Probabilistic personalised cascade with abstention

Tatiana Shpakova, Nataliya Sokolovska

Pattern Recognition Letters 147 (2021) 8–15

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Probabilistic personalised cascade with abstention☆

Tatiana Shpakova, Nataliya Sokolovska*

NutriOmics, INSERM, Sorbonne University, Paris, France

ARTICLE INFO

Article history:
Received 28 August 2020
Revised 26 February 2021
Accepted 29 March 2021
Available online 10 April 2021

Keywords:
Graphical models
Learning under budget
Feature selection
Cascade classifier

ABSTRACT

Cascade learning with abstention and individualised feature selection is a class of models in high demand in personalised medical applications. The cascade consists of sequential *classifiers* and *rejectors*, where the classifiers estimate confidence of prediction, and the rejectors evaluate an expected cost-to-go of features not selected yet. The number of models is exponential in the number of features and, therefore, the challenge is to find efficient heuristics for the NP-hard problem.

The state-of-the-art is based on complex deep neural networks. We introduce an efficient and robust approach based on a probabilistic graphical model representing a unified probabilistic classifier that can be applied at any stage of a multi-stage sequential model. As for the rejector, we build it on the probabilistic-based neural network that incorporates the *very same* probabilistic model to treat unobserved feature values.

We illustrate the efficiency of the proposed method on several data sets, and compare our results to the state-of-the-art.

1. Introduction

Cascade classification with abstention is a scenario where an algorithm can abstain from providing a class label, and ask for more information. Usually one means a confidence-based abstention where the classifier abstains, if it is uncertain about its prediction, and the rejection comes with some cost. The process of gradual feature acquisition is governed by two functions: *classifier* and *rejector/selector* (which performs also feature selection, we refer to it as rejector in the following), and is drafted on Fig. 1. At each stage, the classifier makes prediction $y \in \{1, \ldots, K\}$ where K is the number of classes, and simultaneously estimates confidence of this prediction, and the rejector evaluates an expected cost-to-go for every feature unobserved at this stage.

A seminal work of [1] proposed theoretical foundations for learning with rejection. An important aspect of multi-stage sequential classifiers is to learn whether to reject a decision to classify, or to label a current observation. Some recent publications proposed solutions to this problem discussing possible scenarios to estimate regions where a classifier is confident in its decision, i.e., they considered how to learn a rejector efficiently [2–4]. An interesting research direction is to explore feature redundancies to reduce the cost of learning and prediction [5], and to explore structure in cost sensitive learning, using, e.g., Bayesian networks [6].

We are motivated by the challenges coming from personalised medicine where it is crucial to find optimal — in terms of money, time, other budget constrains, and predictive accuracy — individual diagnostic protocols. The goal of this work is to learn simultaneously the classifier and the rejector, and to perform *personalised*, i.e., individual for each observation, feature selection. As the state-of-the-art methods that are particularly relevant to our work, we can mention the cascade approaches based on the deep-Q-network [7]. They were introduced by Clertant et al. [8] and further developed by Janisch et al. [9].

We challenge the following crucial problems. First, how to learn a *unified classification model*, and second, how to avoid learning an exponential number of models in case of individualised feature selection. In [8,9], an attempt to approximate such a unique model was done within a neural network (NN) paradigm.

In the cascading model, all feature values can be available on the last stage only. On all previous stages, the unobserved feature values can be considered as *missing data*, and treated using methods for missing values processing. The NN-based approaches are sensible to missing values. Both in [8,9] binary masks are used as an extra input to the neural network to replace them. This classical trick was discussed and criticised by Yi et al. [10] where it was reported that the zero imputation is far from being the best strategy

☆ Editor: "Jiwen Lu".

* Corresponding author.
E-mail address: nataliya.sokolovska@sorbonne-universite.fr (N. Sokolovska).

https://doi.org/10.1016/j.patrec.2021.03.029
0167-8655/

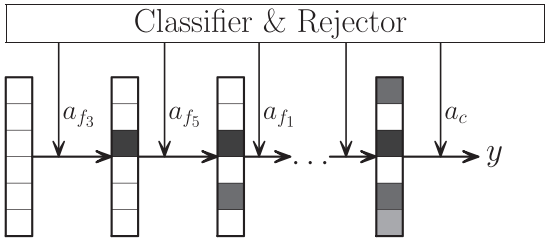

Fig. 1. Sequential process of classification. Actions a_{f_i} are features selected by the rejector/selector, action a_c is the action to classify, and $y \in \{1, \dots, K\}$ is the label predicted by the classifier. The dark features are ones selected by the cascade at different stages.

due to biasness of the estimation. To deal with this issue, we have got inspired from the work of [11] and build in the generalised neuron's activation in our cascade model.

To sum up, in this work we aim to make use of a probabilistic classifier, and our motivation to use a probabilistic approach is two-fold. First, a probabilistic graphical model (*classifier*) provides us with some machinery to treat missing values naturally. Second, a way to treat missing values in a neural network (*rejector*) is to apply a generalised neuron's activation introduced by Smieja et al. [11] which assumes that data belong to a Gaussian mixture distribution. Note, that the rejector computes cost-to-go values for all features, what can be efficiently done by a reinforcement learning algorithm finding an optimal policy, and we do not aim to replace the rejector by a probabilistic model, but to make the rejector be able to take the Gaussian parameters of the probabilistic classifier into account. In this work, we consider Gaussian graphical models, since one of their main advantages is that the conditional independence is modelled by the covariance matrix only.

