



HAL
open science

Multimodal Extraction of Proofs and Theorems from the Scientific Literature

Shrey Mishra

► **To cite this version:**

Shrey Mishra. Multimodal Extraction of Proofs and Theorems from the Scientific Literature. Information Retrieval [cs.IR]. Université Paris Sciences & Lettres, 2024. English. NNT : . tel-04665528

HAL Id: tel-04665528

<https://hal.science/tel-04665528v1>

Submitted on 31 Jul 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License



THÈSE DE DOCTORAT

DE L'UNIVERSITÉ PSL

Préparée à l'École normale supérieure

Multimodal Extraction of Proofs and Theorems from the Scientific Literature

Soutenue par

Shrey Mishra

Le 4 juillet 2024

École doctorale n°386

**Sciences Mathématiques de
Paris Centre**

Spécialité

Informatique

Composition du jury :

Elena Cabrio Université Côte d'Azur	<i>Rapportrice</i>
Mohammed Hasanuzzaman Queen's University Belfast	<i>Examineur</i>
Jean Ponce ENS-PSL & New York University	<i>Examineur</i>
Pierre Senellart ENS-PSL	<i>Directeur de thèse</i>
Fabian Suchanek Télécom Paris, IP Paris	<i>Rapporteur</i>
Eric Villemonte de la Clergerie Inria Paris	<i>Examineur</i>

Dedicated to my parents...



Abstract

This thesis examines the extraction of mathematical statements and proofs from scholarly PDF articles by approaching it as a multimodal classification challenge. It is part of the broader TheoremKB project, which seeks to convert scientific literature into a comprehensive, open-access knowledge base of mathematical statements and their proofs. The research leverages a range of techniques from traditional machine learning to advanced deep learning architectures, including LSTMs, CNNs, Object detectors, CRFs, transformers, etc.

The study utilizes a novel combination of text, font characteristics, and bitmap images from PDF pages as separate input modalities. It proposes a modular, sequential, multimodal machine learning strategy incorporating a cross-modal attention mechanism to produce multimodal paragraph embeddings. These embeddings are processed through a novel multimodal sliding window transformer architecture that captures sequential data across paragraphs. This innovative approach does not rely on Optical Character Recognition (OCR) preprocessing, \LaTeX sources during inference or custom pre-training on specialized losses, making it adept at handling multi-page documents and page breaks, typically in complex scientific, mathematical texts.

The findings indicate a marked performance improvement when moving from unimodal to multimodal processing and integrating sequential paragraph modelling, underscoring the effectiveness of the proposed method for handling intricate scholarly documents.



Résumé

Cette thèse étudie l'extraction d'énoncés et de preuves mathématiques à partir d'articles scientifiques PDF en l'abordant comme un problème de classification multimodale. Elle fait partie du projet plus large TheoremKB, qui cherche à convertir la littérature scientifique en une base de connaissances complète et en libre accès d'énoncés mathématiques et de leurs preuves. La recherche exploite une gamme de techniques allant de l'apprentissage automatique traditionnel aux architectures avancées d'apprentissage profond, notamment les LSTM, les CNN, les détecteurs d'objets, les CRF, les transformeurs.

L'étude exploite une combinaison originale du texte, de caractéristiques de fontes et d'images bitmap provenant des pages PDF comme modalités de saisie distinctes. Elle propose une stratégie d'apprentissage automatique multimodale modulaire et séquentielle qui intègre un mécanisme d'attention intermodale pour produire des plongements multimodaux de paragraphes. Ces plongements sont ensuite traités via une nouvelle architecture de transformeur à fenêtre glissante multimodale qui capture les données séquentielles dans les paragraphes. Cette approche innovante ne repose pas sur le prétraitement de reconnaissance optique de caractères (OCR), sur les sources \LaTeX lors de l'inférence ou sur le pré-entraînement ad hoc avec des fonctions de perte spécialisées, ce qui la rend apte à gérer des documents multipages et des sauts de page typiques des textes mathématiques scientifiques complexes.

Les résultats indiquent une nette amélioration des performances lors du passage du traitement unimodal au traitement multimodal et de l'intégration de la modélisation de paragraphes séquentiels, soulignant l'efficacité de la méthode proposée dans le traitement de documents scientifiques complexes.



