors	Decision Trees
Random Forest	0.9351	0.0071	0.9324	0.0073	300 predictors	Decision Trees
Logistic Regression	0.9178	0.0096	0.9237	0.0067	-	Coefficients
Passive Aggressive	0.9117	0.0104	0.8831	0.0115	-	Coefficients
SGD	0.91	0.0074	0.9035	0.0152	-	Coefficients
SVC	0.9211	0.0122	0.9154	0.0065	Linear kernel	Coefficients
Ridge	0.8971	0.0138	0.8305	0.0062	-	Coefficients
Bagging	0.9151	0.0120	0.9110	0.0077	300 predictors	Decision Trees
Average	0.918463	-	0.9044	-	-	-

Fig. 5.12 shows a heatmap comparing the relative frequency of the overall top 100 most frequent miRNA features, for each considered classifier. As expected, not all classifiers used the same features to separate the types of cancer, and thus, evaluating their consensus is more robust than just relying upon a single algorithm, as it is commonly accepted in the field of machine learning [AK17]. It is interesting to notice that while the most common biomarkers appear among the top for most classifier, others make use of only a few. For example, `Bagging` and `Ridge` do not use the vast majority of the features exploited by other techniques to discriminate between classes. A further difference between the two classifiers is that features used by `Bagging` that also appear in the top 100 are clearly important for the classifier, being used in almost 100% of its 10 runs; while it is noticeable

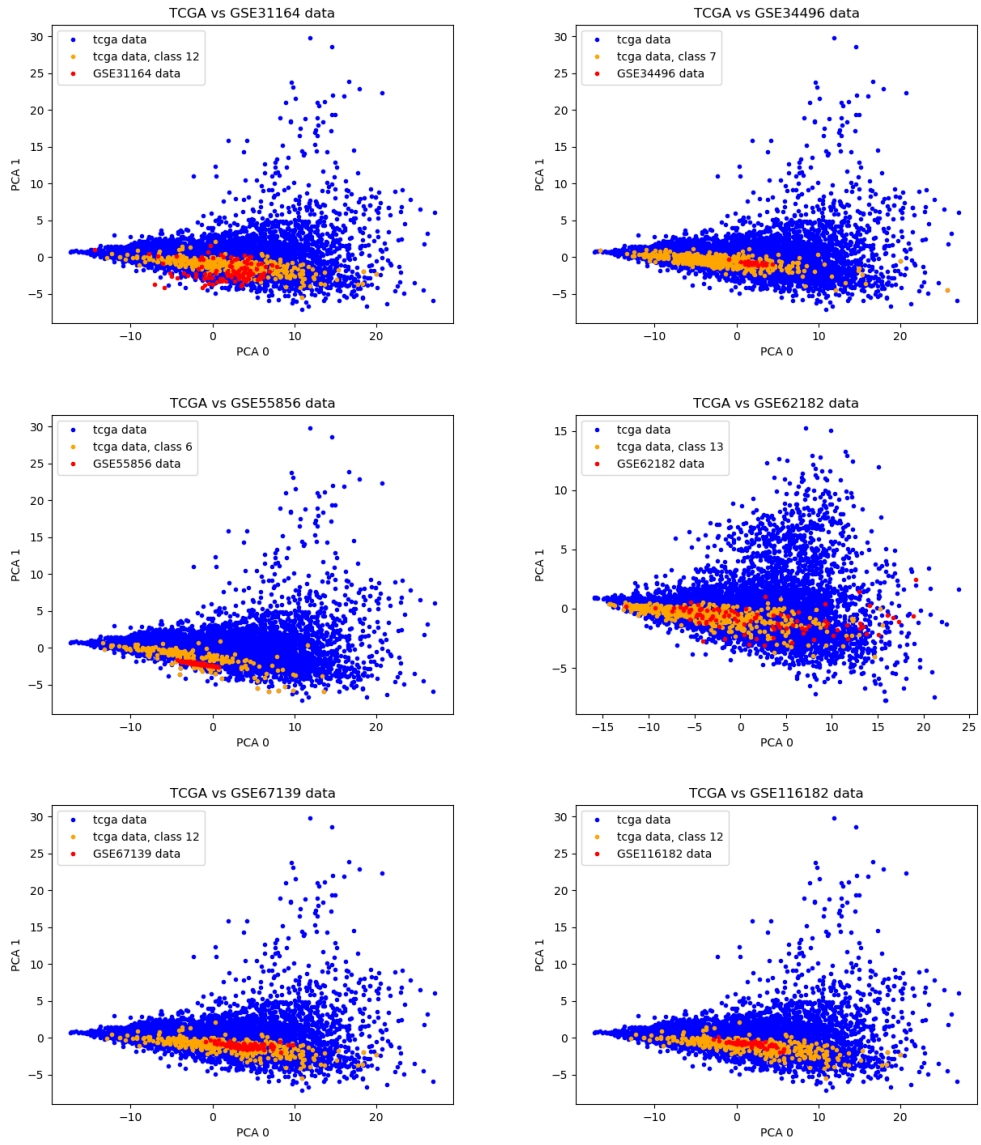


Figure 5.11: Examples of PCA projections of GEO datasets transformed into the TCGA dataset space. Orange data points represent samples from the target class from the TCGA dataset, the blue data points are other samples in TCGA, and the red points are the projected samples from GEO datasets.

how `Ridge` probably bases its discrimination on features that do not appear among the top 100. This would also explain why `Ridge` is the only algorithm that presents a decrease in performance when using the 100-miRNA signature. It's important to note that, while the results emerging from the heatmap suggest that this is indeed the case, `Ridge`'s decision boundaries should be analyzed more in-depth, for each class and multiple instances, in order to have absolute certainty, a task that is outside of the scope of the current work. Fig. 5.13 shows the difference between 1,046 features and 100 features for each cancer type and classifier.

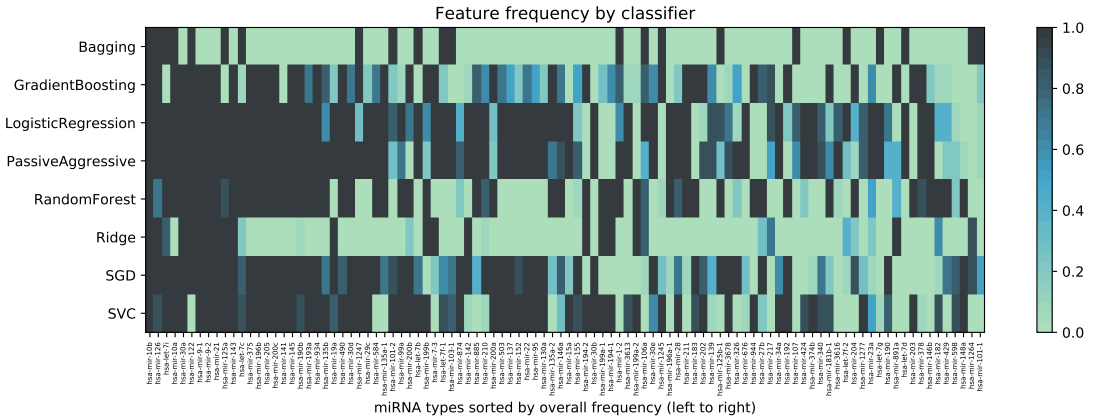


Figure 5.12: Heatmap with the frequency of the overall top 100 most frequent features, divided by classifier. Features are sorted from overall most to least frequent, from left to right, using information from the whole ensemble. For example, the most frequent is mir-10b, that is considered important by all classifiers. Color intensity is computed using information from instances of the same classifier, only. This shows the different importance that different classifiers assign to each feature.

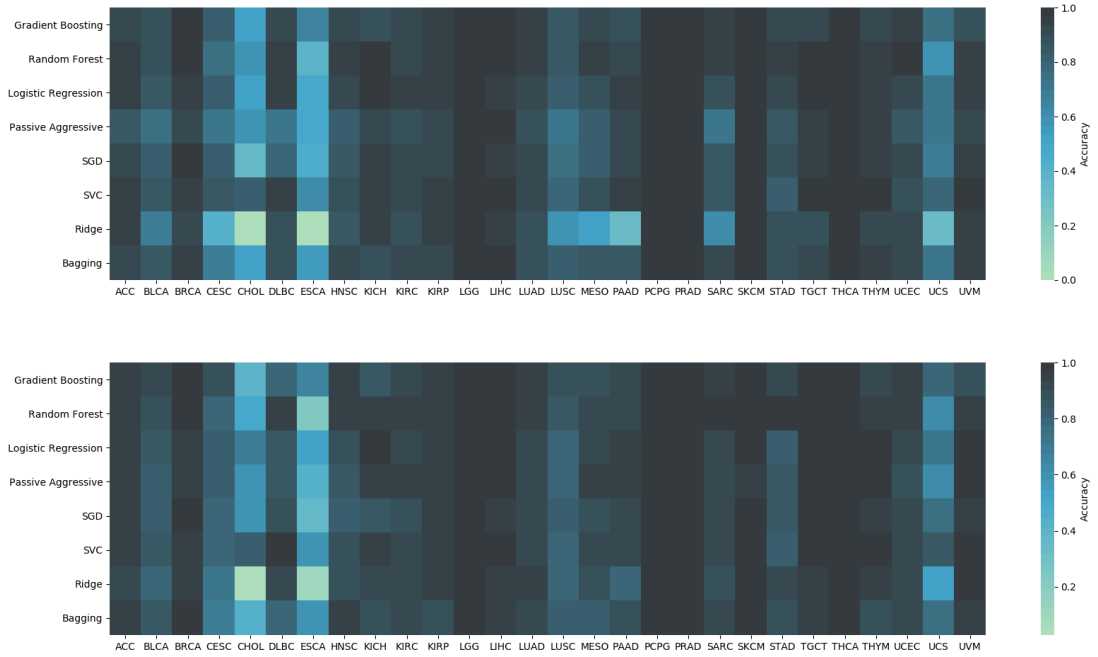


Figure 5.13: Heatmap of the accuracy by cancer type, by classifier using the 1,046 features (top) and the 100-miRNA signature (bottom).

Normal vs Tumor Tissue classification

A classification between Tumor Tissue (TT) vs Normal Tissue (NT) is performed in a 10-cross fold validation, using stratified cross-validation to maintain the proportions for the two classes inside the

folds. The global score and the classification accuracy by class are reported in Table 5.12. All of the classifiers have fair quality for differentiating between normal tissue and tumor tissue, except Ridge, which is more sensitive to the unbalanced number of examples.

Table 5.12: Accuracy for each classifier in a 10-fold cross-validation for the comparison between Tumor Tissue (TT) and Normal Tissue (NT) for 1,046 and 100 features.

Classifier	**100-NT**	**100-TT**	**1046-NT**	**1046-TT**	**100-Global**	**1046-Global**
Gradient Boosting	0.8612	0.9944	0.8707	0.9950	0.9846	0.9859
Random Forest	0.8091	0.9978	0.7256	0.9985	0.9839	0.9785
Logistic Regression	0.8423	0.9908	0.8659	0.9764	0.9799	0.9683
Passive Aggressive	0.7177	0.9798	0.8123	0.9728	0.9606	0.9611
SGD	0.8060	0.9902	0.7445	0.9936	0.9767	0.9754
SVC(linear)	0.8517	0.9892	0.8218	0.9771	0.9791	0.9657
Ridge	0.2997	0.9981	0.5994	0.9923	0.9470	0.9635
Bagging	0.8028	0.9953	0.7792	0.9966	0.9812	0.9807

Comparison to established feature selection methods

Several feature selection techniques have been proposed for microarray data [HG15]. The most effective approaches include Univariate Feature Selection (UFS), Recursive Feature Elimination (RFE), Elastic Net (EN), Genetic Algorithms (GALGO), Least Absolute Shrinkage and Selection Operator (LASSO) and Ensemble Feature Selection with Complete Linear Aggregation (EFS-CLA). UFS aims at finding the best features, scoring them using univariate statistical tests, such as the ANOVA F-value [LR78], and ultimately taking the k features with the highest scores. RFE runs several times a machine learning algorithm capable of scoring features, such as SVC, iteratively removing the feature with the lowest score [Guy+02] until it reaches the user-specified k features. EN simply runs the machine learning algorithm Elastic Net [FHT10], and takes the k highest-scored features. As Elastic Net is trying to balance accuracy and weight size in a linear model, exploiting L1 and L2 regularization, it is a popular choice for feature selection in bio-informatics [Bas+18; Sok+16], because it tends to create sparse models with few weights different from zero. LASSO is a regression analysis method, performing variable selection and regularization to improve prediction accuracy and interpretability of the statistical model it produces [Tib96], so it can be easily used for feature selection, only. All considered feature selection methods are implemented in the machine learning package `scikit-learn`, already used in the previous experiments. GALGO is a genetic algorithms-based feature selection library in R that ranks the features using several calls to a classifier and choosing the features that appear the most after evolving a subset several times [TF06]. EFS-CLA is a method that uses instances of SVM with several calls to a subsample of the data, ranks the features by weight value and reduces a percentage at each iteration [Abe+09].

As some of these techniques require the user to specify the number of features k to be taken, to provide a comparison with the approach presented in this section, $k = 100$ features are selected, using all the formerly described feature selection methods and compared classification accuracy on the considered classifiers with a 10-fold cross validation. For RFE, the SVC classifier is used in the loop, as not only it is commonly adopted for feature selection in bioinformatics [Guy+02; Sei+17], but also represents a good compromise between accuracy and speed of convergence on our specific dataset. For EN, `ElasticNetCV` scikit-learn method is selected, which exploits a 3-fold

cross-validation to automatically adapt the internal parameter α, balancing the importance of L1 and L2 regularization in the model. For the same reasons, the `LassoCV` scikit-learn method is selected for LASSO. For EFS-CLA, percentage of reduction $E = 20\%$, 40 as SVM calls per step, and $k = 100$, are set as parameters. Finally, a random selection of 100 features is also performed, to be used as a baseline reference to portray the efficiency of the feature selection algorithms.

From the results presented in Table 5.13, it is immediately clear that the 100 features selected by UFS are much less informative than the ones found by the proposed approach. RFE performs better, especially when considering SVC as the classifier used for the cross validation, but overall the performance for the other classifiers is lower. It must also be noted that, among all the methods, RFE is the most computationally expensive, as it calls the considered classifier, SVC in this case, $N-k = 1,046-100 = 946$ times, where N is the original number of features. All feature selection algorithms, as expected, perform much better than the baseline random selection of features.

Table 5.13: Comparison between different feature selection techniques and the proposed ensemble method for $k = 100$, on the TCGA dataset.

Classifier	**Random**	**GALGO**	**EFS-CLA**	**UFS**	**EN**	**LASSO**	**RFE**	**EFS**
Gradient Boosting	0.8588	0.8782	0.8871	0.9028	0.9208	0.9315	0.9309	0.9359
Random Forest	0.8515	0.8787	0.8824	0.8929	0.9224	0.9341	0.9288	0.9324
Logistic Regression	0.8015	0.8295	0.8832	0.8813	0.8988	0.8996	0.9088	0.9237
Passive Aggressive	0.6986	0.7235	0.8111	0.8091	0.8406	0.8424	0.8506	0.8831
SGD	0.7278	0.764	0.8446	0.8334	0.8649	0.8648	0.8824	0.9035
SVC	0.8077	0.8348	0.8706	0.885	0.9049	0.9008	0.9103	0.9154
Ridge	0.6534	0.6614	0.7422	0.7504	0.7753	0.7751	0.7954	0.8305
Bagging	0.822	0.8382	0.8562	0.8719	0.8889	0.9078	0.9061	0.911
Global Average	0.7777	0.8010	0.8472	0.8534	0.8771	0.8820	0.8892	0.9044
Calls to Classifier	-	60,000	480	-	-	10	946	80

A qualitative analysis of the features selected by each method shows that the highest-scoring ones are easily found by all considered approaches. In particular, from the 100 features found by our approach, 8 are in common with Random, 11 with GALGO, 29 with EFS-CLA, 38 are common to the group obtained through UFS, 44 are shared with the group found by LASSO, 48 again are found by EN, and 54 are in common with RFE.

Cross-platform validation on GEO datasets

As different datasets present distinctive sets of miRNAs, it is important to assess the performance of the identified signature on unseen data. Using the methodology previously described, the proposed approach is validated on the 14 GEO datasets. Each run of a classifier on a dataset was repeated 10 times, to compensate possible random elements that appear during the training phase of specific algorithms, e.g. `RandomForest`. It is worth noticing how this validation presents considerable challenges. Since the datasets are obtained by different platforms, not all of the 100 features in the signature were available everywhere. For most GEO datasets 98 were available, while for GSE62182 featured 75 of them. Furthermore, despite the transformation needed to bring the samples of the GEO datasets in the TCGA dataset space, samples measured by platforms used in the GEO datasets might prove particularly difficult to tackle for classifiers trained on TCGA samples, as most GEO datasets use microarray technology while TCGA uses sequencing. The properties of the used GEO datasets are summarized in Table 5.14.

Table 5.14: Summary of the used GEO datasets, and the number of features in common with our 100-miRNA signature.

Dataset ID	Platform	Tumor Type	#Samples	Total Feats.	Common Feats.	Reference
GSE34496	GPL8786	HNSC	44	847	98	-
GSE36802	GPL8786	PRAD	21	847	98	[Lin+13]
GSE67138	GPL8786	LIHC	57	847	98	-
GSE67139	GPL8786	LIHC	115	847	98	-
GSE45604	GPL14613	PRAD	50	2,143	98	[Cas+14]
GSE48088	GPL14613	BRCA	33	2,143	98	[Peñ+14]
GSE55856	GPL14613	ESCA	108	2,143	98	[Jan+17]
GSE86277	GPL14613	BRCA	72	2,143	98	[Rom+18]
GSE116182	GPL14613	LIHC	64	2,143	98	-
GSE86278	GPL16384	BRCA	49	3,242	98	[Rom+18]
GSE86281	GPL16384	BRCA	50	3,242	98	[Rom+18]
GSE31164	GPL10850	LIHC	110	851	98	[Mur+13]
GSE105134	GPL10850	BRCA	50	851	98	-
GSE62182	GPL11154	LUAD	94	3,242	75	[Vuc+14]

Fig. 5.14 shows the outcomes of the validation for all classifiers. In spite of the difficulties, most algorithms yielded good classification results, with `Logistic and SGD` in particular featuring over 93% average accuracy on all GEO datasets. Several classifiers, on the other hand, show poor performance on specific datasets, probably due to the way their decision boundaries for that specific class were learned on the TCGA dataset. In this sense, dataset GSE45604 proves to be the overall hardest to classify correctly for most algorithms. GSE86277,GSE86278 and GSE86281, deal with different molecular subtypes of BRCA, that could explain some of the performance issues. Finally the average performance in GSE62182, is because the classifiers have problems differentiating LUAD and LUSC. In general, however, different algorithms seem to have difficulties for different classes and datasets, which suggests that an ensemble approach for classification could compensate local issues.

Figure 5.14: Results with the 100 selected features in the GEO datasets, using a 10-fold cross-validation. From the average accuracy and standard deviation, SGD proves to be significantly better than the rest using a Kolmogorov-Smirnov test ($p < 0.05$)

To the best of our knowledge, the most similar work in literature that that the results can be

compared with is Telonis et al. [Tel+17], where isoform quantification was adopted to classify three of the GEO datasets used in this study (GSE36802, GSE67138, GSE67139), training `SVC` on a TCGA-derived dataset. For GSE36802, [Tel+17] reports an accuracy of 76%, that is surpassed by all of the classifiers. Considering GSE67138, for which an accuracy of 91% is reported, all the algorithms in our case perform better. Finally, for GSE67139, a 96% accuracy, again all the algorithms outperform that value. It must be noted, however, that even this comparison is made difficult by differences in how data was treated: for example, [Tel+17] reduced the number of classes to 6 and tested on 4 different types of tumors. In our study, all 28 classes are kept for testing.