Our contribution is multi-fold:

- We propose a *unified cascade classifier with abstention* that is more efficient than the state-of-the-art models [8,9], and also than multi-stage models with an exponential number (in features) of classifiers and rejectors needed at each stage. A *generative graphical model plays the role of the classifier, and shares its parameters with the rejector* (a neural network) leading to a computationally attractive framework for high-dimensional problems.
- We derive a *novel learning algorithm* for the proposed *cascade classifier*, and we show that the learning time for our approach is up to *5 times faster* than for the state-of-the-art methods.
- The proposed probabilistic cascade classifier *naturally treats missing values*, it relies on methods that integrate them out, and it can have a *semi-supervised flavour* and incorporate some *prior knowledge*.
- Finally, we illustrate by our experiments on benchmarks and real data that the proposed approach is efficient, achieves the state-of-the-art performance, increases the *interpretability* of the classifier, since it carries probabilistic nature and sense, and has an elegant form.

The paper is organised as follows. We discuss the related state-of-the-art methods in Section 2. The problem we challenge in this work is formalised in Section 3. The ideas how to integrate prior knowledge are considered in Section 4. We introduce our approach and discuss the learning procedure in Section 5. Section 6 is dedicated to the results of our numerical experiments. Concluding remarks and perspectives close the paper. To facilitate the reading, we decided to put our results on structure learning, as well as supplementary results on data imputation methods in the Supplementary Material.

2. Related work

There are several recent approaches to learn models with abstention [3,12–14], and they are motivated by different practical needs. A number of methods are focused on learning the reject regions, assuming the classifiers are already trained. One work to mention is of [2] that proposes a multi-stage sequential classifier which is formulated as a Markov Decision Process (MDP), to learn regions where the classifier is uncertain and prefers to abstain from making a decision.

In this work, we build our approach on the method of learning cascade classifiers with abstention which is introduced in [8]. It is a Partially Observable Markov Decision Process (POMDP)-based framework to estimate cost-sensitive heterogeneous multi-stage classifiers. The cascade classifier of [8] consists of two neural networks, a classifier neural network which returns a probability of belonging to each class, and a rejector/selector neural network which returns a cost-to-go for each feature, and that helps to decide whether to return a label at the current stage of the cascade, or to pay for a new feature acquisition. The problem of learning dynamic diagnostic protocols where an optimal order of features for each individual is essential, is closely related to feature selection.

As we show below, in an individualised cascade learning, a classifier faces a problem of missing values processing, since not all features are acquired and explored on all its stages. To deal with the missingness, a number of approaches have been proposed. The most commonly used approach is data imputation, which is a class of methods to complete complex data containing missing entries. Among the discriminative methods, the most used are probably MICE [15], matrix completion [16], and MissForest [17]. The generative methods are based either on the Expectation-Maximisation algorithm [18], or on deep learning [19]. GAIN [19] is an approach to impute data using generative adversarial networks. Due to the space limitations, we do not focus on the imputation approaches. However, we tested some of the state-of-the-art data imputation methods, and these results can be found in the appendix.

Processing of missing values by neural networks.

In the neural networks setup, features with missing values serve as an input layer of the rejector and the classifier. In [8,9], the problem of missing values was treated by substituting them by zeros and adding binary masks for features available and not available. This pre-processing is rather ad-hoc and can be avoided. The authors of [11] propose a natural approach to deal with missing data for several types of neural networks (e.g., neural networks where the first layer has a ReLU activation function).

In brief, the idea of [11] is to replace the standard neuron's response in the first hidden layer by its expected value. So, the output of the first layer is an expected value of the ReLU function given a probabilistic model. During the learning procedure, the Gaussian mixture model's (GMM) weights are updated altogether with the weights of the neural network. For more details take a look at [11].

As our classifier relies on a Gaussian probabilistic model, we will take an advantage of the similar nature of these two models and use the parameters of the Gaussian models both for the rejector and the classifier. So, we can alternate between the updates of the classifier and the rejector during the parameter learning procedure.

3. Problem formulation: a POMDP framework for cascade learning

We challenge to learn cascade classifiers, and we solve this problem from graphical models perspective using deep Q-

learning [7]. We follow mainly the notations also used by Clertant et al. [8].

We consider a dataset $\mathcal{D} = (x_n, y_n)_{n=1,\dots,N}$ which consists of data points $x_n \in \mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_D$ and the corresponding labels $y_n \in \{1, \dots, K\}$. Here we assume that all feature values are available for each observation. On each step t of the cascade classifier, we choose an action a_t that can be either to acquire a new previously unseen feature from $\{1, \dots, D\}$, or to stop and make a prediction, i.e., assign a class from $\{1, \dots, K\}$. Note that at a stage t, the model has access to all features $x_{[t]}$ collected prior to t.