Acknowledgments

I express my deepest gratitude to my thesis supervisor, Pierre Senellart, for his invaluable guidance and mentorship throughout my doctoral studies. Working with him has been an extraordinary journey of learning and discovery. The depth of our scientific discussions has inspired me to explore complex topics, and his hands-on approach confidently has significantly enhanced my technical skills. I am particularly grateful for his unwavering support during the challenging times of my illness in 2021 and his assistance in navigating various administrative procedures. His commitment to my growth and timely, constructive feedback has been fundamental to my success.

I also want to thank my thesis committee – Elena, Fabian, Eric, Jean, and Mohammed. Your willingness to evaluate my work and insights will, I am sure, improve its quality.

I extend my special thanks to Antoine, whose collaboration was instrumental in finalizing some diagrams and initial evaluations of the Conditional Random Field (CRF) model. His expertise and insightful feedback were crucial in refining the models and enhancing the overall quality of my research. Antoine's dedication and meticulous attention to detail have contributed significantly to the precision of our results and have been a source of inspiration throughout this journey.

A special acknowledgement goes to my colleagues and friends at DI ENS, particularly Antoine, Yacine, Shufan, Yann, and the broader teams within the Valda and Talgo projects. Your collaboration and the friendly working environment have considerably lightened the burdens of daily doctoral challenges. I am particularly thankful to everyone on the admin team- Linda, Tiffany, Fanny, and especially Lise-Marie for their endless support and for making administrative tasks smoother and less daunting.

This work was granted access to the HPC resources of IDRIS under the allocation 2020-AD011012097 made by GENCI (Jean Zay supercomputer), for which I am grateful. I am also grateful to the CLEPS infrastructure from the Inria of Paris for providing resources and support.

To my family – my father, Deepak Mishra; my mother, Jyotsna Mishra; and my little sister, Swasti Mishra – your love, sacrifice, and encouragement have been my pillars of strength. Moving abroad for my studies was more manageable, knowing I had your unconditional support and belief in my capabilities. The distance from home was challenging, but your constant support and care made it bearable and fueled my determination.

I am also immensely grateful to my friends – Shirish, Anushka, Omkar, Claudio, Veera, Madhavee, Hitesh, Sukh, Subham, Neil, Anadi, Anastasia, Samuel and Prakash, thank you for enriching my life with joy, companionship, and memorable moments. Each of you has played a pivotal role in making my doctoral journey bearable, delightful, and memorable.

In addition, I must thank every academic and support staff member at DI ENS whose daily interactions, big or small, have impacted my academic and personal life. Though perhaps unmentioned, your contributions to my journey have not gone unappreciated.

Finally, my journey would not have been the same without the broader academic community and all those who have indirectly influenced my work and life during these years. Your roles in this journey, though unseen, have been vital.

Thank you all for being part of my doctoral adventure and contributing to my experiences' tapestry.

Contents

Abstract	iii
Résumé	v
Acknowledgments	vii
1 Introduction	1
1.1 General Introduction	1
1.2 Rationale and Manuscript Organization	2
1.2.1 Chapter 3: TheoremKB Project	2
1.2.2 Chapter 4: Line-Based Extraction	2
1.2.3 Chapter 5: Unimodal Approaches	3
1.2.4 Chapter 6: Multimodal Approaches	3
1.2.5 Chapter 7: Sequential Paragraph Approaches	3
1.2.6 Chapter 8: Discussions and Conclusions	4
1.3 Other Contributions	4
1.4 Closing Remarks	4
2 Introduction en français	7
2.1 Introduction générale	7
2.2 Justification et organisation du manuscrit	8
2.2.1 Chapitre 3 : Le projet TheoremKB	8
2.2.2 Chapitre 4 : Extraction ligne par ligne	9
2.2.3 Chapitre 5 : Approches unimodales	9
2.2.4 Chapitre 6 : Approches multimodales	9
2.2.5 Chapitre 7 : Approches séquentielles sur les paragraphes	10
2.2.6 Chapitre 8 : Discussions et conclusions	10
2.3 Autres contributions	10
2.4 Remarques finales	10
3 TheoremKB Project	13
3.1 What is the TheoremKB Project?	13
3.2 Why is TheoremKB Important?	14
3.3 TheoremKB Proposal	16
3.4 Information Extraction	17
3.4.1 Collaborative Research Insights by TheoremKB team	17
3.4.2 Daria's Preliminary Work	17
3.4.3 Théo's Work on the Graph of Mathematical Results	18
3.4.4 Lucas's Work on Extraction through CRFs	20
3.4.5 Own Preliminary Work published at DocEng 2021	20
3.4.6 Yacine's Work on Connecting Theorems Across Papers	23
3.4.7 Antoine's work on Document Class Inference	24
3.4.8 Own Work on Multimodal Machine Learning	25
3.4.9 Ilyas's work on Embedding Equation Structures	26
3.4.10 Shufan's work on Extracting Definienda	27
3.4.11 Own Work on Modular Multimodal Extraction	28
3.5 How is the Extraction Task Linked to other Tasks?	29