Tumor subtypes

To further test the proposed approach, the 100-miRNA signature is used to classify tumor subtypes. As a comparison with GEO datasets is important for the validation, the molecular subtype in breast cancer (BRCA) is selected, as it's the only tumor class for which molecular subtype information is available in the GEO datasets. From the information in [Col+15; Net+12], it is possible to label 764 of the 777 BRCA samples in the TCGA dataset in 5 different subtypes (Luminal A, Luminal B, Triple-negative/basal-like, HER2-enriched and Normal-like). More information on the subtypes can be found in [Wei13]. Next, the accuracy in a 10-fold cross validation is computed for the 1,046 TCGA features and the 100-miRNA signature, with results reported in Table 5.15 and Table 5.16 respectively.

Table 5.15: Molecular subtype classification accuracy of Breast Cancer for the 1,046 features.

	Normal	**LumA**	**LumB**	**TNBC**	**Her2**	**Global**
#Samples	33	399	139	135	58	764
Gradient Boosting	0.1818	0.9348	0.5396	0.9333	0.5172	0.7987
Random Forest	0.0606	0.9724	0.4532	0.9630	0.0345	0.7657
Logistic Regression	0.1212	0.8747	0.5540	0.9259	0.4483	0.7606
Passive Aggressive	0.1515	0.8622	0.5612	0.9111	0.4483	0.7539
SGD	0.3030	0.9073	0.4604	0.9556	0.4655	0.7752
SVC	0.2727	0.8797	0.5252	0.9185	0.5345	0.7697
Ridge	0.1515	0.7293	0.4317	0.3704	0.2759	0.5524
Bagging	0.3333	0.9298	0.5108	0.9704	0.4310	0.7973
Average	0.1970	0.8863	0.5045	0.8685	0.3944	0.7467

Table 5.16: Molecular subtype classification accuracy of Breast Cancer for the 100 features.

	Normal	**LumA**	**LumB**	**TNBC**	**Her2**	**Global**
#Samples	33	399	139	135	58	764
Gradient Boosting	0.2424	0.9248	0.5324	0.9333	0.5517	0.7975
Random Forest	0.2121	0.9599	0.4029	0.9704	0.2069	0.7712
Logistic Regression	0.2727	0.8997	0.4892	0.9037	0.5517	0.7727
Passive Aggressive	0.3939	0.8546	0.4460	0.8667	0.5000	0.7358
SGD	0.4545	0.8897	0.4460	0.8444	0.4310	0.7475
SVC	0.5152	0.8446	0.5108	0.9037	0.5517	0.7581
Ridge	0.0606	0.9474	0.4388	0.8593	0.3966	0.7594
Bagging	0.2727	0.9173	0.4964	0.9481	0.3793	0.7777
Average	0.3030	0.9048	0.4703	0.9037	0.4461	0.7650

The best classification results are obtained for subtypes Triple-Negative Breast Cancer (TNBC) and Luminal A (LumA), due to the scarcity of samples for other subtypes (especially Normal and

Her2). Luminal B (LumB) presents considerable similarities to LumA, and the classifiers have difficulty separating the two subtypes using the data at our disposal. For these reasons, and the practical concern that TNBC is the subtype of BRCA with the worst prognosis, it is decided to tackle the issue as a binary classification problem, separating TNBC from the other classes. TNBC is a subtype of cancer where the cells have tested negative for estrogen receptors (ER), hormone epidermal growth factor receptor 2 (Her2), and progesterone receptors (PR). This subtype of cancer has limited treatment options and poor prognosis, as hormone therapies or targeted drugs do not work on it. Results of the binary classification problem on TCGA are reported in Table 5.17.

Finally, the binary subtype classification of BRCA is tested for the GEO datasets, using just the 100-miRNA signature. A single dataset composed of 4 series (GSE86281, GSE86277, GSE86278, GSE46823), is created with 2 classes: TNBC, featuring 139 samples, and all other molecular subtypes (LumA, LumB, and Her2), with 32 samples in total. Using the stem-loop sequences from platform GPL14613, and GPL1368, the 98 common stem-loop miRNAs of the 100 in the signature are used for the classification. Table 5.18 shows the results of the classification in a 10-fold cross validation, and the accuracy by class.

Table 5.17: TNBC classification from the other molecular subtypes in the TCGA dataset, using 1,046 features and 100 signature.

	TNBC-100	**TNBC-1046**	**Other-100**	**Other-1046**	**Global-100**	**Global-1046**
#Samples	135	135	629	629	764	764
Gradient Boosting	0.9111	0.8963	0.9857	0.9857	0.9725	0.9699
Random Forest	0.8889	0.8815	0.9905	0.9905	0.9725	0.9712
Logistic Regression	0.8963	0.9630	0.9793	0.9587	0.9647	0.9593
Passive Aggressive	0.8815	0.9630	0.9714	0.9523	0.9556	0.9540
SGD	0.8000	0.8222	0.9809	0.9841	0.9490	0.9555
SVC	0.8444	0.8963	0.9666	0.9825	0.9451	0.9673
Ridge	0.8000	0.7259	0.9825	0.9237	0.9503	0.8888
Bagging	0.8444	0.8963	0.9793	0.9825	0.9555	0.9673
Average	0.8583	0.8806	0.9795	0.9700	0.9582	0.9542

Discussion

The results of the five experiments performed with the 100-miRNA signature (Tumor Type Classification, Tumor Tissue vs Normal Tissue, GEO datasets, BRCA subtype in TCGA, and BRCA subtype in GEO datasets), are reported in Table 5.19. All classifiers show high levels of accuracy over all trials, with the validation on the GEO datasets (both tumor type and subtype classification) proving to be the hardest task.

As miRNAs have been shown to regulate approximately 30% of the human genes, and because their dysregulation has been associated with the development and progression of cancer, miRNAs have been found to have the potential to play a critical role in computational oncology. Nevertheless, their analysis and their employment in clinically relevant settings still faces various, specific technical challenges: a) the extremely small size of the miRNAs leads to diverse complications for example with respect to hybridization techniques, b) there is a lack of specificity in detection because of the high similarity of several miRNA family members, and c) the low expression of various miRNAs requires detection methods of utmost sensitivity [Li+14]. To date, most new miRNAs are discovered through cloning, despite these methods being time-consuming, low-throughput, and being biased

Table 5.18: Molecular subtype classification of Breast Cancer to separate TNBC from other breast cancer subtypes using the 100-miRNA signature, on the GEO dataset.

	TNBC	**Other**	**Global**
#Samples	139	44	183
Gradient Boosting	0.9353	0.7500	0.8909
Random Forest	0.9424	0.6136	0.8634
Logistic Regression	0.9065	0.6590	0.8476
Passive Aggressive	0.8561	0.7045	0.8197
SGD	0.9065	0.5227	0.8145
SVC	0.8561	0.7727	0.8355
Ridge	0.8993	0.6136	0.8300
Bagging	0.9496	0.7727	0.9070
Average	0.9065	0.6761	0.8511

Table 5.19: Comparison of the 8 classifiers, for the different experiments with the 100-miRNA signature. Logistic Regression was the best across all experiments, and Ridge has the worst accuracy.

Classifier	**TCGA**	**TT vs NT**	**GEO**	**TCGA (Subtype)**	**GEO (Subtype)**	**Global**
Gradient Boosting	0.9359	0.9846	0.6697	0.9725	0.8909	0.8907
Random Forest	0.9324	0.9839	0.8085	0.9725	0.8634	0.9121
Logistic Regression	0.9237	0.9799	0.9351	0.9647	0.8476	0.9302
Passive Aggressive	0.8831	0.9606	0.8678	0.9556	0.8197	0.8974
SGD	0.9035	0.9767	0.9393	0.9490	0.8145	0.9166
SVC	0.9154	0.9791	0.7724	0.9451	0.8355	0.8895
Ridge	0.8305	0.9470	0.8867	0.9503	0.8300	0.8889
Bagging	0.9110	0.9812	0.7682	0.9555	0.9070	0.9046

toward the discovery of abundant miRNAs [Che+09; LR09].

Nevertheless, it is possible conclude from the results that the extracted 100-miRNA signature is able to reliably classify the 28 different types of cancer in the TCGA dataset, and distinguish between normal and tumor tissue. In addition, it is sufficiently stable to be applicable across platforms, such as the ones such as the ones used in the ten GEO datasets and ahich show a good accuracy in differentiating TNBC from other molecular subtypes of BRCA. Looking ahead into the possibility of classifying tumor types using miRNAs, it becomes more and more pressing to consider circulating miRNAs, and their relationship to cancer studies.

For the miRNAs included in the signature, a bibliographic meta-analysis of specialized literature is performed. The proposed meta-analysis is mainly based on 5 surveys of circulating miRNAs for cancer studies [CLG13; Che15; He+15; Lar+16; Wan+14b]. Out of the 100 miRNAs in the signature, 77 appear as circulatory miRNAs, either in their stem-loop form or mature sequence. The complete list for the 100-miRNAs is reported in Annex A of the online supplementary material, in Fig. 5.15 shows the expression levels by type of cancer of the top 50 miRNAs.

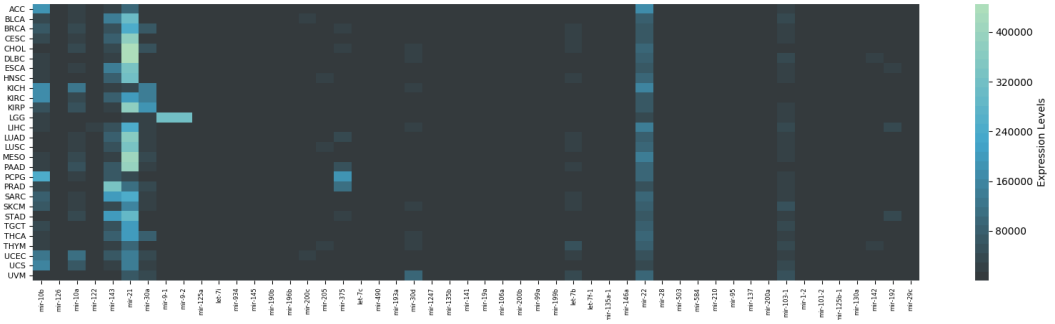


Figure 5.15: miRNAs mean expression levels (RPMs) of the top 50 miRNAs for each type of cancer tumor tissue.

Across all surveys analyzed, *hsa-miR-21*, included in the identified signature in stem-loop form, appears to be the most commonly over-expressed miRNA for all classes of tumors, as it is expected of a known oncomarker. 23 miRNAs in the signature do not appear in the surveys, but they are mentioned in recent research papers, as promising research leads whose role may need further corroboration: *miR-211* [Mar+15], *miR-135a* [Kog+10], *miR-3678-3p* [Giu+17], *miR-204* [Men+13], *miR-1228* [Tan+14], *miR-374b* [Sum+15], *miR-424* [Gir+13] *miR-217-5p* [Gir+13] *miR-3613-5p* [Mat+15], *miR-124* [Med+14], *miR-1277-5p* [Zhe+14] *miR-190* [Sch+14], *miR-934* [Tsu+15], *miR-490* [Jia+18], *miR-1247* [Wan+14a], *miR-199b* [Mon+18], *miR-135a* [Kog+10], *miR-503* [Shi+15], *miR-584* [Wan+18], *miR-137-3p* [Hsu+12], and *miR-103* [Jia+16].

Interestingly, *hsa-mir-135a-1* and *hsa-mir-135a-2*, located inside chromosomes 3 and 12, respectively, generate the same mature active sequence [Tri+16]. In the same manner, *hsa-mir-124-1*, *hsa-mir-124-2*, and *hsa-mir-124-3*, generate the same mature sequence *hsa-miR-124-5p*, and *miR-124* is known as a tumor suppressor in head and neck squamous cell carcinoma [Zha+17], hepatocellular carcinoma [Cai+17] and breast cancer [Wan+16]. All of them were identified by the proposed feature selection approach, indicting the presence of miRNA pathways shared across different tumor types. Targeting these miRNA pathways with anti-miRNA-based approaches such as infection with viral particles (having antisense sequence against the specific miRNA) or even

drug design of small molecules inhibitors of miRNAs (SMIRs) which can be considered potential anti-tumoral therapy. On the other hand, the down regulation of tumor suppressor miRNAs also contributes to the acquisition of malignant features. For example, by ectopic expression of *hsa-miR-944* which decreases malignant features in gastric [Pan+17], colorectal [Wen+17] and endometrial [He+17] cancers. Strikingly, *miR-944* and other understudied miRNAs could have been detected by the proposed approach analizing 28 different types of cancer, suggesting that they could play a key role in the biology of cancer. Future works will include further analyses of the 100-miRNA signature, crossing the information with genetic sources, assessing measures of gene quality and biomarker stability, using tools such as sigQC [Dha+17].

Conclusions

miRNAs fine-tune the regulation of the transcriptome [CCZ16; Mun+09]. Alterations in miRNA expression profiles are associated with several diseases, such as cancer. On the other hand, the altered miRNA expression profiles present in cancer could be used as prognostic and/or diagnostic markers. In summary, several miRNA signatures are associated with clinically relevant factors [Lam+15; SH17]. Therefore, the miRNA signature, which is obtained by using data from different types of cancers, can highlight the presence of so far underestimated miRNA's such as *miR-944*, and overall has the potential to be used in the frame of microarray based assays, as a potential building block in clinical decision support. Of course, further experimental validation on cancer patient samples will be required to weigh the biological significance of the signature in terms of diagnosing, treating and prognosing the outcome of cancer. The code and the datasets are available at `https://github.com/steppenwolf0/miRNAs100`

5.2.4 Experiment 2

Using the previously described methodology, a new set of experiments is performed on circulating miRNAs, only. When compared to other miRNA biomarkers, circulating miRNAs are easier to detect using non-intrusive clinical tests, like blood or urine tests, and could thus provide a more reliable foundation for personalized therapy.

Circulating miRNAs

A list of circulating miRNAs (mature sequences) is compiled, based on 5 reviews of circulating miRNAs from cancer studies [CLG13; Che15; He+15; Lar+16; Wan+14b]. Next, from this list, only miRNAs that appear in blood, serum, urine, plasma and saliva are selected. To narrow it further, only miRNAs that can be detected by Affymetrix platforms Affy-1 (GPL8786), Affy-2 (GPL14613), and Affy-3 (GPL16384) are kept. The choice of restricting to datasets from Affymetrix platforms GPL8786, GPL14613, GPL16384 has the aim of avoiding the known issue of miRNAs expression levels being platform and technology-dependent [BAB14; Del+13; Les+13]. After this selection, a total of 253 miRNA remain. The detailed list is included in Table 5.20.

Classification of cancer types

From the gene expression omnibus (GEO) repository [EDL02] 16 datasets for 10 different types of cancer are selected, based on clinical studies: Breast (BRCA), esophageal (ESCA), head and neck squamous cell (HNSC), liver hepatocellular (LIHC), prostate (PRCA), glioblastoma (GBM), colorectal (CRC), non-small-cell lung (NSCLC), gastric (GC) and ovarian (OVC), as summarized in Table 5.21. For each dataset, the raw data is downloaded and processed using the function *Affyrma()* from the Matlab bioinformatics toolbox™. This function processes the probe intensity values using RMA background adjustment, quantile normalization, and summarizing procedures,

let-7a	miR-140-3p	miR-19b	miR-335	miR-513a-3p
let-7a*	miR-141	miR-200a	miR-338-3p	miR-516b
let-7b	miR-142-3p	miR-200b	miR-338-5p	miR-518b
let-7c	miR-143	miR-200c	miR-339-3p	miR-520a-3p
let-7d	miR-144	miR-202	miR-339-5p	miR-548b-5p
let-7e	miR-145	miR-203	miR-340*	miR-557
let-7f	miR-146a	miR-205	miR-342-3p	miR-564
let-7g	miR-146b-3p	miR-206	miR-345	miR-566
let-7i	miR-146b-5p	miR-20a	miR-346	miR-571
miR-1	miR-148a	miR-20b	miR-34a	miR-574-3p
miR-100	miR-148b	miR-21	miR-34b	miR-574-5p
miR-101	miR-150	miR-210	miR-361-3p	miR-587
miR-106b	miR-150*	miR-212	miR-365	miR-589
miR-107	miR-151-5p	miR-214	miR-371-5p	miR-595
miR-10a	miR-152	miR-215	miR-372	miR-601
miR-10b	miR-155	miR-218	miR-373	miR-616*
miR-1182	miR-15a	miR-22	miR-375	miR-618
miR-122	miR-15b	miR-221	miR-376a	miR-622
miR-122*	miR-15b*	miR-222	miR-376c	miR-625
miR-1224-5p	miR-16	miR-223	miR-377	miR-625*
miR-1229	miR-16-2*	miR-23a	miR-378	miR-628-3p
miR-1231	miR-17	miR-23b	miR-378*	miR-629
miR-1245	miR-181a	miR-24	miR-379	miR-630
miR-1246	miR-181a-2*	miR-25	miR-382	miR-638
miR-1254	miR-181b	miR-26a	miR-409-3p	miR-646
miR-125b	miR-181d	miR-26b	miR-409-5p	miR-650
miR-125b-2*	miR-182	miR-27a	miR-410	miR-652
miR-126	miR-1825	miR-27b	miR-411	miR-654-3p
miR-1260	miR-183	miR-296-5p	miR-421	miR-656
miR-1268	miR-184	miR-298	miR-423-5p	miR-668
miR-127-3p	miR-185	miR-299-5p	miR-425	miR-675
miR-1275	miR-186	miR-29a	miR-425*	miR-7
miR-128	miR-187	miR-29b	miR-429	miR-708
miR-1280	miR-187*	miR-29c	miR-431	miR-744
miR-1284	miR-18a	miR-301a	miR-431*	miR-744*
miR-1285	miR-18b	miR-302b	miR-432	miR-760
miR-1288	miR-18b*	miR-30a	miR-451	miR-874
miR-1290	miR-190b	miR-30b	miR-452	miR-885-5p
miR-1295	miR-191	miR-30c	miR-454	miR-922
miR-129-5p	miR-192	miR-30c-1*	miR-454*	miR-92a
miR-1304	miR-193a-3p	miR-30d	miR-483-3p	miR-92a-2*
miR-130a*	miR-193b	miR-30e	miR-483-5p	miR-92b
miR-130b	miR-194	miR-31	miR-484	miR-93
miR-1323	miR-195	miR-32	miR-486-3p	miR-93*
miR-133a	miR-196a	miR-320a	miR-486-5p	miR-936
miR-133b	miR-196b	miR-320c	miR-487b	miR-939
miR-134	miR-197	miR-320d	miR-493	miR-942
miR-138	miR-198	miR-324-3p	miR-494	miR-99a
miR-138-2*	miR-199a-3p	miR-326	miR-497	miR-99b
miR-139-3p	miR-199a-5p	miR-328	miR-502-5p	
miR-139-5p	miR-19a	miR-331-3p	miR-504	

Table 5.20: List of all circulating miRNAs.

then outputs *Expression* (non-dimensional). The resulting aggregated dataset for the multi-class classification problem presents 845 samples, 253 features and 10 different tumor classes. Next, the Z-score normalization is applied to the dataset, to then run the feature selection algorithm in a 10-fold stratified cross-validation scheme.