A state of the model can be expressed as $s_t = (a_{[t]}, x_{[t]})$, and we call it *historic* state, since it contains all previous observations $x_{[t]}$ and previous decisions $a_{[t]}$. A decision $a_t \in A = \{-D, \dots, -1, 1, \dots, K\}$, whether to extract a new feature, or to classify and stop, can be taken according to the following rule:

$$a_t^* = \arg\min_{a_t \in A} \left[\mathbb{I}_{[a_t > 0]} F(a_t|s_{t-1}) + \mathbb{I}_{[a_t < 0]} R(a_t|s_{t-1}) \right] \tag{1}$$

$$= \arg\min_{a_t \in A} Q^\pi(a_t|s_{t-1}) = \pi(s_{t-1}), \tag{2}$$

where $F(a_t|s_{t-1})$ is called *classifier* and evaluates the classification mistake of the prediction $\hat{y} = a_t$, and $R(a_t|s_{t-1})$ is called *rejector* and computes the cost-to-go for each possible feature $-a_t$ not selected yet. Overall, our policy π and the whole trajectory of actions is defined by:

$$Q^\pi(a_t|s_{t-1}) = \begin{cases} 1 - \mathbb{P}(y = a_t|s_{t-1}), & \text{for } a_t > 0 \\ \delta_{-a_t} + V^\pi(s_t), & \text{for } a_t < 0, \end{cases} \tag{3}$$

where δ_{-a_t} is the cost of feature $-a_t$, and $V^\pi(s)$ is an expected cost from state s under policy π. In [8,9], the function $Q^\pi(a_t|s_{t-1})$ is approximated by two deep neural networks, one for selector/rejector $R(a_t|s_{t-1})$ and one for classifier $F(a_t|s_{t-1})$.

We are intended, in particular, to learn another function $F(a_t|s_{t-1})$ which applies at any t of the cascade for any observation. In addition, we are going to modify the rejector $R(a_t|s_{t-1})$ *to relate it to the classifier through the same graphical model*. Below, we focus on how to learn this unique classifier and the corresponding rejector using probabilistic graphical models.

4. A semi-supervised generalised neural response

In a number of applications, in particular in such domains as medicine and biology, scientists have access to cheap prior knowledge, e.g., KEGG (Kyoto Encyclopedia of Genes and Genomes), GO (Gene Ontology). The problem of prior knowledge integration which is closely related to semi-supervised learning, is known to be challenging, and a big number of research papers report negative results [20]. We focus on probabilistic classifiers where the prior knowledge can be integrated in the form of the marginal probability of observations [21].

We propose a *semi-supervised extension of the generalised neural response* [11], and our approach is expected to be more efficient in terms of empirical predictive performance, in particular for cases where the number of labeled observations is small. Note, the implementation of our method is simple.

Following the notations of [11], an observation with missing data is denoted by (x, m), $x \in \mathbb{R}^D$, and $m \subset \{1, \dots, D\}$ are features ids with missing values. We are in a semi-supervised setting, and we assume that we have access to huge (possibly infinite) number of observations that allows us to approximate (or to compute the true values) of the D-dimensional probability distribution q. So, the missing data at random in a given data set are generated by this (unknown) distribution q.

Comparing the proposed semi-supervised approach and Eq. (1) of [11], the density conditioned on the observed features is replaced by $q(x)$ which is either approximated from an external big data set, or provided as a ground truth by human experts:

$$\mathcal{F}(x) = \begin{cases} q(x), & \text{if } x \text{ is missing,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Then, the result of the generalised neuron response activation holds also for the semi-supervised setting. Suppose that $x \sim \mathcal{N}(\mu, \Sigma)$, and

$$w^T x + b \sim \mathcal{N}(w^T x + b, w^T \Sigma w). \tag{5}$$

If x comes from a Gaussian mixture $x \sim \sum_{i=1}^M p_i \mathcal{N}(\mu_i, \Sigma_i)$ with M components, then

$$w^T x + b \sim \sum_{i=1}^M p_i \mathcal{N}(w^T \mu_i + b, w^T \Sigma_i w). \tag{6}$$

Theorem 1. *Given w and b, where $w \in \mathbb{R}^D$ and b are parameters of a neuron, and where $\mathcal{F}(x)$ for missing data is completed using Eq. (4), then*

$$\mathrm{ReLU}(\mathcal{F})_{w,b} = \sum_{i=1}^M p_i \mathrm{NR}\left(\frac{w^T \mu_i + b}{\sqrt{w^T \Sigma_i w}} \right), \tag{7}$$

where NR (neuron's response) $= \mathrm{ReLU}(\mathcal{N}(w, 1))$.

$$NR(w) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{w^2}{2} + \frac{w}{2}(1 + \mathrm{erf}(\frac{w}{\sqrt{(2}})) \right),$$

where $w = \frac{\mu}{\sigma}$ and can be calculated for the each component and each dimension separately, and $\mathrm{erf}(z) = \frac{2}{\pi} \int_0^z \exp(-a^2) da$. See [11] for more details.

5. Learning a unified cascade classifier with abstention

At each stage of the heterogeneous cascade, we have values missing at random (MAR). *This situation arises, since at various stages of the cascade learning, different features are decided to be explored for different observations.* So, a classifier gets a data matrix $X = (X^{obs}, X^{mis})$ which contains a lot of missing values in the beginning of the cascade learning, and starts to be full when all features are explored.

5.1. Gaussian graphical models

Classifier. We can build our classifier function $F(a_t|s_{t-1})$ based on a probabilistic model in the following way:

$$F(a_t|s_{t-1}) = 1 - \mathbb{P}(y = a_t|x^{obs}),$$

where x^{obs} is the part of the signal that we observe, y is the predicted label, and $F(a_t|s_{t-1})$ represents the probability of misclassification.

It is worth noting, that we have to evaluate probability $\mathbb{P}(y = a_t|x^{obs})$ at any step of the cascade, i.e., for signals x^{obs} with different number of observed elements. This fact conceals the main challenge of the learning task: our model should be able to provide a probability value for any subvector or any subset of features. Our idea is to introduce a Gaussian probabilistic model $\mathbb{P}(y = a_t|x)$ for fully-observed signal x, and, second, use the tools of the graphical models theory to derive the expressions for all possible $\mathbb{P}(y = a_t|x^{obs})$. A Gaussian graphical model $P(x|y = k, \theta)$ is fully defined by the parameter $\theta = (\mu_k, \Sigma_k)_{k=1}^K$, where K is the number of classes and θ is the parameter to be estimated. The challenge is to use the probability values as training instances. For the classifier, it can be done via weighted likelihood estimation considered in Section 5.4.

We can not fit a generative Gaussian classifier using all features in advance, since the feature selection in the cascade classifier is

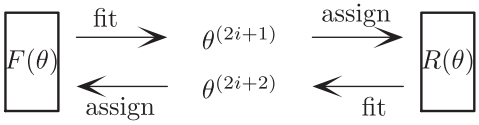

Fig. 2. An alternative view of the learning procedure sketched as Algorithm 1.

cost-dependent. However, we can apply a warm start based on some prior knowledge available, as we discussed in Section 4.

Rejector Our rejector is a neural network similar to the state-of-the-art approaches. One of the reasons why we do not apply a Gaussian model for the rejector, is that it is not obvious how to design a rejector via probabilistic graphical models (PGM). Attempting to do it, we can try to express $R(a_t|s_{t-1})$ through $\mathbb{P}(x_i|x^{obs})$ for each i. A conditional probability $\mathbb{P}(x_i|x^{obs})$ can be re-written through the known values of probabilistic graphical model, however, it is impossible to compute the cost-to-go values for the features. So, we apply a neural network – the deep Q-learning to compute the cost-to-go for all features – with the PGM-based generalized neuron's response.

5.2. Sharing parameters between classifier and rejector

The classifier in our architecture are Gaussian graphical models with parameters $\theta = (\mu_k, \Sigma_k)_{k=1}^K$, one model per class k. The rejector is a neural network with a generalised neuron's response which can be either identical to one proposed by Smieja et al. [11], or a semi-supervised neuron's activation described in Section 4. In both cases, the Gaussian mixtures with parameters $\theta' = (\mu'_k, \Sigma'_k)_{k=1}^K$ are used in the rejector for the neuron's activation response.

In our classifier and our rejector, the input of the models coincide (it is the observed features of each data point). Therefore, we can consider the following model:

$$\theta = (\mu_k, \Sigma_k)_{k=1}^K = (\mu'_k, \Sigma'_k)_{k=1}^K = \theta', \tag{8}$$

where the rejector and the classifier share and update the same set of parameters. In this way, we *accelerate the learning procedure and profit from the double source of information* for the same shared parameters θ. Note that the rejector being a neural network, has much more parameters than θ', we also estimate matrices of weights and intercepts for each layer of the deep network (we use the ReLU and the softmax layers).

5.3. Learning procedure

The learning procedure we propose is drafted as Algorithm 1. It relies on the deep Q-learning procedure [7]. We perform E episodes (one episode considers one randomly sampled observation) and T iterations, where each iteration corresponds to a new feature acquisition, or to the decision to classify the current observation. To update the parameters θ, on each episode e, we sample a minibatch of fixed size J, and either perform one step of the gradient descent for the NN-based rejector, or update the parameters θ of the graphical model depending on the current number (odd or even) of the episode. The sketch of this iterative process is presented on Fig. 2.

In the following section, we describe how we update the parameters of the model using the weighted log-likelihood.

5.4. Weighted log-likelihood

The cascade learning procedure drafted in Algorithm 1 has to deal with non categorical class values y. It comes from the fact that the classifier F, approximated by a neural network or by a probabilistic model, returns conditional probabilities of a class given an observation. The parameters θ_k for each class ($\theta_k = [\mu_k, \Sigma_k]$) are updated using $\mathbb{P}(y = k|x, \theta_k)$, and not hard labels.

We interpret this probability as a degree of sureness. To explore this knowledge, we can use a weighted likelihood approach [22], where the dataset now consists of $\{x_n, \nu_n\}$, where $\nu_n \in [0, 1]$ represents the soft label. In this case, the objective has the following form for each θ_k:

$$\max_{\theta_k} \sum_{n=1}^{N} \nu_n \log \mathbb{P}(x_n|y = k, \theta_k). \tag{9}$$

Note that in a hard labels case where $\nu_n \in \{0, 1\}$, the expression (9) reproduces the standard maximum likelihood objective.