Table 5.21: GEO repository datasets of miRNA cancer studies used in the project for platforms GPL8786, GPL14613 and GPL16384.BRCA: breast cancer; ESCA: esophageal cancer; HSNC: head and neck squamous cell cancer; LIHC: liver hepatocellular cancer; PRCA: prostate cancer; GBM: gliobastoma; CRC: colorectal cancer; NSCLC: non-small-cell lung cancer; GC: gastric cancer; OVC: ovarian cancer

Dataset	Samples	Type	Reference	Class	Platform
GSE48088	33	BRCA	[Peñ+14]	0	GPL14613
GSE86277	72	BRCA	[Rom+18]	0	GPL14613
GSE86278	49	BRCA	[Rom+18]	0	GPL14613
GSE86281	50	BRCA	[Rom+18]	0	GPL16384
GSE55856	108	ESCA	[Jan+17]	1	GPL14613
GSE34496	44	HSNC	-	2	GPL8786
GSE67138	57	LIHC	-	3	GPL8786
GSE67139	115	LIHC	-	3	GPL8786
GSE116182	64	LIHC	-	3	GPL14613
GSE36802	21	PRCA	[Lin+13]	4	GPL8786
GSE45604	50	PRCA	[Cas+14]	4	GPL14613
GSE104554	38	GBM	[Her+17]	5	GPL14613
GSE110402	75	CRC	[Jep+18]	6	GPL14613
GSE46729	24	NSCLC	-	7	GPL8786
GSE63121	15	GC	[Zha+15]	8	GPL8786
GSE47841	30	OVC	[Elg+14]	9	GPL14613

Then, the results are compared against two current state-of-the-art feature selection methodologies: A homogeneous ensemble classifier, exploiting variations of SVC [Abe+09]; and a feature selection tool based on genetic algorithms, called GALGO [TF06]. Since each algorithm contains stochastic elements, each algorithm is run 10 times (Fig. 5.16) and the set of features with the best average accuracy is kept. When applied to the 253 circulating miRNAs, the top features obtained by each classifier appear as in Fig. 5.17.

The homogeneous ensemble uses several runs of SVC to rank the features by weight, and reduces the number of features by a given percentage at each step. In this case the same parameters as for algorithm 2 are used; 20% step reduction and 90% accuracy as stop parameter. In contrast, for GALGO to obtain a fair comparison, the requested number of features is set to the resulting number of features from the heterogeneous ensemble feature selection classifier.

Finally, the genes targeted by the candidate miRNAs are analyzed using miRNet [Fan+16]. The parameters for the miRNet analysis are: target genes as main function with a 0.05 *Betweenness* filter, and pathway enrichment analysis using the Kyoto Encyclopedia of Genes and Genomes [Kan+16] (KEGG) and Gene Ontology-Biological Process [FG06] (GO:BP). Using a *Betweenness* filter implies that the genes must be targeted by at least 2 miRNAs.

The resulting most significant 5 features uncovered by the presented algorithm are hsa-let-7a,

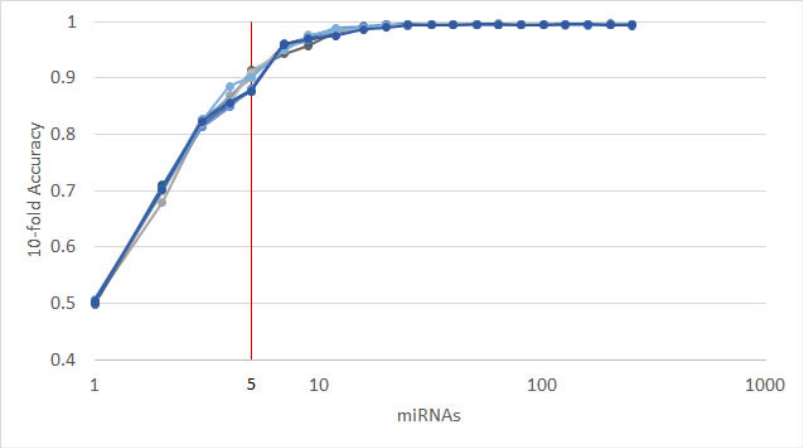


Figure 5.16: Results of 10 runs of the recursive ensemble feature selection for cancer type classification. The x axis cuts at 5 variables, where all runs cross the 90% average accuracy stop parameter (the subsequent values are computed as a reference).

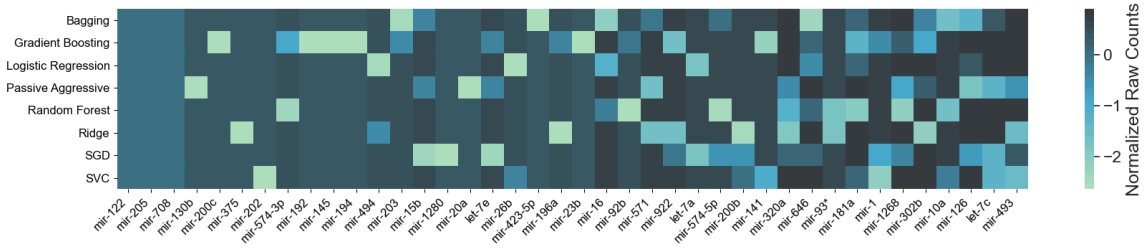


Figure 5.17: Feature importance by classifier. On the horizontal axis, the top features are reported, following their ensemble ranking. The intensity of the color in each square represents the frequency of appearance of that particular feature in the 10 instances of the same classifier produced by cross-validation; the darker the color the most frequent the appearance of that feature among the most important. It is noticeable how different classifiers rank features differently. For this figure, only the top 42 features are reported, for visualization purposes.

hsa-miR-23b, hsa-miR-122, hsa-miR-708, and hsa-miR-200c, with seemingly different *Expression* levels for each cancer type (Fig. 5.18). The classifiers Gradient Boosting, Random Forest, SVC and Bagging seem to work in a satisfying way for all tumor types using only 5 miRNAs, whereas the rest have issues classifying the types of cancer HNSC, GC and OVC, while having better performance when using the full 253 miRNAs, as shown in Fig. 5.19. Interestingly, hsa-let-7 and hsa-miR-200c were also discovered by the homogeneous ensemble, while GALGO's performance seems to be less effective and has no miRNAs in common with the proposed approach. From the comparison with GALGO and the homogeneous ensemble classifier with SVC, it is noticeable how the proposed heterogeneous ensemble classifier outperforms the other feature selection techniques in Table 5.22.

Table 5.22: Comparison of the results of the different feature selection algorithms, reducing from the initial 253 to 5 features to differentiate cancer types.

	Heterogeneous Ens. 5 Feats.		Homogeneous Ens. 5 Feats.		GALGO 5 Feats.		253 Feats.	
	μ	σ	μ	σ	μ	σ	μ	σ
Gradient Boosting	0.9751	0.0134	0.9797	0.0154	0.8374	0.0453	0.975	0.0128
Random Forest	0.9761	0.0192	0.9854	0.0155	0.8656	0.0383	1	0
Logistic Regression	0.8877	0.0239	0.8777	0.0281	0.4954	0.0416	1	0
Passive Aggressive	0.8239	0.0544	0.8144	0.0707	0.4545	0.0590	1	0
SGD	0.8937	0.0305	0.8632	0.0362	0.5204	0.0832	0.9941	0.0078
SVC	0.9620	0.0197	0.9499	0.0186	0.5308	0.0454	1	0
Ridge	0.8083	0.0272	0.6900	0.0173	0.5010	0.0451	0.9977	0.0045
Bagging	0.9702	0.0193	0.9643	0.0165	0.8418	0.0425	0.9894	0.0121
Global	0.9121	0.0260	0.8906	0.0273	0.6309	0.0500	0.9945	0.0047

Numerical validation

To further validate the methodology, the dataset described in subsection 5.2.4 is split into a training (90%) and a validation (10%) subsets. Then, 10 instances of the recursive ensemble feature selection algorithm are run with 90% of the data in a 10-fold cross-validation, which yields the curve in Fig. 5.20.

Next, the smallest signature that provided an accuracy of 90% or above is selected, having as result: hsa-let-7a, hsa-mir-122, hsa-mir-200c, hsa-mir-708 and hsa-mir-23b, the same miRNAs identified in the previous experiment described in subsection 5.2.4. Then, this signature is tested on the 10% subset, comparing against signatures identified by other approaches: homogeneous ensemble feature selection, GALGO, K-best univariate feature selection (using f-score), and 3 random selected subsets. In addition, the test set's labels are shuffled to verify the proper working of the classifiers (Table 5.23). Finally, the Matthews Correlation Coefficient (MCC) is comptued for all of the signatures and classifiers [JRF12] (Table 5.24).

From the 10 instances, the frequency of appearance of miRNAs in the top 5 features is then measured for each run. From the original 253 features, only 10 features appear in the top 5 for the heterogeneous recursive ensemble feature selection algorithm, with frequencies presented in Fig. 5.21. The same procedure was repeated for 10 runs of the homogeneous ensemble feature selection algorithm (feature frequency presented in Fig. 5.22) and GALGO (feature frequency presented in Fig. 5.23). The variability of the output signature for each algorithm reflected that the

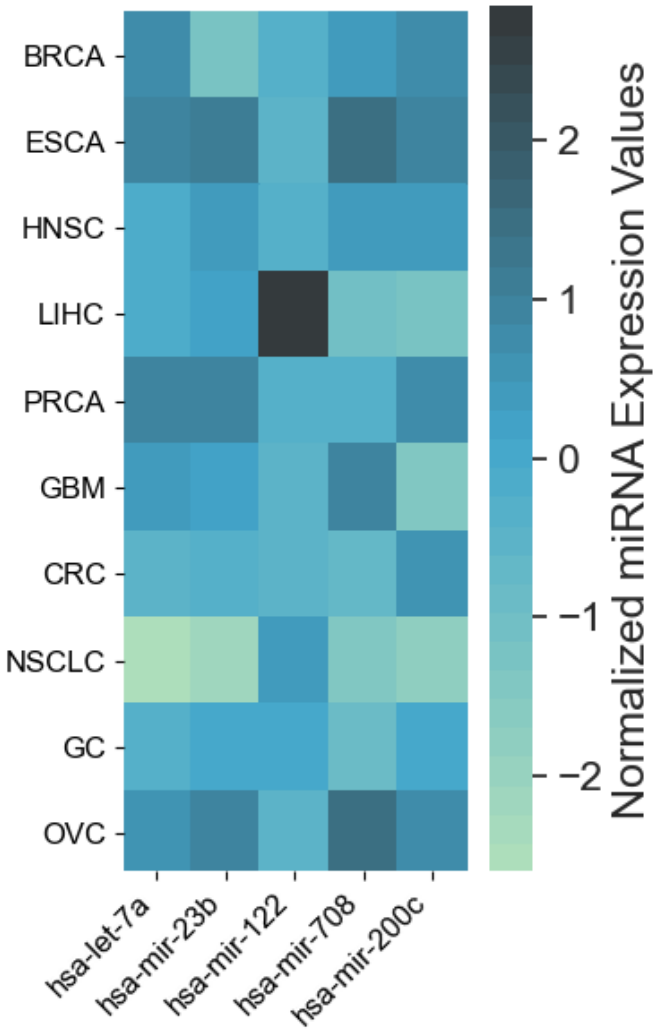


Figure 5.18: Heatmap of average expression levels by cancer type for the 5 miRNAs identified by the proposed approach. Cancer types: breat (BRCA); esophageal (ESCA); head and neck squamous cell (HNSC); liver hepatocelluar (LIHC); prostate (PRCA); gliobastoma (GBM); colorectal (CRC); non-small-cell lung (NSCLC); gastric (GC); ovarian (OVC).

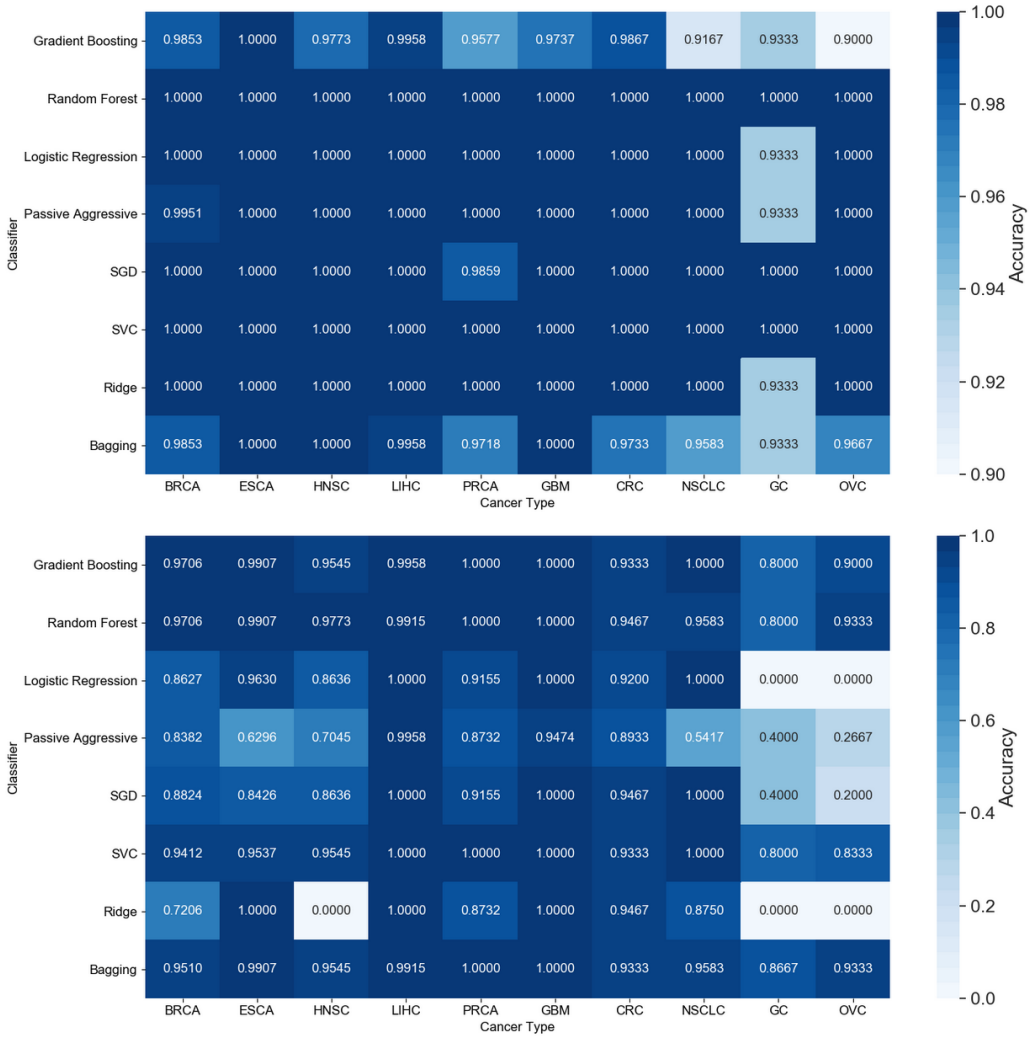


Figure 5.19: Comparison of accuracy by classifier and tumor type for all the 253 features (top) and the 5 features identified by the proposed approach (bottom). Cancer types: breast (BRCA); esophageal (ESCA); head and neck squamous cell (HNSC); liver hepatocellular (LIHC); prostate (PRCA); gliobastoma (GBM); colorectal (CRC); non-small-cell lun g (NSCLC); gastric (GC); ovarian (OVC)

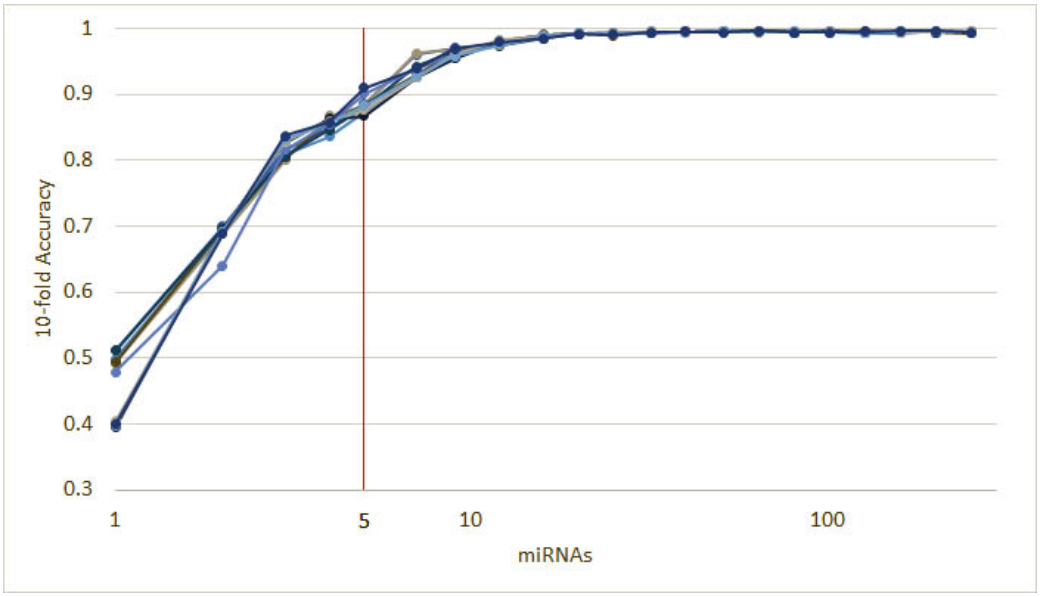


Figure 5.20: 10 runs of the heterogeneous ensemble recursive selection algorithm. From the 10 runs, the minimum number of necessary miRNA to have an accuracy above 90% is 5; hsa-let-7a, hsa-miR-23b, hsa-miR-122, hsa-miR-708, and hsa-miR-200c.

Table 5.23: Accuracy on the 10% data for testing the feature selection algorithm. Results for the signatures found by the heterogenous recursive ensemble feature selection algorithm, the homogeneous recursive ensemble feature selection algorithm, K-Best feature selection algorithm using f-score as selection citeria, 3 random feature subsets, and a random shuffle of the test labels.

	Heterogeneous	Homogeneous	Univariate	GALGO	Random 1	Random 2	Random 3	Shuffle
Gradient Boosting (n_estimators=300)	0.9412	0.9294	0.9412	0.9176	0.8824	0.8353	0.8000	0.2471
Random Forest (n_estimators=300)	0.9412	0.9529	0.9412	0.9059	0.8941	0.8235	0.8235	0.2471
Logistic Regression	0.9059	0.8824	0.8588	0.8706	0.6353	0.5412	0.5882	0.2824
Passive Aggressive	0.8706	0.7765	0.7176	0.8471	0.4235	0.4118	0.5294	0.1765
SGD	0.8824	0.8588	0.7765	0.7882	0.5294	0.4235	0.3765	0.2353
SVC(linear)	0.9765	0.9176	0.8941	0.8588	0.6235	0.6235	0.5412	0.2824
Ridge	0.8118	0.7059	0.7412	0.7059	0.5882	0.4588	0.4000	0.2706
Bagging (n_estimators=300)	0.9412	0.9294	0.9176	0.8824	0.8706	0.8471	0.8235	0.2118
Average	0.9089	0.8691	0.8485	0.8471	0.6809	0.6206	0.6103	0.2442

average and standard deviation for accuracy and MMC for the proposed heterogeneous recursive ensemble feature selection algorithm is better than the homogeneous recursive ensemble feature selection algorithm and GALGO (see Table 5.25).

Pathway Analysis

Next, using the 5 candidate miRNAs identified by the proposed approach to separate the tumor type, miRNet is run to identify the targeted genes, obtaining a total of 1,732 genes. After the application of a 0.05 *Betweenness* filter, the list is reduced to 156 genes. From these genes, BCL2, CCNG1, COX1, TUBB2A, CELF1 and FOXJ3 are targeted by at least 3 of the 5 miRNAs (Fig. 5.24). Finally, using the function Explorer of miRNet, a functional enrichment analysis is performed, using a hypergeometric test of the genes from the KEGG database and GO: BP. Table 5.26 and Table 5.27, show the results of the top 10 functional enrichment analysis for KEGG and GO:BP respectively.

Table 5.24: Matthews Correlation Coefficient values for the 10% data left for testing the feature selection algorithm. Results for the heterogenous recursive ensemble feature selection algorithm, the homogeneous recursive ensemble feature selection algorithm, K-Best feature selection algorithm using f-score as selection citeria, 3 random feature subsets, and a random shuffle of the test labels.

	Heterogeneous	Homogeneous	Univariate	GALGO	Random 1	Random 2	Random 3	Shuffle
Gradient Boosting (n_estimators=300)	0.9346	0.9216	0.9346	0.9085	0.8693	0.8170	0.7778	0.1634
Random Forest (n_estimators=300)	0.9346	0.9477	0.9346	0.8954	0.8824	0.8039	0.8039	0.1634
Logistic Regression	0.8954	0.8693	0.8431	0.8562	0.5948	0.4902	0.5425	0.2026
Passive Aggressive	0.8562	0.7516	0.6863	0.8301	0.3595	0.3464	0.4771	0.0850
SGD	0.8693	0.8431	0.7516	0.7647	0.4771	0.3595	0.3072	0.1503
SVC(linear)	0.9739	0.9085	0.8824	0.8431	0.5817	0.5817	0.4902	0.2026
Ridge	0.7908	0.6732	0.7124	0.6732	0.5425	0.3987	0.3333	0.1895
Bagging (n_estimators=300)	0.9346	0.9216	0.9085	0.8693	0.8562	0.8301	0.8039	0.1242
Average	0.8987	0.8546	0.8317	0.8301	0.6454	0.5784	0.5670	0.1601

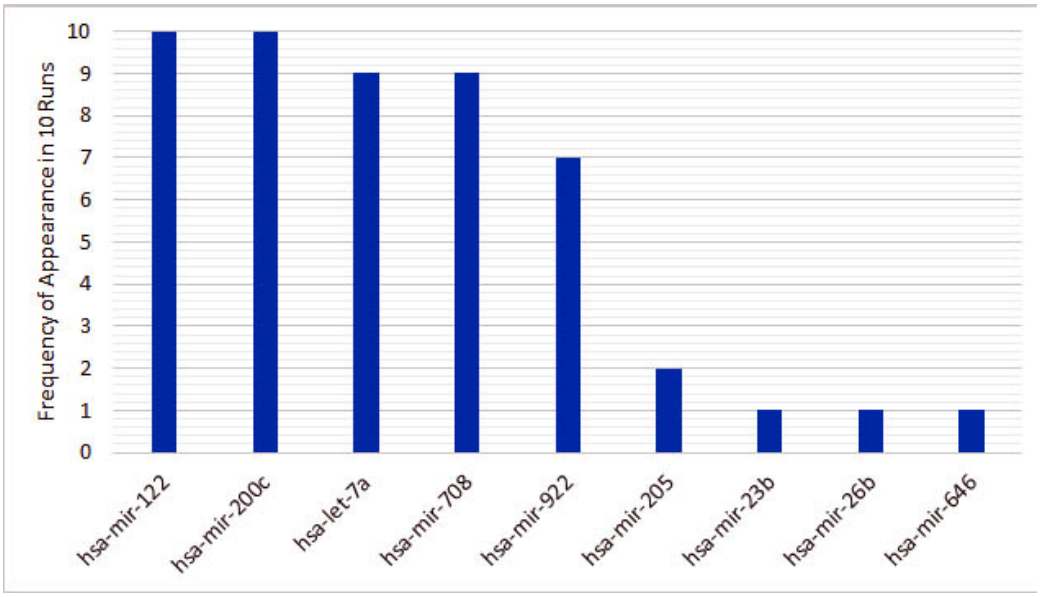


Figure 5.21: 10 recurrent features in the 5-feature signature for the heterogeneous ensemble feature selection algorithm.

Table 5.25: μ and σ for accuracy and MMC over 10 runs using the top 5 features, for each algorithm.

	Accuracy		MCC	
	μ	σ	μ	σ
Heterogeneous	0.8840	0.0120	0.8691	0.0156
Homogeneous	0.8518	0.0183	0.8353	0.0204
GALGO	0.8227	0.0255	0.8132	0.0338

The first result in KEGG is the P53 signaling pathway. The P53 protein is a tumor suppressor protein and it is involved in several anticancer mechanisms [Ste12]. In GO: BP database, the first result is the cellular response to stress, with 44 of the genes in the pathway; cellular stress is a component of the P53-mediated tumor suppression [CBS07].

Table 5.26: Top 10 miRNet Enrichment Analysis in the KEGG Dataset for miRNAs hsa-miR-122, hsa-let-7a, hsa-miR-23b, hsa-miR-708, and hsa-miR-200c.

Pathway	Total	Expected	Hits	Pval
p53 signaling pathway	68	1	10	3.70E-06
Pathways in cancer	310	4.57	19	3.70E-06
Prostate cancer	87	1.28	11	3.70E-06
Glioma	65	0.958	8	0.000207
Melanoma	68	1	7	0.00196
Bladder cancer	29	0.428	5	0.00196
Colorectal cancer	49	0.722	6	0.00217
Chronic myeloid leukemia	73	1.08	7	0.00227
Focal adhesion	200	2.95	11	0.00327
Small cell lung cancer	80	1.18	7	0.00327

Table 5.27: Top 10 miRNet Enrichment Analysis in the GO:BP Dataset for miRNAs hsa-miR-122, hsa-let-7a, hsa-miR-23b, hsa-miR-708, and hsa-miR-200c.

Pathway	Total	Expected	Hits	Pval
Cellular response to stress	1620	15.4	44	3.03E-08
Positive regulation of cell proliferation	786	7.43	27	1.66E-06
Response to hypoxia	245	2.31	15	2.53E-06
Regulation of cell cycle	886	8.37	28	2.53E-06
Regulation of cell proliferation	1430	13.5	36	4.30E-06
Response to abiotic stimulus	876	8.28	27	5.44E-06
Negative regulation of cell cycle	520	4.91	20	1.04E-05
Regulation of molecular function	2250	21.2	46	1.08E-05
Regulation of cyclin-dependent protein kinase activity	89	0.841	9	1.40E-05
Negative regulation of apoptotic process	679	6.42	22	2.56E-05

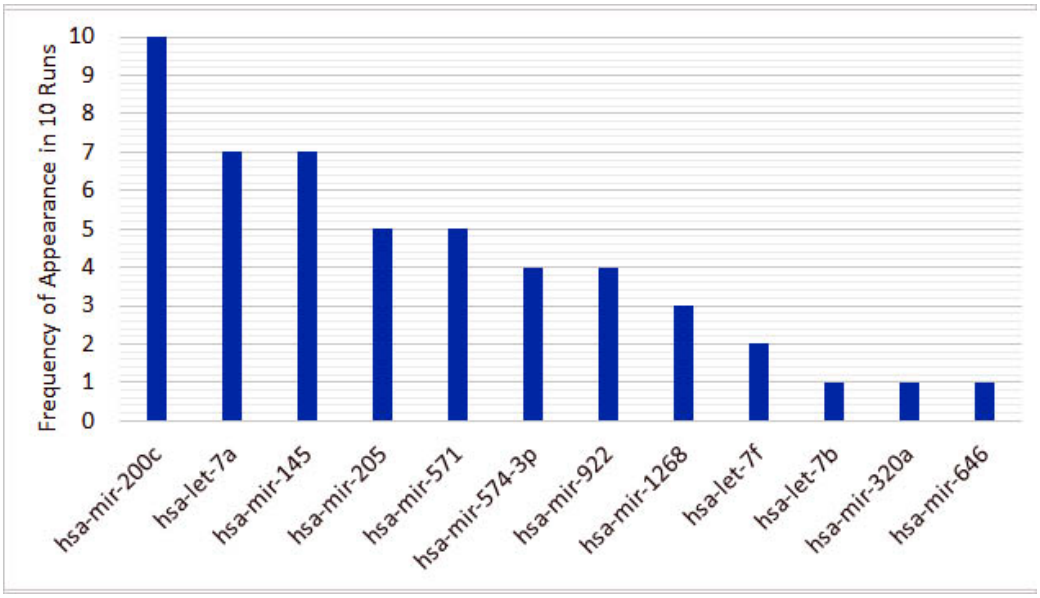


Figure 5.22: 12 recurrent features in the 5-feature signature for the homogeneous ensemble feature selection algorithm.

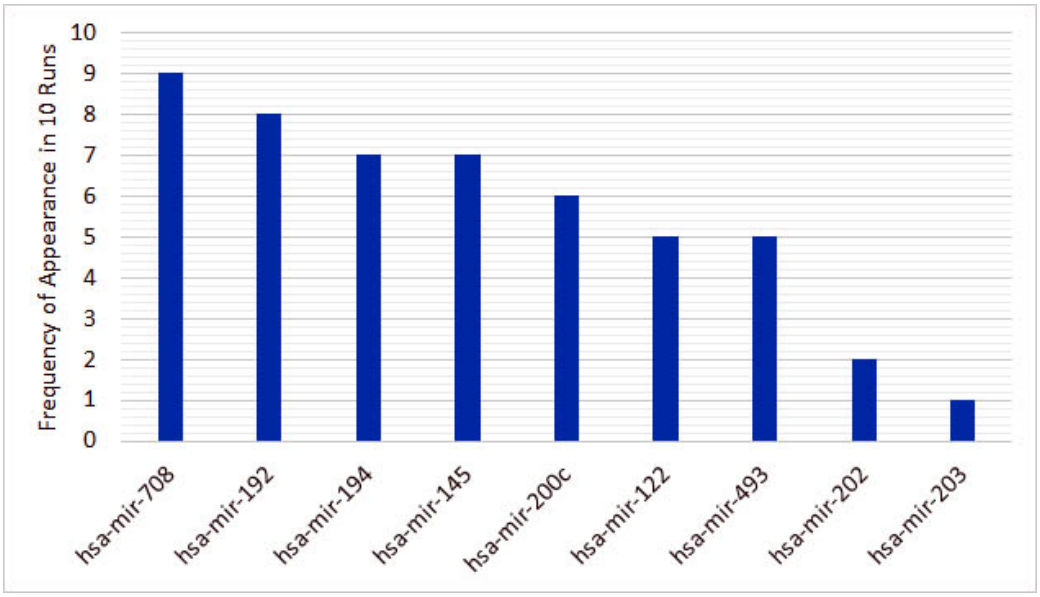


Figure 5.23: 9 recurrent features in the 5-feature signature for the GALGO.

Triple-Negative Breast Cancer Classification

Cancer tumors are divided into tumor subtypes, which can be treated by different strategies depending on their classification. From the available data in the GEO repository, it was possible to compile a dataset to assess the possibility of classifying tumor subtypes (Luminal A, Luminal B, HER2-enriched, Triple-Negative and Normal [Dai+15]) in breast cancer (BRCA) using circulating miRNAs. Then, datasets GSE86277, GSE86278, GSE86281 and GSE46823 were selected, all of

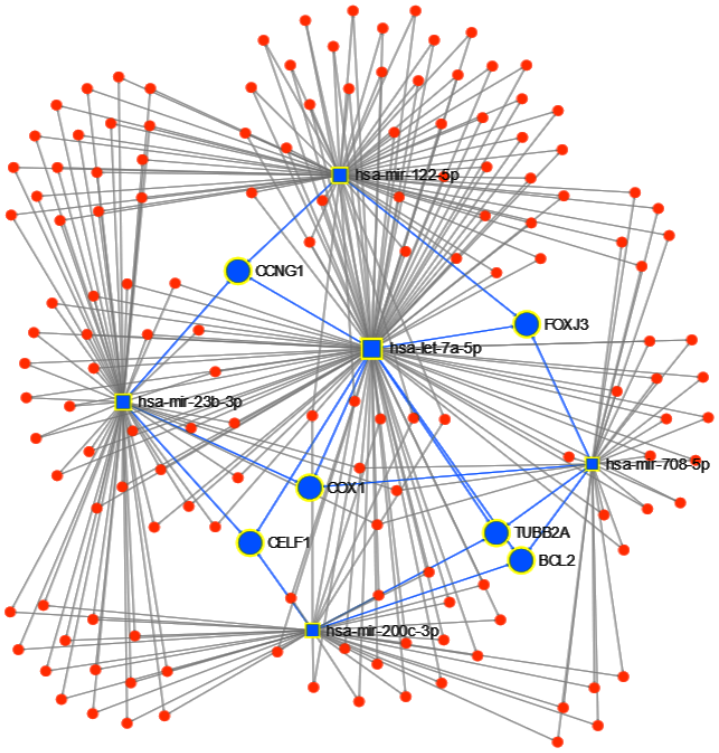


Figure 5.24: miRNET targeted genes analysis, showing genes targeted by at least 3 of the 5 miRNA to classify cancer type: BCL2, CCNG1, COX1, TUBB2A, CELF1 and FOXJ3.

them BRCA studies with sub-type information. From the BRCA subtypes, Triple-Negative has the worst prognosis, as it is resistant to hormone therapies [Wei13]. For this reason, the labels of the resulting dataset are set to separate the Triple-Negative subtype from the rest. Although making an analysis of all the sub-types would have been more interesting, the unbalance in the subtypes samples found in the original data makes it impossible: Thus, more precisely, the resulting dataset has 139 Triple-Negative samples and only 44 from all the rest of the subtypes, for a total of 183 samples, 253 features, and 2 classes (Triple-Negative/Other).

Next, the function *Affyrma()* from the Matlab bioinformatics toolbox™is applied, along with a Z-score normalization on the dataset to run the feature selection algorithm in a 10-fold stratified cross-validation scheme. As in the previous experiment, the feature selection algorithm was set to identify the smallest miRNA subset sufficient to obtain a 90% accuracy. In addition, the results are compared with the 31-miRNA signature proposed by Romero et al. [Rom+18] to separate Triple-Negative Breast Cancer (TNBC) from other sub-types of BRCA using miRNA-mRNA integrative analysis in TNBC tumors, based on the differential expressed transcripts. It is important to take into consideration that this 31-miRNA list considers non-circulating miRNAs, that are not included in the proposed method, and could potentially access more information. Finally, miRNet is run using the candidate miRNAs, as in the previous experiment.

The heterogeneous ensemble algorithm is run 10 times, identifying 5 meaningful miRNA features for separating Triple-Negative BRCA from the other sub-types (Fig. 5.25). The resulting miRNAs are: hsa-miR-378*, hsa-miR-221, hsa-miR-342-3p, hsa-miR-630, and hsa-miR-145. The corresponding expression levels for the identified miRNAs in TNBC and non-TNBC are reported in Fig. 5.26.

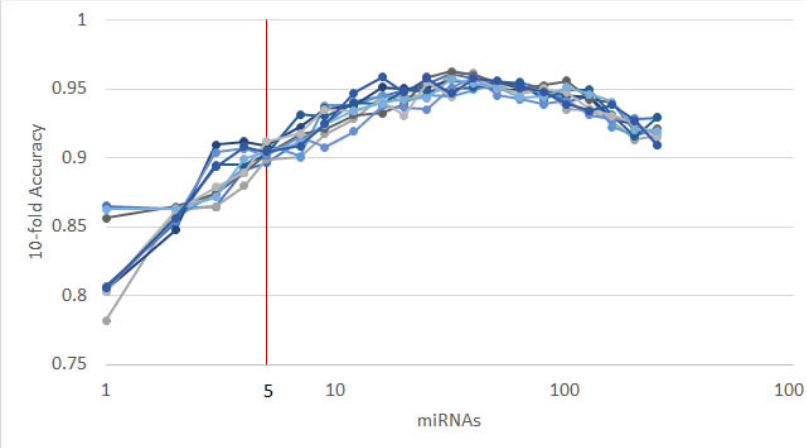


Figure 5.25: Results of 10 runs of the recursive ensemble feature selection for the TNBC discrimination example. The *x* axis cuts at 5 variables, which is where most evaluations cross the average 0.90 accuracy stop parameter.

Next, the accuracy of all classifiers is compared, using all the 253 miRNAs in the dataset, using the identified 5-miRNA signature, and the 31-miRNA signature proposed by Romero et al. for distinguishing TNBC from other cancers (Table 5.28). From the results, the proposed algorithm outperforms the 31-miRNA signature. In addition, the area under the curve (AUC) of the results (Fig. 5.27) calculated with Gradient Boosting classifier is above 90%. This is considered as an *outstanding* results, following the guidelines in [Man10; Šim09] for clinical use of algorithmic methodologies.

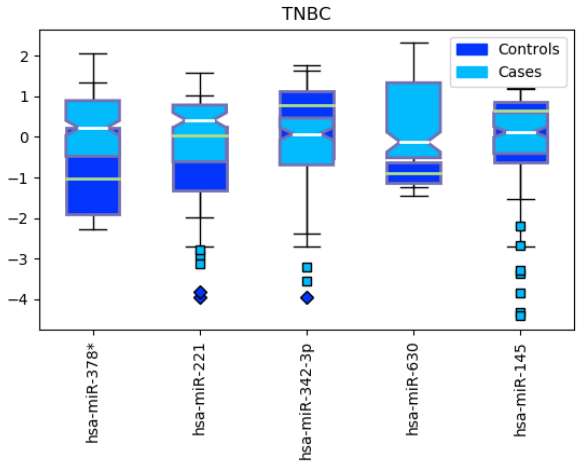


Figure 5.26: Boxplot for the *Expression* levels between Triple Negative Breast Cancer (TNBC, cases) and other subtypes (controls).

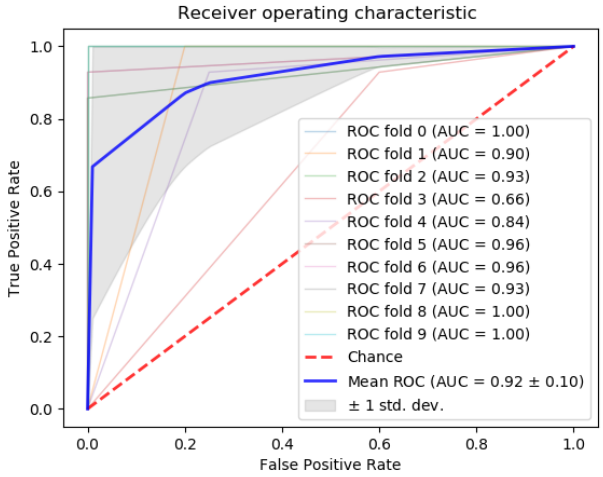


Figure 5.27: ROC Curve using the gradient boosting classifier to separate Triple Negative Breast Cancer (TNBC) from the rest of breast cancer subtypes.