Algorithm 1 Learning probabilistic personalised cascade with abstention

for $e = 1, \ldots, E$ **do**
  Initialise s_1
  for $t = 1, \ldots, T$ **do**
    Take an action: $a_t^* = \arg\min_{a_t \in A} Q^\pi(a_t|s_{t-1}, \theta^{(e)})$
    Update and store in memory the historic state:
    $$s_t = (a_{[t]}, x_{[t]})$$
  end for
  Update labels for minibatch data (a_j, s_j, y_j) sampled from memory:
  $$y_j = \delta_{a_j} + \min_{a \in A} Q^\pi(a|s_{j-1}, \theta^{(e)})$$
  Sample a minibatch $(a_j, s_j, y_j)_{\{1,\ldots,J\}}$ from memory
  if e is odd **then**
    Update parameter $\theta^{(e)}$ of the classifier $F(\theta)$ using the weighted log-likelihood (eq. 9)
  else
    Update parameter $\theta^{(e)}$ of the rejector $R(\theta)$ using the minibatch data and the gradient descent
  end if
end for

6. Experiments

In this section, we illustrate the performance of the proposed approach on simulated and real-world benchmark data. We compare our approach — the POMDP framework with the unified probabilistic classifier that shares its parameters with the rejector — with the state-of-the-art approaches of [8,9] (https://github.com/jaromiru/cwcf), and also several baselines. Our primary motivation are small sample size problems where generative learning approaches are more promising compared to discriminative approaches.

Overall, we will present the results for the following competitive methods:

1. The state-of-the-art cascade classifier of [8] based on two neural networks;
2. A baseline model where the classifier is a Gaussian graphical model and the covariance matrix is diagonal, i.e., we assume all features are independent; rejector is a neural network taken from Clertant et al. [8];
3. Another baseline model where the classifier is a Gaussian graphical model and the covariance matrix is not diagonal; the rejector is neural network taken from Clertant et al. [8];

4. The rejector is the neural network with the generalised neuron activation from Smieja et al. [11]; classifier is neural network taken from Clertant et al. [8];
5. Our probabilistic cascade classifier described in Section 5, where the rejector is the neural network with the generalised neuron activation from Smieja et al. [11] and the classifier is a Gaussian graphical model;
6. Another recently published state-of-the-art method of [9]; similarly to the approach of [8], it is also based on the deep Q-learning.

On the figures below, the tested methods are presented in the same order.

We introduce the shared cost $\delta = 1/(d \cdot D)$ for all features similarly to [8]. In the cost expression, D is the number of features in a data set, and d is a hyper-parameter that we vary to simulate scenarios with various costs. We considered d in range $[1, 2, \ldots, 9]$. The more expensive a feature is, less accurate result is expected, since smaller costs encourage feature exploration. On all our figures illustrating performance, on the horizontal axis we show the average number of selected features by the corresponding models, and on the vertical axis we show test accuracy. We also show the standard deviation divided by 10 to facilitate the plots reading.

All the hyper-parameters were fixed by 10-fold cross validations. We used the code of [8], and we used the same hyper-parameters as in his implementation. We also performed the grid search among the parameters for the Q-learning and neural networks: the number of iterations (episodes) from 1000 to 5000 with the step 1000, exploration rate decay in range [0.9, 0.995, 0.999], learning rate in range [0.1, 0.01, 0.001, 0.0001].

It is an interesting point that curves have different number of points. Similarly to [9], in the cost-accuracy plane, we use the test set to select the most accurate model instances, which form a convex hull over all tested models. Also, all considered methods performing feature exploration, stop the feature acquisition earlier or later, and do not continue (on the test set) until no features are left.

6.1. Simulated data

For the simulated setup, we sampled two different 2-dimensional binary classification problems. One is a general classification problem (we used `make_classification` function of `sklearn`), and isotropic Gaussian blobs (`make_blobs` function) with default parameters. In both scenarios we generate data for a classification problem with two classes. Our motivation was to consider a data set, which is not separable if one feature (dimension) is selected, but can be easily separated in a two-dimensional space. We also add three extra white noise features to make it challenging for our cascade models to select proper features. To be precise, we considered a 5-dimensional data set, where 2 features are drawn using the scikit-learn package, `make_classification` function, and the other 3 features are white noise, and are sampled from the standard Gaussian distribution.

The performance on the simulated data sets – accuracy as a function of the number of selected features – is shown on Figs. 3 and 4. It is easy to see that the proposed POMDP with the generalised neuron activation (rej:GNR on the figures) and the introduced probabilistic cascade classifier (classif:indNB+rej:GNR) are highly competitive compared to the state-of-the-art methods.

6.2. Real-world benchmarks

The benchmark data we considered can be downloaded from the UCI Machine Learning Repository. They are binary prediction tasks:

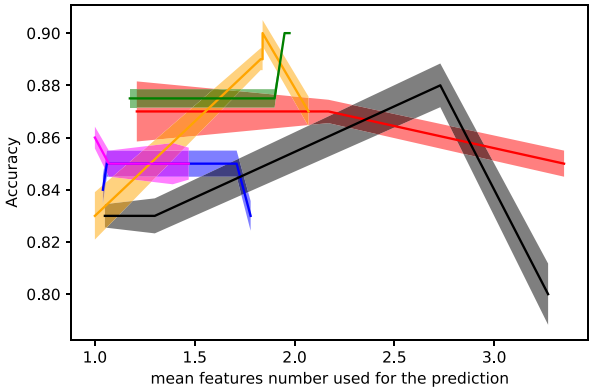

Fig. 3. Simulated dataset 1.

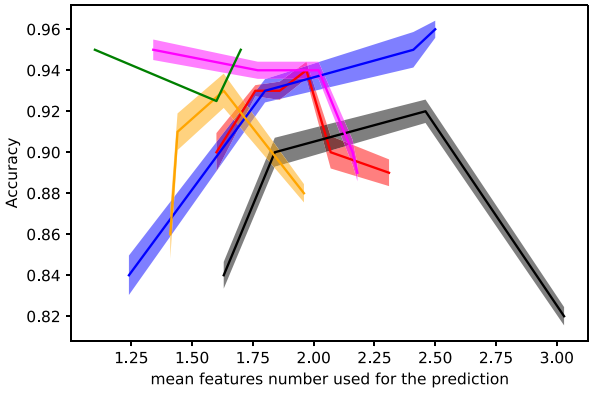

Fig. 4. Simulated dataset 2.

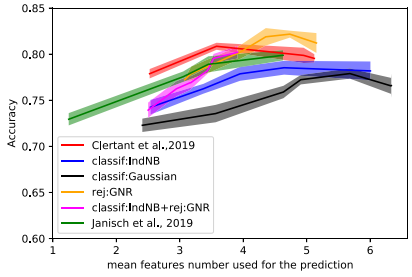

Fig. 5. Heart data, 90% for train.

- Heart Disease data set contains information about 303 patients and 13 features.
- Mammographic mass data set is dedicated to discrimination of benign and malignant mammographic masses. There are 6 attributes and 961 patients in the original data. We transformed the categorical variables into binary via the one-hot-encoding, and we obtained a data set with 14 variables.
- Breast Cancer Wisconsin (Prognostic). The data set contains 30 variables describing characteristics of the cell nuclei for 569 patients.

Figs. 5–7 illustrate our results on the Heart Disease data set, Figs. 8–10 – on the Mammography data, and Figs. 11–13 on the Breast Cancer data set. Figs. 5, 8, 11 represent the scenario where we do the 10-fold cross validation, and we use 90% data for training and 10% for test. We observe that the state-of-the-art methods of [8,9], as well as our approach where the rejector uses the generalised neuron activation, achieve the state-of-the-art performance. It is interesting that the behavior of the estimators depends heavily on the number of observations provided for learning, what

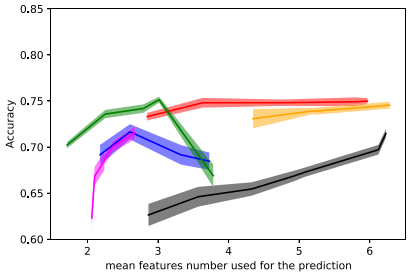

Fig. 6. Heart data; 10% for train (semi-supervised scenario).

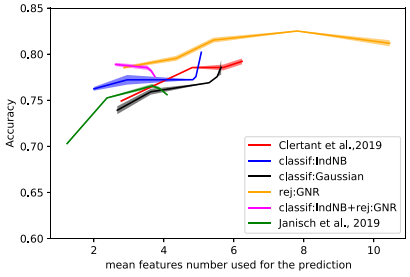

Fig. 7. Heart data; 50% for train (semi-supervised scenario).

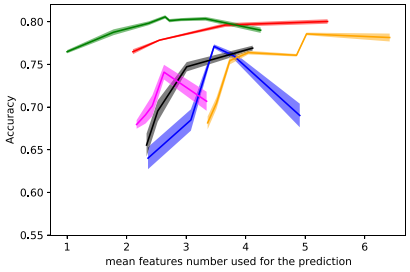

Fig. 8. Mammo; 90% for train.

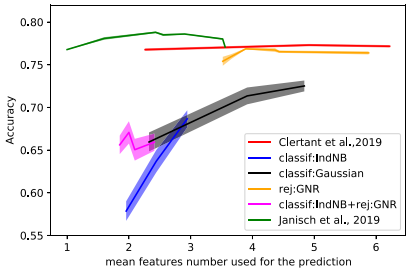

Fig. 9. Mammo; 10% for train (semi-supervised scenario).

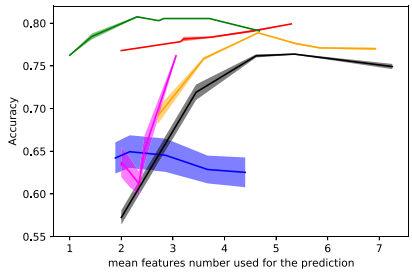

Fig. 10. Mammo; 50% for train (semi-supervised scenario).