Table 5.28: Accuracy comparison for all classifiers, using all 253 features, the 5-miRNA signature found by the proposed approach, and the 31-miRNA signature from Romero et al., for separating Triple-Negative from the rest of BRCA sub-types.

	5 Feats.		253 Feats.		Romero et al.	(31 Feats.)
Classifier	μ	σ	μ	σ	μ	σ
GradientBoosting	0.9345	0.0523	0.9134	0.0487	0.9239	0.0485
RandomForest	0.9354	0.0617	0.9459	0.0416	0.9184	0.0432
LogisticRegression	0.9243	0.0487	0.9406	0.0612	0.8958	0.0643
PassiveAggressive	0.9076	0.0550	0.9076	0.0778	0.8797	0.0637
SGDClassifier	0.9085	0.0628	0.8918	0.0770	0.8692	0.0700
SVC(linear)	0.9243	0.0487	0.9242	0.0655	0.8572	0.0400
Ridge	0.9079	0.0754	0.9085	0.0533	0.8856	0.0611
Bagging	0.9295	0.0411	0.9076	0.0544	0.9341	0.0412
Global	0.9215	0.0557	0.9175	0.0599	0.8955	0.0540

Finally, the results of miRNet found 1,294 genes targeted by the 5 miRNAs, with 79 having at least 2 miRNAs in common. From those 79, MTDH is targeted by 4 miRNAs, while IGF1R and CDK6 are targeted by by 3, see Fig. 5.28. From the enrichment analysis, the most important functional pathway in the KEGG database (Table 5.29) is the *p53 signaling pathway* (the same identified in the previous experiments for separating cancer types); and in GO:BP (Table 5.30), the *negative regulation of cell proliferation*, with 12 of the 79 genes followed by *regulation of cell proliferation* and just *cell proliferation*. This results show an important involvement of cell proliferation in TNBC.

Table 5.29: Top 10 miRNet enrichment analysis results for miRNAs hsa-miR-378*, hsa-miR-221, hsa-miR-342-3p, hsa-miR-630 and hsa-miR-145 using the KEGG database.

Pathway	Total	Expected	Hits	Pval
p53 signaling pathway	68	0.509	6	0.000518
Pancreatic cancer	69	0.516	6	0.000518
Glioma	65	0.486	6	0.000518
Melanoma	68	0.509	6	0.000518
Chronic myeloid leukemia	73	0.546	6	0.000576
Bladder cancer	29	0.217	4	0.00197
Cell cycle	124	0.927	6	0.00821
Pathways in cancer	310	2.32	9	0.009
Non-small cell lung cancer	52	0.389	4	0.0133
Adherens junction	70	0.524	4	0.0368

Discussion

In this section, an analysis of the candidate miRNAs identified by the proposed feature selection method is reported, using the available literature in cancer studies.

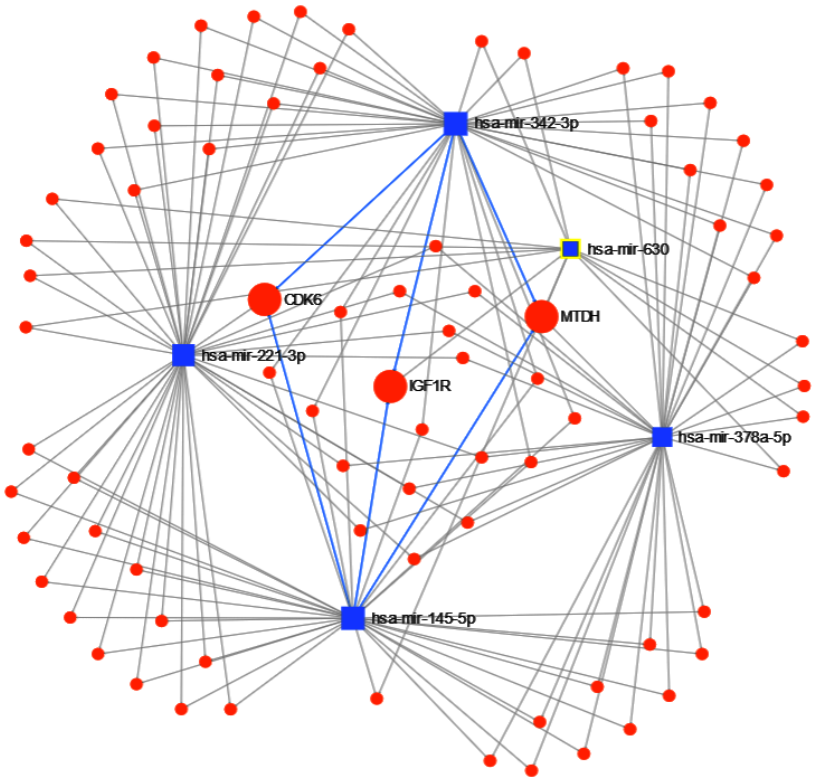


Figure 5.28: miRNET targeted genes analysis, showing genes targeted by at least 3 of the 5 miRNA to separate Triple Negative Breast Cancer (TNBC) from other breast cancer subtypes: metastasis gene metadherin-positive (MTDH), type 1 insulin-like growth factor receptor-positive (IGF1R), cyclin-dependent kinase 6-positive (CDK6).

,

Table 5.30: Top 10 miRNet enrichment analysis results for miRNAs hsa-miR-378*, hsa-miR-221, hsa-miR-342-3p, hsa-miR-630 and hsa-miR-145 using the GO:BP database.

Pathway	Total	Expected	Hits	Pval
negative regulation of cell proliferation	585	2.7	12	0.00631
regulation of cell proliferation	1430	6.6	19	0.00631
cell proliferation	1900	8.79	22	0.00674
G1 phase of mitotic cell cycle	47	0.217	4	0.00882
enzyme linked receptor protein signaling pathway	1180	5.43	16	0.00882
myeloid cell differentiation	296	1.37	8	0.00882
G1 phase	49	0.226	4	0.00882
response to endogenous stimulus	1360	6.3	17	0.0114
positive regulation of cell proliferation	786	3.63	12	0.0166
response to organic substance	2500	11.5	24	0.0166

miRNAs from Cancer Type Classification

The five circulating miRNAs identified by the proposed method as the most informative for cancer type classification are hsa-miR-122, hsa-let-7a, hsa-miR-23b, hsa-miR-708, and hsa-miR-200c.

hsa-miR-122 is a 22-nucleotide RNA molecule that plays an important role in liver functions [TG15]. It is related with regulation of cholesterol, fatty acid metabolism, and hepatocytes differentiation. Evidence indicates that hsa-miR-122 acts like a tumor suppressor, and its depletion is related with liver inflammation and hepatocellular cancer in mice [Ban+15; TG15]. In breast cancer, hsa-miR-122 has different expression patterns according to the subtype [Erg+15]. In addition, miR-122 promotes aggression and epithelial-mesenchymal transition in TNBC [WW19], and cell survival in radio-resistance cells [Per+19]. High plasma miR-122 levels have been detected in AFP-producing gastric cancer [Mar+18].

The let-7 miRNAs show a high evolutionary conservation between organisms. Vertebrates have multiple let-7 isoforms and play an important role in development and tumor suppression [Lee+16]. hsa-let-7a is a member of the family and shows a downregulated expression in many tumor types like breast cancer [Kha+18; Liu+15], lung adenocarcinoma [Zha+18] and gastric cancer [Yan+11].

hsa-miR-23b is known to target tumor suppressor and cancer promoter genes. hsa-miR-23b is disregulated in proliferation, invasion, migration, apoptosis, autophagy and cell survival [Gro+18]. As a circulating biomarker hsa-miR-23b is down-regulated in colon cancer measured in plasma [Kou+16]. In contrast, it is up regulated in gastric cancer [Zhu+16], lung cancer [Zhu+17], and pancreatic cancer [Che+17].

hsa-miR-708, also known as miR-708-5p, is a microRNA encoded within an intron of ODZ4 gene. It can be found in different tissues with varying expression patterns like reproductive, secretory, muscle, gastrointestinal, nervous, and lung [ML17]. hsa-miR-708 acts as a tumor suppressor or oncogene according to the cancer type. It has been associated with poor prognosis in lung adenocarcinoma [Jan+12] and carcinogenesis in colon [Fed+16] and bladder [Son+13]. On the other hand, normal levels of hsa-miR-708 decrease cell growth, invasion and increase apoptosis in renal cancer cells [Sai+11].

hsa-miR-200c has been identified in lung, gastric, breast, ovarian and endometrial cancer with different expression patterns related with prognosis, aggressiveness and chemoresistance [Coc+10; Liu+12]. Moreover, hsa-miR-200c is involved in signaling cascades such as TGF-β, PI3K/Akt, Notch, VEGF, and NF-κB making it a candidate biomarker in cancer [Mut+16].

The result with the smallest $p-value$ from the Enrichment Analysis in the KEGG Dataset identified a strong relationship between the P53 signaling pathway and hsa-miR-122, hsa-let-7a, hsa-miR-23b, hsa-miR-708, and hsa-miR-200c. P53 is an important tumor suppressor that regulates the expression of many genes and is one of the most common mutated genes in cancer. Many miRNAs work as direct and indirect mediators of the P53 activity and the components of its pathway [Liu+17; TL09]. Moreover, the normal function of this tumor suppressor helps to the maturation of some miRNAs with growth-suppressing function [Suz+09].

On the other hand, the first result in the Enrichment Analysis in the GO:BP Dataset was cellular stress response. In normal cells, there is a balance between the activation of survival and cell death pathways, according to the type and duration of the stress [Ful+10]. Cancer cells develop molecular mechanisms that facilitate their adaptation to different conditions like oxidative, metabolic, mechanical, and genotoxic stresses, avoiding the restriction of the growth and increasing cell proliferation [CX18]. Importantly, miRNAs have the capacity to modify the stress response in cancer by making cells more susceptible or resistant to chemotherapy [BSW08]. This findings prove that miRNAs play an important role in cancer biology, and could be used as powerful circulating

biomarkers for diagnosis and prognosis in human malignancies.

miRNAs from Triple-Negative Breast Cancer Classification

From the analysis performed, 5 candidate miRNAs are selected as the most informative to separate cancer TNBC from the other subtypes in BRCA: hsa-miR-378*, hsa-miR-221, hsa-miR-342-3p, hsa-miR-630, and hsa-miR-145. All of them had already been shown to have potential as circulating cancer biomarkers in cancer studies, e.g. [Cit+10; Eic+10; He+13; Wan+09; Wei+14; Yan+14; Yin+14; Zha+08; Zho+16].

hsa-miR-378* is considered a onco-miRNA for its relationship with tumor growth and cell-renewal. It is associated with progression of breast cancer and the Warburg effect. Furthermore, hsa-miR-378* is capable of discriminating between breast cancer patients and controls [Eic+10; Yin+14].

Evidence indicates that hsa-miR-221 is up-regulated and its expression is related with proliferative pathways [Che+13; San+18]. Several studies have linked microRNA cluster 221/222 with chemoresistance. The miR-221/222 expression participates in the clinically aggressive basal-like subtype [Sti+11] and tamoxifen resistance in ER-positive breast cancer cells [Wei+14; Zha+08]. Furthermore, this cluster interfers with ERα expression [Wei+14] and miR-221/222 knockdown induces to growth arrest and apoptosis in cells exposed to tamoxifen [Zha+08].

On the other hand, hsa-miR-342-3p expression correlates with ERα mRNA expression and its downregulation is related with tamoxifen resistance. hsa-miR-342-3p plays an important role to therapy response of tamoxifen in ER-positive breast cancer [Cit+10; He+13]. Moreover, hsa-miR-342-3p activity affects some metabolic pathways like lactate and glucose fluxes in TNBC [Rom+18].

hsa-miR-630 is considerably suppressed in BRCA [Zho+16]. From in-vitro experiments, in which hsa-miR-630 mimics was transfected into MDA-MB-231 cells, it could be detected that the expression of hsa-miR-630 was decreased. miR-630 was also capable to inhibit MDA-MB-231 cells migration and invasion targeting SOX4-3'-UT. Additionally, SOX4 over expression plasmid was transfected to further confirm hsa-miR-630 played its role by down-regulation [Liu+16].

Finally, hsa-miR-145 acts as a tumor suppressor through the inhibition of different proteins like ERBB3 and RTKN [Wan+09; Yan+14]. Additionally, hsa-miR-145 cooperates with P53 and has a pro-apoptotic effect in patients with breast cancer [Spi+10].

The miRNet enrichment analysis yields that P53 and the negative regulation of cell proliferation were the signaling pathways mostly involved with these miRNAs. Furthermore, the MTDH, IGF1R and CDK6 genes are targeted by at least 3 of the 5 miRNAs used to identify TNBC. Zare et al. [Zar+18] described the interplay of methilation patterns in miRNAs and the epithelial-mesenchymal transition. They identified that some genes like MTDH, IGF1R and CDK6 can be affected by miRNAs and modify cellular processes in breast cancer.

5.2.5 Abbreviations

Adrenocortical carcinoma ACC
Bladder Urothelial Carcinoma BLCA
Breast invasive carcinoma BRCA
Cervical squamous cell carcinoma CESC
Cholangiocarcinoma CHOL
Lymphoid Neoplasm Diffuse Large B-cell Lymphoma DLBC
Esophageal carcinoma ESCA

Head and Neck squamous cell carcinoma HNSC
Kidney Chromophobe KICH
Kidney renal clear cell carcinoma KIRC
Kidney renal papillary cell carcinoma KIRP
Lower Grade Glioma LGG
Liver hepatocellular carcinoma LIHC
Lung adenocarcinoma LUAD
Lung squamous cell carcinoma LUSC
Mesothelioma MESO
Pancreatic adenocarcinoma PAAD
Pheochromocytoma and Paraganglioma PCPG
Prostate adenocarcinoma PRAD
Sarcoma SARC
Skin Cutaneous Melanoma SKCM
Stomach adenocarcinoma STAD
Testicular Germ Cell Tumors TGCT
Thyroid carcinoma THCA
Thymoma THYM
Uterine Corpus Endometrial Carcinoma UCEC
Uterine Carcinosarcoma UCS
Uveal Melanoma UVM
microRNA miRNA
The Cancer Genome Atlas TCGA
Gene Expression Omnibus GEO
Stochastic Gradient Descent SGD
Support Vector Machines Classifier SVC
Read Per Million RPM
Normal Tissue NT
Tumor Tissue TT
Root Mean Squared Error RMSE
Principal Component Analysis PCA
Univariate Feature Selection UFS
Recursive Feature Elimination RFE
Elastic Net EN
Least Absolute Shrinkage and Selection Operator LASSO
Ensemble Feature Selection with Complete Linear Aggregation EFS-CLA
Triple Negative Breast Cancer TNBC
Luminal A LumA
Luminal B LumB

5.2.6 Acknowledgements

The results published here are based upon data generated by The Cancer Genome Atlas and GSM.

Bibliography

[Abe+09] Thomas Abeel et al. “Robust biomarker identification for cancer diagnosis with ensemble feature selection methods”. In: *Bioinformatics* 26.3 (2009), pages 392–398 (cited on pages 193, 194, 203, 212).

[Afz20] Adeel Afzal. “Molecular diagnostic technologies for COVID-19: Limitations and challenges”. In: *Journal of advanced research* (2020) (cited on page 175).

[Akh+15] Most Mauluda Akhtar et al. “Bioinformatic tools for microRNA dissection”. In: *Nucleic acids research* 44.1 (2015), pages 24–44 (cited on page 193).

[AK17] Naomi Altman and Martin Krzywinski. “Ensemble methods: Bagging and random forests”. In: *Nature Methods* 14.10 (Sept. 2017), pages 933–934 (cited on page 200).

[BSW08] Imran A Babar, Frank J Slack, and Joanne B Weidhaas. “miRNA modulation of the cellular stress response”. In: *Future Oncology* (2008) (cited on page 226).

[Ban+15] Simonetta Bandiera et al. “miR-122–a key factor and therapeutic target in liver disease”. In: *Journal of hepatology* 62.2 (2015), pages 448–457 (cited on page 226).

[BT09] Claudine L Bartels and Gregory J Tsongalis. “MicroRNAs: novel biomarkers for human cancer”. In: *Clinical chemistry* 55.4 (2009), pages 623–631 (cited on page 193).

[BAB14] Niccolò Bassani, Federico Ambrogi, and Elia Biganzoli. “Assessing agreement between miRNA microarray platforms”. In: *Microarrays* 3.4 (2014), pages 302–321 (cited on pages 198, 210).

[Bas+18] Amrita Basu et al. “RWEN: Response-Weighted Elastic Net For Prediction of Chemosensitivity of Cancer Cell Lines.” In: *Bioinformatics* 1 (2018), page 8 (cited on page 203).

[Bei13] Beijing Institute of Genomics, Chinese Academy of Science. *China National Center for Bioinformation & National Genomics Data Center*. `https://bigd.big.ac.cn/ncov/?lang=en`. Online; accessed 27 January 2020. 2013 (cited on pages 179, 181, 189).

[BZC12] Anindya Bhattacharya, Jesse D Ziebarth, and Yan Cui. "SomamiR: a database for somatic mutations impacting microRNA function in cancer". In: *Nucleic acids research* 41.D1 (2012), pages D977–D982 (cited on page 193).

[Bor+13] Grzegorz M Boratyn et al. "BLAST: a more efficient report with usability improvements". In: *Nucleic acids research* 41.W1 (2013), W29–W33 (cited on page 177).

[Bre99] Leo Breiman. "Pasting small votes for classification in large databases and on-line". In: *Machine Learning* 36.1-2 (1999), pages 85–103 (cited on page 194).

[Bre01] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pages 5–32 (cited on page 194).

[Bre+84] Leo Breiman et al. *Classification and regression trees*. CRC press, 1984 (cited on page 194).

[Cai+17] Qian Qian Cai et al. "MiR-124 inhibits the migration and invasion of human hepatocellular carcinoma cells by suppressing integrin αV expression". In: *Scientific reports* 7 (2017), page 40733 (cited on page 209).

[Cal+02] George Adrian Calin et al. "Frequent deletions and down-regulation of micro-RNA genes miR15 and miR16 at 13q14 in chronic lymphocytic leukemia". In: *Proceedings of the National Academy of Sciences* 99.24 (2002), pages 15524–15529 (cited on page 192).

[CLG13] Federica Calore, Francesca Lovat, and Michela Garofalo. "Non-coding RNAs and cancer". In: *International journal of molecular sciences* 14.8 (2013), pages 17085–17110 (cited on pages 193, 209, 210).

[Cas+14] Irene Casanova-Salas et al. "Identification of miR-187 and miR-182 as biomarkers of early diagnosis and prognosis in patients with prostate cancer treated with radical prostatectomy". In: *The Journal of urology* 192.1 (2014), pages 252–259 (cited on pages 205, 212).

[CCZ16] Caterina Catalanotto, Carlo Cogoni, and Giuseppe Zardo. "MicroRNA in control of gene expression: an overview of nuclear functions". In: *International journal of molecular sciences* 17.10 (2016), page 1712 (cited on page 210).

[Che+03] Chris Cheadle et al. "Analysis of microarray data using Z score transformation". In: *The Journal of molecular diagnostics* 5.2 (2003), pages 73–81 (cited on page 198).

[Che+17] Dayang Chen et al. "Upregulated exosomic miR-23b-3p plays regulatory roles in the progression of pancreatic cancer". In: *Oncology reports* 38.4 (2017), pages 2182–2188 (cited on page 226).

[CX18] Miao Chen and Songbo Xie. "Therapeutic targeting of cellular stress responses in cancer". In: *Thoracic cancer* 9.12 (2018), pages 1575–1582 (cited on page 226).