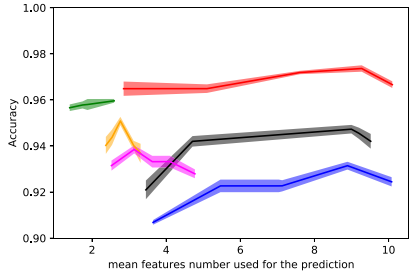

Fig. 11. Breast; 90% for train.

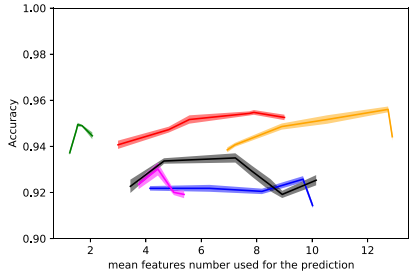

Fig. 12. Breast; 10% for train (semi-supervised scenario).

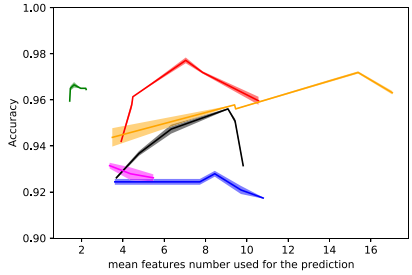

Fig. 13. Breast; 50% for train (semi-supervised scenario).

is quite easy to explain: the more observations are provided to a learning algorithm, the higher the predictive performance. For the semi-supervised setting (described in Section 4), we initialise the Gaussian parameters using all data, and take less data for training (we try 10% and 50% of data available). The proposed probabilistic scenario achieves quite reasonable results in cases where the number of observations is not high (50%), and the generalised neuron activation seems to always achieve the state-of-the-art precision.

Another important point to mention is that some approaches stop and predict labels earlier than the others. It is related to the cost value, and to the trade-off between the cost and the classifiers confidence.

Figs. 14–16 show the training time as a function of cost for the considered benchmarks. It is easy to see that the proposed probabilistic cascade classifier is the most efficient in terms of learning time. The simple Gaussian model where we consider independent features, is quite close to it in computational efficiency but note that the performance of it is not optimal.

We can observe that the proposed methods are comparably accurate, and sometimes they outperform the state-of-the-art cascade with abstention [8]. We also observe that our approach achieves a very reasonable performance for the data that match the Gaussian distribution assumption, and can meet some difficulties for some real data sets where this assumption does not hold.

The question of the computational time is one of main motivations for our method. Our method is computationally efficient, compared to other state-of-the-art methods, including [9]. The results (Figs. 14–16) illustrate that our method achieves the best learning time compared to all other considered approaches. We would like to precise that the neural network of [8] is rather simple (2 hidden layers, 6 neurons in each), however, the network of [9] is more complex (64 neurons in each layer, and 3 hidden lay-

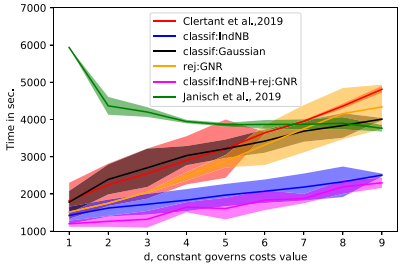

Fig. 14. Learning time Heart data set.

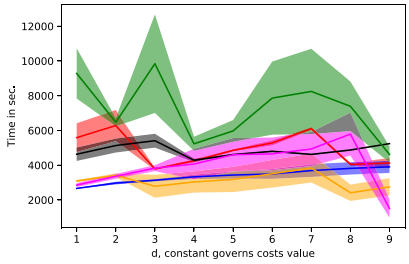

Fig. 15. Learning time Breast data set.

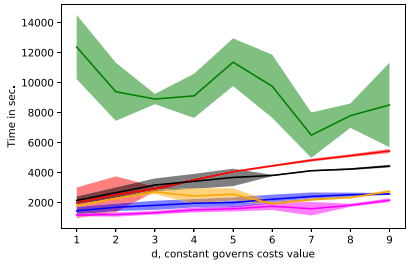

Fig. 16. Learning time Mammo data set.

ers), resulting in longer learning time. We test the smallest (simplest) neural network used by Janisch et al. [9], and their learning time is still always bigger than for our method. The approach of [9] implements an early stopping, where an error rate on a validation set is used to decide when to stop. This procedure (testing on the validation test), takes additional time during the training procedure.

The method of [9] outperforms all other methods on the Mammography data set (although it is still longer to train compared to other approaches). Our intuition why [9] is so accurate on the Mammography, is that it relies on quite a big neural network. The data set is also big (compared to other tested benchmarks), and the number of predictive features in the data set is relatively small: the method stops the feature exploration quite early. This configuration is beneficial for their method.

To run the computations, we used the Google Colab online server which is well-known by practitioners. For each launch it allocates 13GB RAM and 50 GB of the disk. Our implementation in Python, as well as our code for all the experiments reported here will be made publicly available.

7. Conclusion

Our goal was to develop and test a novel heterogeneous cascade classifier with a unified classification model. We propose a cascade classifier with abstention where the classifier and the rejector communicate and share parameters.

One of important constraints in real applications is computational efficiency what explains our motivation to estimate only one model which applies at any stage and for any observation of the cascade classifier. The need for efficiency has also motivated us to focus on multivariate Gaussian models where the unobserved, missing values and prior knowledge can be treated naturally.

The proposed method is much faster than the state-of-the-art cascade classifiers. In general, we observed that the proposed method is always faster, however, it can (potentially) perform worse that other state-of-the-art methods in some setups, since our method can be viewed as a simplified version of the NN-based methods.

Our method is based on the simple (Gaussian) graphical models, using the independence assumption between the features, and it learns fast (but more elaborate methods can perform better). The method of [9] achieves the state-of-the-art performance, however, it takes much longer to train, since the method relies on a rather big neural network, and it makes use of the early stopping, testing an error rate on a validation set.

We also considered the case of prior knowledge integration, and we propose an extension that belongs to the family of semi-supervised learning approaches.

Currently we investigate other generative graphical models as candidates for cascades with abstention with the goal to extend the area of applicability of our method, despite the fact that it can increase its computational complexity. Another interesting research avenue is introduction of discriminative graphical models into the cascades, however, this is a challenging task, since the problem of missing data imputation and the integration over unobserved data need to be solved and, moreover, solved efficiently. Another open issue is the introduction and detailed theoretical analysis of novel data imputation methods. We provide some ideas on it in the appendix.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Acknowledgments

This work was supported by the French National Research Agency (ANR JCJC DiagnoLearn).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patrec.2021.03.029.

References

[1] P.L. Bartlett, M.H. Wegkamp, Classification with a reject option using a hinge loss, J. Mach. Learn. Res. 9 (Aug) (2008) 1823–1840.
[2] K. Trapeznikov, V. Saligrama, Supervised sequential classification under budget constraints, in: Proceedings of the AISTATS, 2013.
[3] H. Ramaswamy, A. Tewari, S. Agarwal, Consistent algorithms for multiclass classification with a reject option, arXiv preprint arXiv:1505.04137(2015).
[4] F. Nan, V. Saligrama, Adaptive classifiers for prediction under a budget, in: Proceedings of the NIPS, 2017.
[5] M. Cebe, C. Gunduz-Demir, Qualitative test-cost sensitive classification, Pattern Recognit. Lett. 31 (13) (2010) 2043–2051.
[6] L. Jiang, C. Li, S. Wang, Cost-sensitive Bayesian network classifiers, Pattern Recognit. Lett. 45 (2014) 211–216.
[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. Bellemare, A. Graves, M. Riedmiller, A. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529.
[8] M. Clertant, N. Sokolovska, Y. Chevaleyre, B. Hanczar, Interpretable cascade classifiers with abstention, in: Proceedings of the AISTATS, 2019.

[9] J. Janisch, T. Pevný, V. Lisý, Classification with costly features using deep reinforcement learning., in: Proceedings of the AAAI, 2019.
[10] J. Yi, J. Lee, K.J. Kim, S.J. Hwang, E. Yang, Why not to use zero imputation? correcting sparsity bias in training neural networks, in: Proceedings of the ICLR, 2020.
[11] M. Smieja, L. Struski, J. Tabor, B. Zielinski, P. Spurek, Processing of missing data by neural networks, in: Proceedings of the NeurIPS, 2018.
[12] C. Cortes, G. DeSalvo, M. Mohri, Learning with rejection, in: Proceedings of the International Conference on Algorithmic Learning Theory, Springer, 2016, pp. 67–82.
[13] C. Cortes, G. DeSalvo, M. Mohri, Boosting with abstention, in: Proceedings of the NIPS, 2016.
[14] C. Cortes, G. DeSalvo, C. Gentile, M. Mohri, S. Yann, Online learning with abstention, in: Proceedings of the ICML, 2018.
[15] S. van Buuren, K. Groothuis-Oudshoorn, mice: Multivariate imputation by chained equations in R, J. Stat. Softw. 45 (3) (2011).
[16] T. Schnabel, A. Swaminatan, A. Singh, N. Chandak, T. Joachims, Recommendations as treatments: debiasing learning and evolution, in: Proceedings of the ICML, 2016.
[17] D.J. Stekhoven, P. Bühlmann, Missforest – nonparametric missing value imputation for mixed-type data, Bioinformatics 28 (1) (2011) 112–118.
[18] P.J. García-Laencina, J. Sancho-Gómez, A.R. Figueiras-Vidal, Pattern classification with missing data: a review, Neural Comput. Appl. 19 (2) (2010) 263–282.
[19] J. Yoon, J. Jordon, M. van der Schaar, GAIN: Missing data imputation using generative adversarial nets, in: Proceedings of the ICML, 2018.
[20] O. Chapelle, B. Schölkopf, A. Zien, Semi-Supervised Learning, The MIT Press, 2006.
[21] N. Sokolovska, O. Cappé, F. Yvon, The asymptotics of semi-supervised learning in discriminative probabilistic models., in: Proceedings of the ICML, 2008.
[22] T. Shpakova, F. Bach, A. Osokin, Marginal weighted maximum log-likelihood for efficient learning of perturb-and-map models, in: Proceedings of the UAI, 2018.