[Che+13] Wei-Xian Chen et al. "miR-221/222: promising biomarkers for breast cancer". In: *Tumor Biology* 34.3 (2013), pages 1361–1370 (cited on page 227).

[Che+09] Yongxin Chen et al. "Reproducibility of quantitative RT-PCR array in miRNA expression profiling and comparison with microarray analysis". In: *BMC genomics* 10.1 (2009), page 407 (cited on page 209).

[Che15] Guofeng Cheng. “Circulating miRNAs: roles in cancer diagnosis, prognosis and therapy”. In: *Advanced drug delivery reviews* 81 (2015), pages 75–93 (cited on pages 209, 210).

[Chu+15] Andy Chu et al. “Large-scale profiling of microRNAs for the cancer genome atlas”. In: *Nucleic acids research* 44.1 (2015), e3–e3 (cited on page 198).

[Cit+10] Diana M Cittelly et al. “Downregulation of miR-342 is associated with tamoxifen resistant breast tumors”. In: *Molecular cancer* 9.1 (2010), page 317 (cited on page 227).

[Coc+10] Dawn R Cochrane et al. “Loss of miR-200c: a marker of aggressiveness and chemoresistance in female reproductive cancers”. In: *Journal of oncology* 2010 (2010) (cited on page 226).

[Col+15] Antonio Colaprico et al. “TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data”. In: *Nucleic acids research* 44.8 (2015), e71–e71 (cited on page 206).

[CBS07] Manuel Collado, Maria A Blasco, and Manuel Serrano. “Cellular senescence in cancer and aging”. In: *Cell* 130.2 (2007), pages 223–233 (cited on page 218).

[Cor+20] Victor M Corman et al. “Detection of 2019 novel coronavirus (2019-nCoV) by real-time RT-PCR”. In: *Eurosurveillance* 25.3 (2020) (cited on page 185).

[Cor+11] Maria Angelica Cortez et al. “MicroRNAs in body fluids–the mix of hormones and biomarkers”. In: *Nature reviews Clinical oncology* 8.8 (2011), page 467 (cited on page 193).

[Cox58] David R Cox. “The regression analysis of binary sequences”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1958), pages 215–242 (cited on page 194).

[Cra+06] Koby Crammer et al. “Online passive-aggressive algorithms”. In: *Journal of Machine Learning Research* 7.Mar (2006), pages 551–585 (cited on page 194).

[Dai+15] Xiaofeng Dai et al. “Breast cancer intrinsic subtype classification, clinical use and future trends”. In: *American journal of cancer research* 5.10 (2015), page 2929 (cited on page 220).

[Del+13] Valerio Del Vescovo et al. “A cross-platform comparison of affymetrix and Agilent microarrays reveals discordant miRNA expression in lung tumors of c-Raf transgenic mice”. In: *PloS one* 8.11 (2013), e78870 (cited on pages 198, 210).

[Dha+17] Andrew Dhawan et al. “sigQC: A procedural approach for standardising the evaluation of gene signatures”. In: *bioRxiv* (2017), page 203729 (cited on page 210).

[DP21] Centre for Disease and Control Prevention. *Emergence of SARS-CoV-2 B.1.1.7 Lineage — United States, December 29, 2020–January 12, 2021*. Jan. 2021 (cited on page 190).

[EDL02] Ron Edgar, Michael Domrachev, and Alex E Lash. “Gene Expression Omnibus: NCBI gene expression and hybridization array data repository”. In: *Nucleic acids research* 30.1 (2002), pages 207–210 (cited on page 210).

[Eic+10] Lillian J Eichner et al. “miR-378 mediates metabolic shift in breast cancer cells via the PGC-1β/ERRγ transcriptional pathway”. In: *Cell metabolism* 12.4 (2010), pages 352–361 (cited on page 227).

[Elg+14] Bente Vilming Elgaaen et al. “Global miRNA expression analysis of serous and clear cell ovarian carcinomas identifies differentially expressed miRNAs including miR-200c-3p as a prognostic marker”. In: *BMC cancer* 14.1 (2014), page 80 (cited on page 212).

[Erg+15] Sercan Ergün et al. “The association of the expression of miR-122-5p and its target ADAM10 with human breast cancer”. In: *Molecular biology reports* 42.2 (2015), pages 497–505 (cited on page 226).

[Fab13] Muller Fabbri. *Non-coding RNAs and cancer*. Springer, 2013 (cited on page 193).

[FG06] Seth Falcon and Robert Gentleman. “Using GOstats to test gene lists for GO term association”. In: *Bioinformatics* 23.2 (2006), pages 257–258 (cited on page 212).

[Fan+16] Yannan Fan et al. “miRNet-dissecting miRNA-target interactions and functional associations through network-based visual analysis”. In: *Nucleic acids research* 44.W1 (2016), W135–W141 (cited on page 212).

[Far21] Nuno Faria. *Genomic characterisation of an emergent SARS-CoV-2 lineage in Manaus: preliminary findings*. Jan. 2021 (cited on page 190).

[Fed+16] Paola Fernanda Fedatto et al. “MiR-708-5p as a Predictive Marker of Colorectal Cancer Prognosis”. In: *Journal of Analytical Oncology* 5.1 (2016), pages 14–23 (cited on page 226).

[Fer+15] Jacques Ferlay et al. “Cancer incidence and mortality worldwide: sources, methods and major patterns in GLOBOCAN 2012”. In: *International journal of cancer* 136.5 (2015), E359–E386 (cited on page 192).

[FVN10] Manuela Ferracin, Angelo Veronese, and Massimo Negrini. “Micromarkers: miRNAs in cancer diagnosis and prognosis”. In: *Expert review of molecular diagnostics* 10.3 (2010), pages 297–308 (cited on page 193).

[FHT10] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. “Regularization paths for generalized linear models via coordinate descent”. In: *Journal of statistical software* 33.1 (2010), page 1 (cited on page 203).

[Fri01] Jerome H Friedman. “Greedy function approximation: a gradient boosting machine”. In: *Annals of statistics* (2001), pages 1189–1232 (cited on page 194).

[Fuk80] Kunihiko Fukushima. “Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position”. In: *Biological Cybernetics* 4.36 (1980), pages 193–202 (cited on page 177).

[Ful+10] Simone Fulda et al. “Cellular stress responses: cell survival and cell death”. In: *International journal of cell biology* 2010 (2010) (cited on page 226).

[Gao+11] Wen Gao et al. “MiR-21 overexpression in human primary squamous cell lung carcinoma is associated with poor patient prognosis”. In: *Journal of cancer research and clinical oncology* 137.4 (2011), pages 557–566 (cited on page 193).

[Gir+13] María Dolores Giráldez et al. “Circulating microRNAs as biomarkers of colorectal cancer: results from a genome-wide profiling and validation study”. In: *Clinical gastroenterology and hepatology* 11.6 (2013), pages 681–688 (cited on page 209).

[Giu+17] M Giulietti et al. “Identification of candidate miRNA biomarkers for pancreatic ductal adenocarcinoma by weighted gene co-expression network analysis”. In: *Cellular Oncology* 40.2 (2017), pages 181–192 (cited on page 209).

[Gro+18] Ilaria Grossi et al. “Functional role of microRNA-23b-3p in cancer biology”. In: *MicroRNA* 7.3 (2018), pages 156–166 (cited on page 226).

[Guy+02] Isabelle Guyon et al. “Gene selection for cancer classification using support vector machines”. In: *Machine learning* 46.1-3 (2002), pages 389–422 (cited on page 203).

[Had+18] James Hadfield et al. “Nextstrain: real-time tracking of pathogen evolution”. In: *Bioinformatics* 34.23 (2018), pages 4121–4123 (cited on page 190).

[HMK03] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. “Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)”. In: *Evolutionary computation* 11.1 (2003), pages 1–18 (cited on page 199).

[He+13] Yue-Jun He et al. “miR-342 is associated with estrogen receptor-α expression and response to tamoxifen in breast cancer”. In: *Experimental and therapeutic medicine* 5.3 (2013), pages 813–818 (cited on page 227).

[He+15] Yuqing He et al. “Current state of circulating microRNAs as cancer biomarkers”. In: *Clinical chemistry* (2015), clinchem–2015 (cited on pages 193, 209, 210).

[He+17] Zhonghui He et al. “miR-944 acts as a prognostic marker and promotes the tumor progression in endometrial cancer”. In: *Biomedicine & Pharmacotherapy* 88 (2017), pages 902–910 (cited on page 210).

[Hea+98] Marti A. Hearst et al. “Support vector machines”. In: *IEEE Intelligent Systems and their Applications* 13.4 (1998), pages 18–28 (cited on page 194).

[Her+17] Simon K Hermansen et al. “A 4-miRNA signature to predict survival in glioblastomas”. In: *PloS one* 12.11 (2017), e0188090 (cited on page 212).

[HG15] Zena M Hira and Duncan F Gillies. “A review of feature selection and feature extraction methods applied on microarray data”. In: *Advances in bioinformatics* 2015 (2015) (cited on page 203).

[Hsu+12] Cheng-Ming Hsu et al. “Circulating miRNA is a novel marker for head and neck squamous cell carcinoma”. In: *Tumor Biology* 33.6 (2012), pages 1933–1942 (cited on page 209).

[IC12] Marilena V Iorio and Carlo M Croce. “MicroRNA dysregulation in cancer: diagnostics, monitoring and therapeutics. A comprehensive review”. In: *EMBO molecular medicine* 4.3 (2012), pages 143–159 (cited on page 193).

[Iri+03] Rafael A Irizarry et al. “Exploration, normalization, and summaries of high density oligonucleotide array probe level data”. In: *Biostatistics* 4.2 (2003), pages 249–264 (cited on page 198).

[Jan+17] Hee-Jin Jang et al. “Integrated genomic analysis of recurrence-associated small non-coding RNAs in oesophageal cancer”. In: *Gut* 66.2 (2017), pages 215–225 (cited on pages 205, 212).

[Jan+12] Jin Sung Jang et al. “Increased miR-708 expression in NSCLC and its association with poor survival in lung adenocarcinoma from never smokers”. In: *Clinical Cancer Research* 18.13 (2012), pages 3658–3667 (cited on page 226).

[Jep+18] Rikke Karlin Jepsen et al. “Early metastatic colorectal cancers show increased tissue expression of miR-17/92 cluster members in the invasive tumor front”. In: *Human pathology* 80 (2018), pages 231–238 (cited on page 212).

[Jia+16] Xiumei Jiang et al. “Serum microRNA expression signatures as novel noninvasive biomarkers for prediction and prognosis of muscle-invasive bladder cancer”. In: *Oncotarget* 7.24 (2016), page 36733 (cited on page 209).

[Jia+18] Yanxia Jiang et al. “The diagnostic and prognostic value of plasma microRNA-125b-5p in patients with multiple myeloma”. In: *Oncology letters* 16.3 (2018), pages 4001–4007 (cited on page 209).

[JRF12] Giuseppe Jurman, Samantha Riccadonna, and Cesare Furlanello. “A comparison of MCC and CEN error measures in multi-class prediction”. In: *PloS one* 7.8 (2012) (cited on page 214).

[KLS+09] Ruslan Kalendar, David Lee, Alan H Schulman, et al. “FastPCR software for PCR primer and probe design and repeat search”. In: *Genes, Genomes and Genomics* 3.1 (2009), pages 1–14 (cited on page 184).

[Kan+16] Minoru Kanehisa et al. “KEGG: new perspectives on genomes, pathways, diseases and drugs”. In: *Nucleic acids research* 45.D1 (2016), pages D353–D361 (cited on page 212).

[Kan+06] Noriyuki Kanzawa et al. “Augmentation of chemokine production by severe acute respiratory syndrome coronavirus 3a/X1 and 7a/X4 proteins through NF-κB activation”. In: *FEBS letters* 580.30 (2006), pages 6807–6812 (cited on page 183).

[Kha+18] Solmaz Khalighfard et al. “Plasma miR-21, miR-155, miR-10b, and Let-7a as the potential biomarkers for the monitoring of breast cancer patients”. In: *Scientific reports* 8.1 (2018), page 17981 (cited on page 226).

[KB14] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cited on page 178).

[Kog+10] Yoshikatsu Koga et al. “MicroRNA expression profiling of exfoliated colonocytes isolated from feces for colorectal cancer screening”. In: *Cancer prevention research* (2010), pages 1940–6207 (cited on page 209).

[Kou+16] Chang-Hua Kou et al. “Downregulation of mir-23b in plasma is associated with poor prognosis in patients with colorectal cancer”. In: *Oncology letters* 12.6 (2016), pages 4838–4844 (cited on page 226).

[KG10] Ana Kozomara and Sam Griffiths-Jones. “miRBase: integrating microRNA annotation and deep-sequencing data”. In: *Nucleic acids research* 39.suppl_1 (2010), pages D152–D157 (cited on page 193).

[Lam+15] Monica Lamberti et al. “Two different serum miRNA signatures correlate with the clinical outcome and histological subtype in pleural malignant mesothelioma patients”. In: *PloS one* 10.8 (2015), e0135331 (cited on page 210).

[Lar+16] Erika Larrea et al. “New concepts in cancer biomarkers: circulating miRNAs in liquid biopsies”. In: *International journal of molecular sciences* 17.5 (2016), page 627 (cited on pages 209, 210).

[LR78] A Verdugo Lazo and P Rathie. “On the entropy of continuous probability distributions (corresp.)” In: *IEEE Transactions on Information Theory* 24.1 (1978), pages 120–122 (cited on page 203).

[Lee+16] Hosuk Lee et al. “Biogenesis and regulation of the let-7 miRNAs and their functional implications”. In: *Protein & cell* 7.2 (2016), pages 100–113 (cited on page 226).

[Les+13] Dena Leshkowitz et al. “Differences in microRNA detection levels are technology and sequence dependent”. In: *Rna* 19.4 (2013), pages 527–538 (cited on pages 198, 210).

[LB10] Kelvin Li and Anushka Brownley. “Primer design for RT-PCR”. In: *RT-PCR Protocols*. Springer, 2010, pages 271–299 (cited on page 177).

[LR09] Wei Li and Kangcheng Ruan. “MicroRNA detection by microarray”. In: *Analytical and bioanalytical chemistry* 394.4 (2009), pages 1117–1124 (cited on page 209).

[Li+14] Xiangqi Li et al. “A convenient system for highly specific and sensitive detection of miRNA expression”. In: *RNA* 20.2 (2014), pages 252–259 (cited on page 207).

[Lin+13] Pei-Chun Lin et al. “Epigenetic repression of miR-31 disrupts androgen receptor homeostasis and contributes to prostate cancer progression”. In: *Cancer research* 73.3 (2013), pages 1232–1244 (cited on pages 205, 212).

[LLC12] Bing Liu, Jiuyong Li, and Murray J Cairns. “Identifying miRNAs, targets and functions”. In: *Briefings in bioinformatics* 15.1 (2012), pages 1–19 (cited on page 193).

[Liu+17] Juan Liu et al. “MicroRNA control of p53”. In: *Journal of cellular biochemistry* 118.1 (2017), pages 7–14 (cited on page 226).

[Liu+15] Kui Liu et al. “Let-7a inhibits growth and migration of breast cancer cells by targeting HMGA1”. In: *International journal of oncology* 46.6 (2015), pages 2526–2534 (cited on page 226).

[Liu+12] Xiao-Guang Liu et al. “High expression of serum miR-21 and tumor miR-200c associated with poor prognosis in patients with lung cancer”. In: *Medical oncology* 29.2 (2012), pages 618–626 (cited on page 226).

[Liu+16] Yun-Xiao Liu et al. “MiR-630 inhibits cells migration and invasion by targeting SOX4 in triple-negative breast cancer”. In: *Int J Clin Exp Pathol* 9.9 (2016), pages 9097–9105 (cited on page 227).

[Lop+21a] A. Lopez-Rincon et al. “Design of Specific Primer Sets for SARS-CoV-2 Variants Using Evolutionary Algorithms”. In: *Proceedings of the 2021 Genetic and Evolutionary Computation Conference, In print*. ACM, July 2021 (cited on pages 175, 186).

[Lop+18] Alejandro Lopez-Rincon et al. “Evolutionary optimization of convolutional neural networks for cancer miRNA biomarkers classification”. In: *Applied Soft Computing* 65 (Apr. 2018), pages 91–100 (cited on page 175).

[Lop+19] Alejandro Lopez-Rincon et al. “Automatic discovery of 100-miRNA signature for cancer classification using ensemble feature selection”. In: *BMC Bioinformatics* 20.1 (Sept. 2019) (cited on pages 175, 181).

[Lop+20] Alejandro Lopez-Rincon et al. “Machine Learning-Based Ensemble Recursive Feature Selection of Circulating miRNAs for Cancer Tumor Classification”. In: *Cancers* 12.7 (July 2020), page 1785 (cited on pages 175, 181).

[Lop+21b] Alejandro Lopez-Rincon et al. “Classification and specific primer design for accurate detection of SARS-CoV-2 using deep learning”. In: *Scientific Reports* 11.1 (Jan. 2021) (cited on pages 175, 186, 190).

[Man10] Jayawant N Mandrekar. “Receiver operating characteristic curve in diagnostic test assessment”. In: *Journal of Thoracic Oncology* 5.9 (2010), pages 1315–1316 (cited on page 222).

[Mar+15] Christiane Margue et al. “Comparison of a healthy miRNome with melanoma patient miRNomes: are microRNAs suitable serum biomarkers for cancer?” In: *Oncotarget* 6.14 (2015), page 12110 (cited on page 209).

[Mar+18] Suguru Maruyama et al. “miR-122-5p as a novel biomarker for alpha-fetoprotein-producing gastric cancer”. In: *World journal of gastrointestinal oncology* 10.10 (2018), page 344 (cited on page 226).

[Mat+15] Nerea Matamala et al. “Tumor microRNA expression profiling identifies circulating microRNAs for early breast cancer detection”. In: *Clinical chemistry* (2015), clinchem–2015 (cited on page 209).

[Med+14] V Medina-Villaamil et al. “Circulating MicroRNAs in blood of patients with prostate cancer”. In: *Actas Urológicas Españolas (English Edition)* 38.10 (2014), pages 633–639 (cited on page 209).

[Men+13] Lourdes Mengual et al. “Using microRNA profiling in urine samples to develop a non-invasive test for bladder cancer”. In: *International journal of cancer* 133.11 (2013), pages 2631–2641 (cited on page 209).

[Miz07] Ilene Mizrachi. “GenBank: the nucleotide sequence database”. In: *The NCBI Handbook [Internet], updated* 22 (2007) (cited on page 180).

[Mon+18] Ruth Montalbo et al. “Prognostic value of circulating microRNAs in upper tract urinary carcinoma”. In: *Oncotarget* 9.24 (2018), page 16691 (cited on page 209).

[ML17] Nicholas J Monteleone and Carol S Lutz. “miR-708-5p: a microRNA with emerging roles in cancer”. In: *Oncotarget* 8.41 (2017), page 71292 (cited on page 226).

[Mun+09] MK Muniyappa et al. “MiRNA-29a regulates the expression of numerous proteins and reduces the invasiveness and proliferation of human carcinoma cell lines”. In: *European Journal of Cancer* 45.17 (2009), pages 3104–3118 (cited on page 210).

[Mur+13] Yoshiki Murakami et al. “The expression level of miR-18b in hepatocellular carcinoma is associated with the grade of malignancy and prognosis”. In: *BMC cancer* 13.1 (2013), page 99 (cited on page 205).

[Mut+16] Merve Mutlu et al. “miR-200c: a versatile watchdog in cancer progression, EMT, and drug resistance”. In: *Journal of Molecular Medicine* 94.6 (2016), pages 629–644 (cited on page 226).

[NH10] Vinod Nair and Geoffrey E Hinton. “Rectified linear units improve restricted boltzmann machines”. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pages 807–814 (cited on page 177).

[Net+12] Cancer Genome Atlas Network et al. “Comprehensive molecular portraits of human breast tumours”. In: *Nature* 490.7418 (2012), page 61 (cited on page 206).

[Org+20] World Health Organization et al. *Molecular assays to diagnose COVID-19: summary table of available protocols*. 2020 (cited on page 175).

[Pad+07] Kartika Padhan et al. “Severe acute respiratory syndrome coronavirus Orf3a protein interacts with caveolin”. In: *Journal of General Virology* 88.11 (2007), pages 3067–3077 (cited on page 183).

[Pan+17] Tao Pan et al. “miR-944 inhibits metastasis of gastric cancer by preventing the epithelial–mesenchymal transition via MACC1/Met/AKT signaling”. In: *FEBS open bio* 7.7 (2017), pages 905–914 (cited on page 210).

[Ped+11] Fabian Pedregosa et al. “Scikit-learn: Machine learning in Python”. In: *Journal of Machine Learning Research* 12.Oct (2011), pages 2825–2830 (cited on page 196).

[PBH08] Elizabeth van Pelt-Verkuil, Alex van Belkum, and John P. Hays. “PCR Primers”. In: *Principles and Technical Aspects of PCR Amplification*. Springer Netherlands, 2008, pages 63–90 (cited on page 176).

[Peñ+14] Maria Peña-Chilet et al. “MicroRNA profile in very young women with breast cancer”. In: *BMC cancer* 14.1 (2014), page 529 (cited on pages 205, 212).

[PC16] Yong Peng and Carlo M Croce. “The role of MicroRNAs in human cancer”. In: *Signal transduction and targeted therapy* 1 (2016), page 15004 (cited on page 192).

[Per+19] Isidro X Perez-Añorve et al. “New insights into radioresistance in breast cancer identify a dual function of miR-122 as a tumor suppressor and oncomiR”. In: *Molecular oncology* 13.5 (2019), pages 1249–1267 (cited on page 226).

[PPR17] Yongjun Piao, Minghao Piao, and Keun Ho Ryu. “Multiclass cancer classification using a feature subset-based ensemble from microRNA expression profiles”. In: *Computers in biology and medicine* 80 (2017), pages 39–44 (cited on page 193).

[Ran+20] Gurjit S Randhawa et al. “Machine learning using intrinsic genomic signatures for rapid classification of novel pathogens: COVID-19 case study”. In: *Plos one* 15.4 (2020), e0232391 (cited on page 183).

[Rin+18] Alejandro Lopez Rincon et al. “Evolutionary Optimization of Convolutional Neural Networks for Cancer miRNA Biomarkers Classification”. In: *Applied Soft Computing* (2018). ISSN: 1568-4946 (cited on page 194).

[Rod+15] Alicia Rodríguez et al. “Design of primers and probes for quantitative real-time PCR methods”. In: *PCR Primer Design*. Springer, 2015, pages 31–56 (cited on page 177).

[Rom+18] Sandra L Romero-Cordoba et al. “Loss of function of miR-342-3p results in MCT1 over-expression and contributes to oncogenic metabolic reprogramming in triple negative breast cancer”. In: *Scientific reports* 8.1 (2018), page 12252 (cited on pages 205, 212, 222, 227).

[SAV08] Yvan Saeys, Thomas Abeel, and Yves Van de Peer. “Robust feature selection using ensemble feature selection techniques”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2008, pages 313–325 (cited on pages 193, 194).

[Sai+11] Sharanjot Saini et al. “MicroRNA-708 induces apoptosis and suppresses tumorigenicity in renal cancer cells”. In: *Cancer research* 71.19 (2011), pages 6208–6219 (cited on page 226).

[San+18] Maria Francesca Santolla et al. “miR-221 stimulates breast cancer cells and cancer-associated fibroblasts (CAFs) through selective interference with the A20/c-Rel/CTGF signaling”. In: *Journal of Experimental & Clinical Cancer Research* 37.1 (2018), page 94 (cited on page 227).

[20] *SARS-CoV-2 Variants*. Dec. 2020 (cited on page 190).

[SH17] Srinivasulu Yerukala Sathipati and Shinn-Ying Ho. “Identifying the miRNA signature associated with survival time in patients with lung adenocarcinoma using miRNA expression profiles”. In: *Scientific reports* 7.1 (2017), page 7507 (cited on page 210).

[SP11] Edward R Sauter and Neel Patel. “Body fluid micro (mi) RNAs as biomarkers for human cancer”. In: *Journal of Nucleic Acids Investigation* 2.1 (2011), page 1 (cited on page 193).

[Sch+14] Anna-Regina Scheffer et al. “Circulating microRNAs in serum: novel biomarkers for patients with bladder cancer?” In: *World journal of urology* 32.2 (2014), pages 353–358 (cited on page 209).

[Sei+17] B. Seijo-Pardo et al. “Ensemble feature selection: Homogeneous and heterogeneous approaches”. In: *Knowledge-Based Systems* 118 (Feb. 2017), pages 124–139 (cited on pages 193, 194, 203).

[She+01] Stephen T Sherry et al. “dbSNP: the NCBI database of genetic variation”. In: *Nucleic acids research* 29.1 (2001), pages 308–311 (cited on pages 181–183, 189).

[Shi+19] Chong-Shan Shi et al. “SARS-Coronavirus Open Reading Frame-8b triggers intracellular stress pathways and activates NLRP3 inflammasomes”. In: *Cell death discovery* 5.1 (2019), pages 1–12 (cited on page 183).

[Shi+15] Vivian Yvonne Shin et al. “A three-miRNA signature as promising non-invasive diagnostic marker for gastric cancer”. In: *Molecular cancer* 14.1 (2015), page 202 (cited on page 209).

[SM17] Yuelong Shu and John McCauley. “GISAID: Global initiative on sharing all influenza data–from vision to reality”. In: *Eurosurveillance* 22.13 (2017) (cited on page 183).

[Šim09] Ana-Maria Šimundić. “Measures of diagnostic accuracy: basic definitions”. In: *Ejifcc* 19.4 (2009), page 203 (cited on page 222).

[Sok+16] Artem Sokolov et al. “Pathway-based genomics prediction using generalized elastic net”. In: *PLoS computational biology* 12.3 (2016), e1004790 (cited on page 203).

[Son+13] Tao Song et al. “miR-708 promotes the development of bladder carcinoma via direct repression of Caspase-2”. In: *Journal of cancer research and clinical oncology* 139.7 (2013), pages 1189–1198 (cited on page 226).

[Spi+10] R Spizzo et al. “miR-145 participates with TP53 in a death-promoting regulatory loop and targets estrogen receptor-α in human breast cancer cells”. In: *Cell Death & Differentiation* 17.2 (2010), pages 246–254 (cited on page 227).

[Ste12] Alexander H Stegh. “Targeting the p53 signaling pathway in cancer therapy–the promises, challenges and perils”. In: *Expert opinion on therapeutic targets* 16.1 (2012), pages 67–83 (cited on page 218).

[Sti+11] Susanna Stinson et al. “TRPS1 targeting by miR-221/222 promotes the epithelial-to-mesenchymal transition in breast cancer”. In: *Science signaling* 4.177 (2011), ra41–ra41 (cited on page 227).

[Sum+15] I Summerer et al. “Circulating microRNAs as prognostic therapy biomarkers in head and neck cancer patients”. In: *British journal of cancer* 113.1 (2015), page 76 (cited on page 209).

[Suz+09] Hiroshi I Suzuki et al. “Modulation of microRNA processing by p53”. In: *Nature* 460.7254 (2009), pages 529–533 (cited on page 226).

[TL09] Apana Takwi and Yong Li. “The p53 pathway encounters the microRNA world”. In: *Current genomics* 10.3 (2009), pages 194–197 (cited on page 226).

[Tan+14] Youwen Tan et al. “A serum microRNA panel as potential biomarkers for hepatocellular carcinoma related with hepatitis B virus”. In: *PloS one* 9.9 (2014), e107986 (cited on page 209).

[TOB11] Cristiana Tanase, Irina Ogrezeanu, and Corin Badiu. *Molecular pathology of pituitary adenomas*. Elsevier, 2011 (cited on page 192).

[Tan+17] Wei Tang et al. “Tumor origin detection with tissue-specific miRNA and DNA methylation markers”. In: *Bioinformatics* 34.3 (2017), pages 398–406 (cited on page 193).

[Teg+20] Houriiyah Tegally et al. “Emergence and rapid spread of a new severe acute respiratory syndrome-related coronavirus 2 (SARS-CoV-2) lineage with multiple spike mutations in South Africa”. In: *medRxiv* (2020) (cited on page 190).

[Tel+17] Aristeidis G Telonis et al. “Knowledge about the presence or absence of miRNA isoforms (isomiRs) can successfully discriminate amongst 32 TCGA cancer types”. In: *Nucleic acids research* 45.6 (2017), pages 2973–2985 (cited on pages 193, 199, 206).

[TG15] Sharda Thakral and Kalpana Ghoshal. “miR-122 is a unique molecule with great potential in diagnosis, prognosis of liver disease, and therapy both as miRNA mimic and antimir”. In: *Current gene therapy* 15.2 (2015), pages 142–150 (cited on page 226).

[Tib96] Robert Tibshirani. “Regression shrinkage and selection via the lasso”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pages 267–288 (cited on page 203).

[Tik43] Andrey Nikolayevich Tikhonov. “On the stability of inverse problems”. In: *Dokl. Akad. Nauk SSSR*. Volume 39. 5. 1943, pages 195–198 (cited on page 194).

[TF06] Victor Trevino and Francesco Falciani. “GALGO: an R package for multivariate variable selection using genetic algorithms”. In: *Bioinformatics* 22.9 (2006), pages 1154–1156 (cited on pages 203, 212).

[Tri+16] Violaine Tribollet et al. “miR-135a inhibits the invasion of cancer cells via suppression of ERR alpha”. In: *PloS one* 11.5 (2016), e0156445 (cited on page 209).

[Tsu+15] Naoto Tsuchiya et al. *Method for detecting pancreatic cancer and detection kit*. US Patent App. 14/410,408. July 2015 (cited on page 209).

[Unt+07] Andreas Untergasser et al. “Primer3Plus, an enhanced web interface to Primer3”. In: *Nucleic acids research* 35.suppl_2 (2007), W71–W74 (cited on pages 183, 187, 188).

[Vuc+14] Emily A Vucic et al. “Smoking status impacts microRNA mediated prognosis and lung adenocarcinoma biology”. In: *BMC cancer* 14.1 (2014), page 778 (cited on page 205).

[Waa+20] Mark B van der Waal et al. “Blockchain-facilitated sharing to advance outbreak R&D”. In: *Science* 368.6492 (2020), pages 719–721 (cited on page 175).

[Wan+15] Dahu Wang et al. “Hsa-miR-21 and Hsa-miR-29 in tissue as potential diagnostic and prognostic biomarkers for gastric cancer”. In: *Cellular Physiology and Biochemistry* 37.4 (2015), pages 1454–1462 (cited on page 193).

[Wan+18] Hao Wang et al. “Circulating microRNAs as potential cancer biomarkers: the advantage and disadvantage”. In: *Clinical epigenetics* 10.1 (2018), page 59 (cited on page 209).

[Wan+14a] Jin Wang et al. “Circulating microRNAs in pancreatic juice as candidate biomarkers of pancreatic cancer”. In: *Journal of Cancer* 5.8 (2014), page 696 (cited on page 209).

[Wan+14b] Jin Wang et al. “Tumor-associated circulating microRNAs as biomarkers of cancer”. In: *Molecules* 19.2 (2014), pages 1912–1938 (cited on pages 209, 210).

[Wan+09] Shihua Wang et al. “miR-145 inhibits breast cancer cell growth through RTKN”. In: *International journal of oncology* 34.5 (2009), pages 1461–1466 (cited on page 227).

[Wan+16] Yanbo Wang et al. “miR-124-3p functions as a tumor suppressor in breast cancer by targeting CBL”. In: *BMC cancer* 16.1 (2016), page 826 (cited on page 209).

[WW19] Zheng Wang and Xiangyu Wang. “miR-122-5p promotes aggression and epithelial-mesenchymal transition in triple-negative breast cancer by suppressing charged multivesicular body protein 3 through mitogen-activated protein kinase signaling”. In: *Journal of Cellular Physiology* 0.0 (2019). eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcp.29188` (cited on page 226).

[Wei+14] Yifang Wei et al. “Exosomal miR-221/222 enhances tamoxifen resistance in recipient ER-positive breast cancer cells”. In: *Breast cancer research and treatment* 147.2 (2014), pages 423–431 (cited on page 227).

[Wei+13] John N Weinstein et al. “The cancer genome atlas pan-cancer analysis project”. In: *Nature genetics* 45.10 (2013), page 1113 (cited on page 194).

[Wei13] M Weiss. “Your Guide to the Breast Cancer Pathology Report”. In: *Breastcancer. org* (2013) (cited on pages 206, 222).

[Wen+17] Liqiang Wen et al. “miR-944 inhibits cell migration and invasion by targeting MACC1 in colorectal cancer”. In: *Oncology reports* 37.6 (2017), pages 3415–3422 (cited on page 210).

[Wu+20] Fan Wu et al. “A new coronavirus associated with human respiratory disease in China”. In: *Nature* 579.7798 (2020), pages 265–269 (cited on page 177).

[Yan+14] Xin Yan et al. “miR-143 and miR-145 synergistically regulate ERBB3 to suppress cell proliferation and invasion in breast cancer”. In: *Molecular cancer* 13.1 (2014), page 220 (cited on page 227).

[Yan+08] Li-Xu Yan et al. “MicroRNA miR-21 overexpression in human breast cancer is associated with advanced clinical stage, lymph node metastasis and patient poor prognosis”. In: *Rna* (2008) (cited on page 193).

[Yan+11] Qiaoyuan Yang et al. “Low-level expression of let-7a in gastric cancer and its involvement in tumorigenesis by targeting RAB40C”. In: *Carcinogenesis* 32.5 (2011), pages 713–722 (cited on page 226).

[Yin+14] Jia-Yu Yin et al. “Association between mir-24 and mir-378 in formalin-fixed paraffin-embedded tissues of breast cancer”. In: *International journal of clinical and experimental pathology* 7.7 (2014), page 4261 (cited on page 227).

[YAK16] Malik Yousef, Jens Allmer, and Waleed Khalifa. “Feature selection for microRNA target prediction comparison of one-class feature selection methodologies”. In: (2016) (cited on page 193).

[Zar+18] Maryam Zare et al. “Aberrant miRNA promoter methylation and EMT-involving miRNAs in breast cancer metastasis: diagnosis and therapeutic implications”. In: *Journal of cellular physiology* 233.5 (2018), pages 3729–3744 (cited on page 227).

[Zha04] Tong Zhang. “Solving large scale linear prediction problems using stochastic gradient descent algorithms”. In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, page 116 (cited on page 194).

[Zha+15] XiaoTian Zhang et al. “Overexpression of E2F mRNAs associated with gastric cancer progression identified by the transcription factor and miRNA co-regulatory network analysis”. In: *PloS one* 10.2 (2015), e0116979 (cited on page 212).

[Zha+08] Jian-Jun Zhao et al. “MicroRNA-221/222 negatively regulates estrogen receptorα and is associated with tamoxifen resistance in breast cancer”. In: *Journal of Biological Chemistry* 283.45 (2008), pages 31079–31086 (cited on page 227).

[Zha+18] Wei Zhao et al. “Induction of microRNA-let-7a inhibits lung adenocarcinoma cell growth by regulating cyclin D1”. In: *Oncology reports* 40.4 (2018), pages 1843–1854 (cited on page 226).

[Zha+17] Yuan Zhao et al. “MiR-124 acts as a tumor suppressor by inhibiting the expression of sphingosine kinase 1 and its downstream signaling in head and neck squamous cell carcinoma”. In: *Oncotarget* 8.15 (2017), page 25005 (cited on page 209).

[Zhe+14] Xiao-Hui Zheng et al. “Plasma microRNA profiling in nasopharyngeal carcinoma patients reveals miR-548q and miR-483-5p as potential biomarkers”. In: *Chinese journal of cancer* 33.7 (2014), page 330 (cited on page 209).

[Zhi+10] Feng Zhi et al. “The use of hsa-miR-21, hsa-miR-181b and hsa-miR-106a as prognostic indicators of astrocytoma”. In: *European Journal of Cancer* 46.9 (2010), pages 1640–1649 (cited on page 193).

[Zho+16] Ci-Xiang Zhou et al. “MiR-630 suppresses breast cancer progression by targeting metadherin”. In: *Oncotarget* 7.2 (2016), page 1288 (cited on page 227).

[Zhu+17] Ying Zhu et al. “Identification of a serum microRNA expression signature for detection of lung cancer, involving miR-23b, miR-221, miR-148b and miR-423-3p”. In: *Lung Cancer* 114 (2017), pages 6–11 (cited on page 226).

[Zhu+16] Kun Zhuang et al. “Up-regulation of plasma miR-23b is associated with poor prognosis of gastric cancer”. In: *Medical science monitor: international medical journal of experimental and clinical research* 22 (2016), page 256 (cited on page 226).

6. Perspectives and Conclusions

This chapter concludes the *Habilitation à Diriger des Recherches*, describing the planned follow-ups of the research lines outlined in the previous chapters. More in detail, the work on epistemology of machine learning (Chapter 2) will be pursued with a new technique to detect extrapolation (Section 6.1). The techniques developed for the automated discovery of SARS-CoV-2 primers and tumor signatures (Chapter 5) can interestingly be generalized to detect any kind of genomic subsequence: a promising development is to use this methodology to uncover bacterial genes that provide resistance to antibiotics, helping experts quickly ascertain the type of antibiotic that could work against a given infection (Section 6.2). The food science applications I worked on (Chapter 4) lead naturally to future applications on the development of alternative proteins and finding compromises for ecosystem services (Section 6.3). My experience in evolutionary computation (Chapter 3) will be used throughout my future research lines.

6.1 Detecting Extrapolation in Machine Learning

The techniques to assess generalization capabilities of machine learning (ML) algorithms described in Section 2.1 rely upon a methodology to detect extrapolation, the computation of the Convex Hull (CH) of a dataset. To summarize, the CH is hyper-polygon with minimal hyper-volume that contains all data points in a given dataset. When a new point is inside the CH of a training set, it is possible to assume that a ML algorithm will interpolate in order to provide a prediction for it; when a new point is outside, the ML algorithm will probably be forced to extrapolate. While this technique is effective, and testing for the relative position of the point is computationally feasible even in high dimensions, there are cases where the CH could deliver misleading information. For example, the training data provided might not be equally distributed in the feature space: as a result, the data points will be denser in some parts of the CH, and extremely rare in others. In this situation, it might be a mistake to assume that a ML algorithm would be in interpolation inside the sparser areas of the CH: the information available to the algorithm for that part of the feature space could have been so scarce that the resulting model should be treated as extrapolating instead, see Figure 6.1 for an example.

In order to tackle this issue, together with Pietro Barbiero, University of Cambridge, UK, and

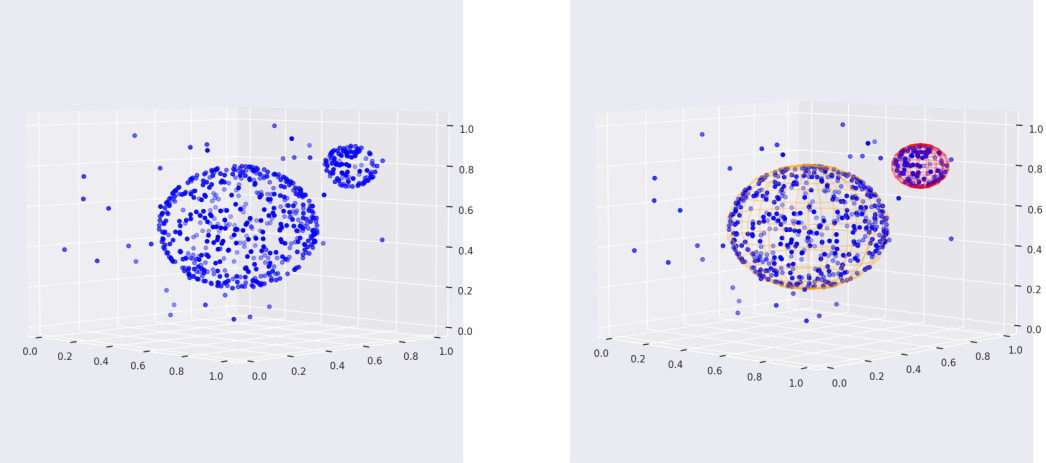


Figure 6.1: Example of a dataset **(left)** with relatively sparse data, except for two dense areas highlighted by spheres **(right)**. In this situation, even if the theoretical convex hull of the dataset is much larger, it is conceivable that a machine learning model trained on this data would interpolate only in the areas delimited by the spheres.

Ben Mathiesen, Capgemini Engineering, France, I am working on a novel technique to assess extrapolation for ML algorithms. The basic idea is to find and isolate areas of the feature space with a high density of data points, and surround them with simple geometric shapes, such as hyperspheres or hypercylinders, surrounded by empty (or extremely sparse) areas. This technique, for the moment termed *P-Sphere Hull*, has several desirable criteria for integration into industrial applications:

- **Modular.** It provides a geometric model of the training domain as a collection of subdomains bounded by simple geometric envelopes. The subdomains can be defined by any clustering method upstream, and the domain model can be consumed by any AI model downstream.
- **Computationally inexpensive.** The domain model should add almost no overhead to the training process. The P-Sphere Hull is generated once, during the exploration phase of a data science project. Testing a new point for membership in the hull requires only a single dot product operation per subdomain. This consideration is the main reason that I explore modeling the domain as a collection of spheres, cylinders, and boxes rather than simplexes or CHs.
- **Intuitive.** The domain model can provide information on the topology of the dataset in high-dimensional space. For example, it can reveal a series of overlapping subdomains with low intrinsic dimensionality and similar orientations extending away from the main body of the point cloud.
- **Compact.** The domain model hypervolume is small compared to the hypercuboid defined by individual feature ranges, or even when compared to the hypervolume of a CH. The P-Sphere Hull can calculate its own volume by geometry or by subsampling the space.
- **Insensitive to scatter.** I assume that industrial data are subject to both measurement errors and random sampling uncertainty. Hence, I view the training domain as a collection of point

clouds rather than a continuous manifold.

- **Permissive.** Since the data are assumed to be noisy, the boundaries of the domain should not contain too many data points.

While preliminary experiments seem promising, there are still several obstacles to be overcome for this technique to be usable out of the box. The main issue lies in the definition of the areas to be surrounded by hyperspheres and hypercylinders, that is currently performed by clustering. Clustering is an unsupervised ML procedure that attempts to find subsets of data points, without ground truth: consequently, its outcomes can vary considerably depending on the hyperparameters of the clustering algorithm used, which might return unsatisfying results for certain datasets. Finding good general hyperparameters for clustering is not straightforward, and we are currently focusing on this part in order to make the P-Sphere Hull viable.

6.2 Prediction of Antibiotic Sensitivity

The techniques developed in Section 5.1, originally conceived for uncovering primer sets in viral RNA, can in principle be generalized to finding any subsequence of genetic material that can separate different classes of organisms. A promising application for this methodology is the prediction of antimicrobial resistance (AMR) in bacteria. AMR is the acquired resistance of fungi, viruses, bacteria and parasites to medicine designed to combat them, and it is a growing threat, due to the misuse of antibiotics: in 2016, 30% of neonatal sepsis deaths were linked to AMR [Fol+17]. Indeed, while fast and adequate treatment is essential to the survival of patients, an increasing amount of so-called *superbugs* are resistant to both first-line and second-line antibiotics recommended by the WHO. In order to assess resistance or sensitivity of a specific bacteria to a given antibiotic, techniques known as antibiotic sensitivity testing (AST) have been developed. While effective, these methodologies are extremely time-consuming, as they require sending samples of the bacteria to laboratory, where disks of drugs are placed in a Petri dish, and an agar plate inoculated with the isolated pathogen is placed in the same dish. After a day or so of incubation, the dish is inspected to determine which drugs either killed or prevented the growth of the microbe: in which case the micro-organism is considered susceptible to the treatment, see Figure 6.2. Due to the incubation step, this process can take up to 24 hours.

A viable alternative would be trying to infer sensitivity just through the analysis of the bacterial genome. As techniques for rapid genomic sequencing are becoming faster and less expensive, it is now possible to obtain a digital version of the genome of a bacteria in a matter of hours. It is then possible to use the same methodology previously outlined in Section 5.1 to find the subsequences that separate bacteria resistant to specific antibiotics from those which are sensitive. I started exploring this line of research, together with my colleagues Pietro Barbiero and Giovanni Squillero, through the supervision of a master student, Arthur Cahu, École Polytechnique, France, and the co-supervision of a second master student, Simone Alessandrì, Politecnico di Torino, Italy.

While the preliminary results are promising, bacteria offer unique challenges, mainly due to the their considerable higher complexity, with respect to viruses previously analyzed. Not only the length of their genomic sequence is order of magnitudes higher (millions of base pairs against tens of thousands), but post-sequencing algorithms that attempt to reconstruct the genomic sequence in-order often struggle with bacteria, producing as a results several files, each one ranging from thousands to millions of bases, that could appear in any order in the bacterial DNA. Everything is further complicated by the presence of mobile genetic elements (MGEs), also known as selfish genetic elements, a type of genetic material that can move around within a genome, or that can be

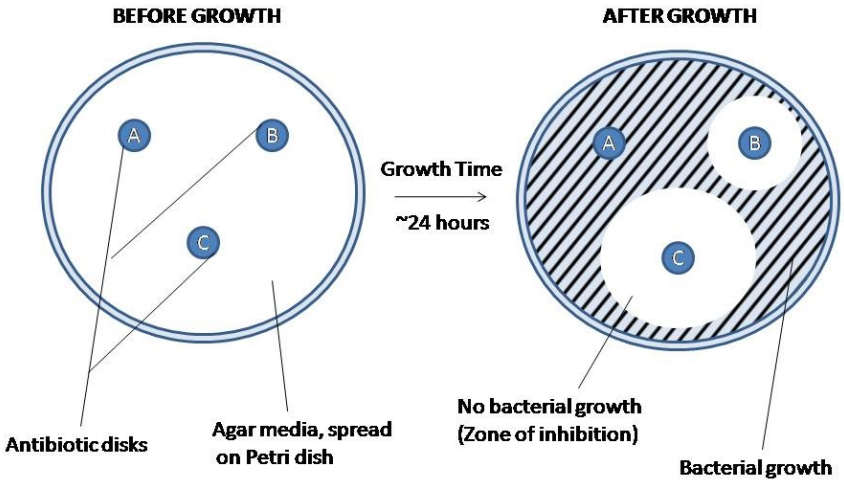


Figure 6.2: Diagram of a disk diffusion test. Antibiotic A does not inhibit bacterial growth, while C does so effectively; the microbe seems only partially susceptible to B. *Source*: Wikimedia Commons (Sommer36)

transferred from one species to another [Koo16]. As convolutional neural networks (CNNs) need to consider the adjacency of the elements, this could potentially be a sizable obstacle.

In order to overcome this hurdle, we are currently exploring alternative approaches: one solution could be using an EA, as described in subsection 5.1.3, but that introduces further research questions on the preservation of locality, or in other words, how to conceive mutations and cross-overs so that they make sense in the complex genomic space of a bacteria; another interesting option is to describe the bacterial genome as a De Bruijn graph [CPT11], and then use a graph neural network in a similar fashion as the CNN was adopted for SARS-CoV-2.

6.3 Multi-objective Optimization for Food Science

Multi-objective optimization [Deb05] is one of the domains where EAs really shine, and since effective multi-objective algorithms now have been available for nearly 20 years [Deb+02], it is not surprising that even domains as far away from computer science as life sciences started adopting these techniques to tackle problems with conflicting criteria. One interesting example is the field of life-cycle analysis/assessment (LCA), a methodology for evaluating environmental impacts associated with all the stages of the life cycle of a commercial product, process, or service, where expert of the domain have been long aware of the need for multi-objective optimization, to find trade-offs between environmental needs, production efficiency, and societal requirements [AC99].

Thanks to my experience with EA and multi-objective optimizaton in particular, I am currently participating in a European Horizon 2020 project on the development of sustainable insect production chains for food and feed, called SUSINCHAIN[1]. In the work package I am part of, dealing with modelling and optimization, I plan to use multi-objective optimization to find trade-offs between the different objectives collected by interviewing domain experts in the same project. This real-world task poses intriguing challenges: for example, so far a total of 18 different objectives have been gathered [Ton+21], a number that tests the limits of multi-objective optimization, and goes into

[1] `https://susinchain.eu/`

what EA practitioners call *many-objective* optimization [ITN08]. It is well known, in fact, that the performance of classical multi-objective EAs degrades as the number of objective increases. Together with a post-doctoral researcher that I recruited on this project, Nisrine Mouhrim, we are going to explore different possibilities to tackle this obstacle, ranging from objective aggregation to dimensionality reduction in the space of the objectives [DS+06], with the ultimate objective of providing reasonable trade-offs for the experts to analyze. I also plan to propose a similar approach in the scope of an Horizon Europe project proposal coordinated by Alessandra Bordoni, University of Bologna, Italy, that I am currently drafting as a work package leader, again focused on the production of alternative proteins for food and feed.

A second domain where multi-objective optimization has the potential of delivering significant results is the study of ecosystem services (ES). ES include all benefits to humans provided by the natural environment and from healthy ecosystems, ranging from crop production, to animal energy production, to carbon sequestration. Land allocation in ES is a naturally occurring multi-objective problem: intuitively, assigning a plot of land to crop production will increase the quantity of food yielded, but will in turn reduce the potential carbon sequestration that would have been obtained by assigning the same land to forests, or the animal energy production in the case of the allocation of the same plot to grasslands for animal husbandry. After preliminary works on multi-objective optimization applied to ES in cooperation with Francesco Accatino, INRAE, France [Acc+19a; Acc+19b] (see Figure 6.3), I am currently progressing our cooperation by addressing the assessment of uncertainty in ES models, a subject still relatively unexplored in the community. To this aim, I developed an open-source package named HumanModels[2], that will make it possible to perform robust, reproducible experiments with human-designed ES models, and compare them to ML models on the same data, in a transparent way.

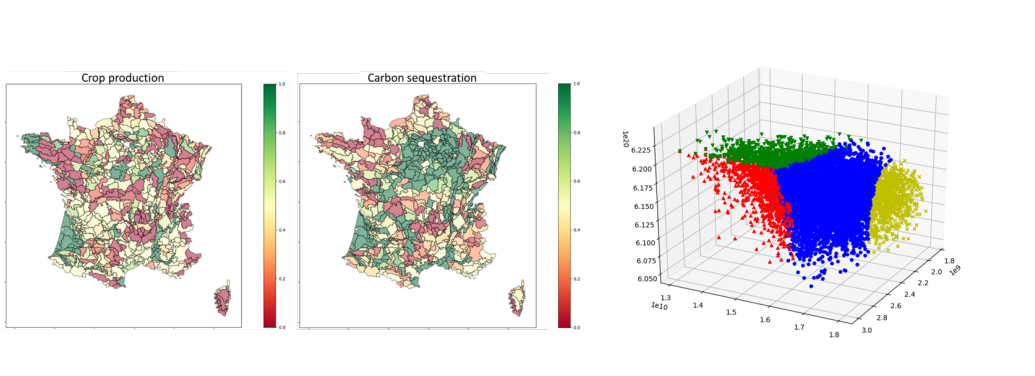
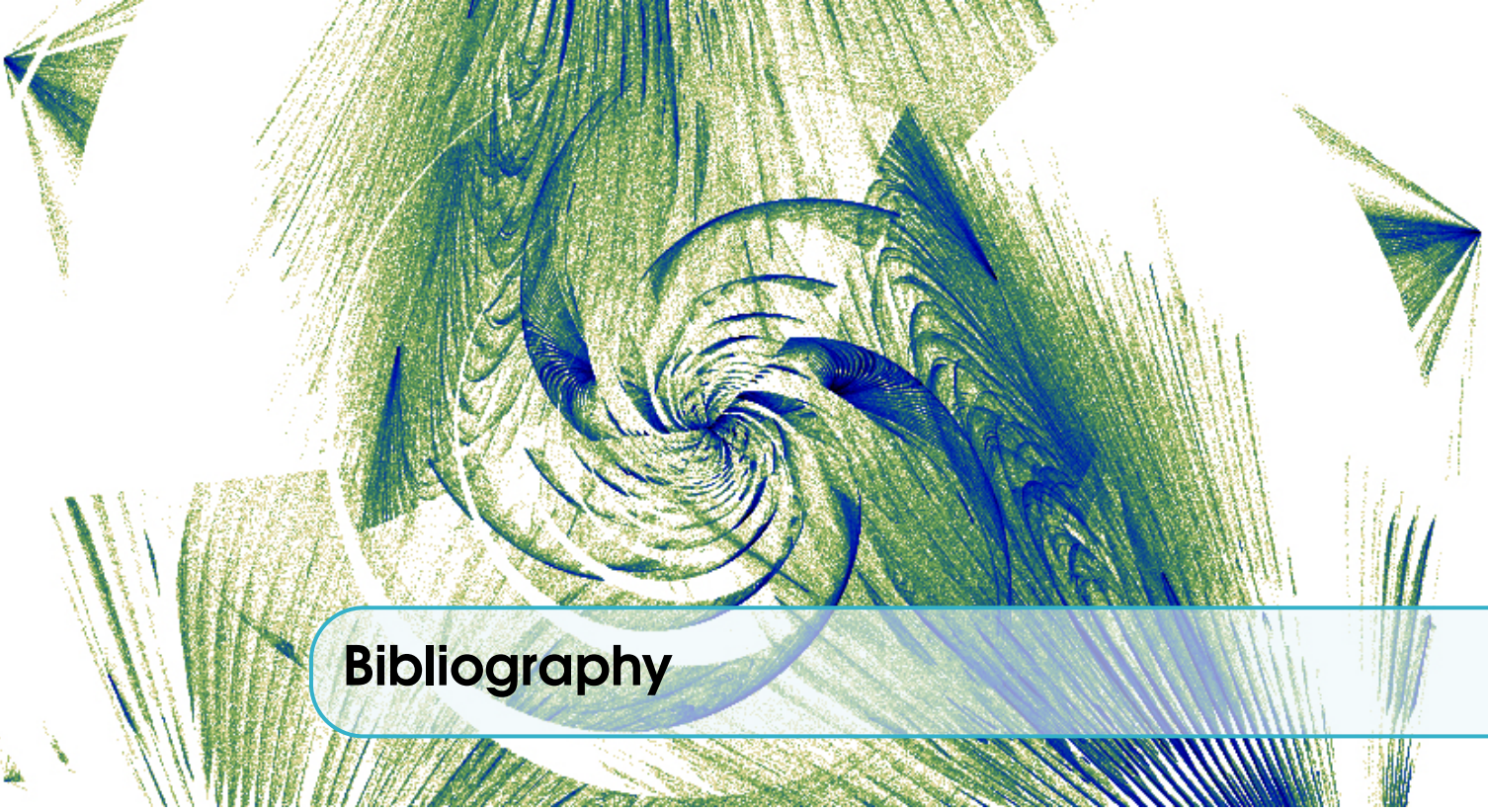


Figure 6.3: Preliminary results of multi-objective optimization applied to ecosystem services. The figures on the left and center show the impact on the French region of a specific candidate solution on the Pareto front for the conflicting objectives of crop production and carbon sequestration. The figure on the right is an example of a Pareto front found by multi-objective evolutionary optimization for three conflicting objectives: crop production, carbon sequestration, and animal energy production. Points in green represent solutions that have better fitness values of carbon sequestration, points in yellow feature better values for crop production, while points in red have better values for animal energy production.

[2] `https://pypi.org/project/humanmodels/`



Bibliography

[Acc+19a] Francesco Accatino et al. “The Adaptability of an Extensive Cattle Beef Farming System to Contrasted Societal Preferences - Coupling Multi-Objective Analysis and a Participatory Approach”. In: *European Association of Agricultural Economists (EAAE) Seminar*. 2019 (cited on page 247).

[Acc+19b] Francesco Accatino et al. “Trade-offs and Synergies Between Livestock Production and Other Ecosystem Services”. In: *Agricultural Systems* 168 (Jan. 2019), pages 58–72 (cited on page 247).

[AC99] A. Azapagic and R. Clift. “Life Cycle Assessment and Multiobjective Optimisation”. In: *Journal of Cleaner Production* 7.2 (Mar. 1999), pages 135–143 (cited on page 246).

[CPT11] Phillip E C Compeau, Pavel A Pevzner, and Glenn Tesler. “How to apply de Bruijn graphs to genome assembly”. In: *Nature Biotechnology* 29.11 (Nov. 2011), pages 987–991 (cited on page 246).

[Deb+02] K. Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *Evolutionary Computation, IEEE Transactions on* 6.2 (Apr. 2002), pages 182–197. ISSN: 1089-778X (cited on page 246).

[Deb05] Kalyanmoy Deb. “Multi-Objective Optimization”. In: *Search Methodologies*. Edited by Edmund K. Burke and Graham Kendall. Springer US, 2005, pages 273–316. ISBN: 978-0-387-28356-2 (cited on page 246).

[DS+06] Kalyanmoy Deb, D Saxena, et al. “Searching for Pareto-optimal Solutions Through Dimensionality Reduction for Certain Large-dimensional Multi-objective Optimization Problems”. In: *Proceedings of the world congress on computational intelligence (WCCI-2006)*. 2006, pages 3352–3360 (cited on page 247).

[Fol+17] Laura Folgori et al. “Tackling antimicrobial resistance in neonatal sepsis”. In: *The Lancet Global Health* 5.11 (Nov. 2017), e1066–e1068 (cited on page 245).

[ITN08] Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. “Evolutionary many-objective optimization: A short review”. In: *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, June 2008 (cited on page 247).

[Koo16] Eugene V. Koonin. “Viruses and mobile elements as drivers of evolutionary transitions”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 371.1701 (Aug. 2016), page 20150442 (cited on page 246).

[Ton+21] Alberto Tonda et al. “Multi-objective Optimization for Sustainable Insect Chains”. In: *EAAP Annual Meeting 2021*. 2021 (cited on page 246).