3.6	Dataset Construction and Preprocessing Tools	32
3.6.1	Collecting and Filtering Dataset	32
3.6.2	Using L ^A T _E X Sources for Ground Truth Labeling	36
3.6.3	Producing an Annotated XML File	37
3.6.4	Related Datasets	39
3.6.5	Problems with Extraction	42
4	Line-Based Extraction	45
4.1	Introduction to the Task	45
4.2	Related Work	47
4.2.1	Computer Vision: Object Detector	47
4.2.2	Natural language Processing: Transformer-Based	48
4.2.3	Styling and Sequence: CRF based	49
4.3	Experimental Setup and Results	49
4.4	Discussion of the Results	55
4.4.1	Conclusions	55
4.4.2	Limitations	57
5	Unimodal Approaches	59
5.1	Introduction	59
5.1.1	From Lines to Paragraphs	60
5.1.2	Related Work (Task)	61
5.2	Font Modality	62
5.2.1	Fonts to Features	62
5.2.2	Exploratory Data Analysis – Fonts	64
5.2.3	Simple ML Classifiers	65
5.2.4	Feature-Level Importance	68
5.2.5	Preliminary Experimental Results – LSTM Model	69
5.2.6	Font Model Evaluation on Large Dataset	71
5.3	Vision Modality	74
5.3.1	Image Preprocessing	74
5.3.2	Evolution of Neural Network Architectures in Computer Vision	77
5.3.3	Preliminary Experimental results – CNNs	83
5.3.4	Evaluation on Large Dataset	87
5.4	Text Modality	88
5.4.1	Related Work	89
5.4.2	Preleminary Work (Finetuning Models, Pretraining Language Model)	95
5.4.3	Evaluation of Language Model on Large Dataset	95
6	Multimodal Approaches	101
6.1	Introduction	101
6.2	Preleminary Results – Multimodal	101
6.2.1	Manual labelling to filter outliers	102
6.2.2	Stacking Classifier – Multimodal	103
6.3	Related Work	104
6.3.1	Multimodal Deep Learning (using Cross-Modality Losses)	105
6.3.2	Multimodal Deep Learning (using Feature-Level Fusion)	106
6.4	Evaluation of Multimodal Model on Large Dataet	110
7	Sequential Approaches	115
7.1	Introduction	115
7.2	Related Work	116
7.2.1	Conditional Random Fields	116
7.2.2	BiLSTMs	117
7.2.3	Transformers	117
7.2.4	Transforming Attention mechanism to handle long Sequences	118
7.2.5	Hierarchical Attention Transformers	119
7.3	Evaluation of Sequential Model on Large Dataset	121

8	Discussions and Conclusions	127
8.1	Introduction	127
8.2	Conclusions	128
8.3	Post-Hoc Explanations	129
8.4	Limitations	129
8.5	Ethical considerations	131
A	Classification Report	133
A.1	Font Sequence Models	133
A.2	Vision-Based Models	135
A.3	NLP Models	138
A.4	Multimodal Models	139
A.5	Sequential Paragraph Models	143
A.5.1	CRF-Based	143
A.5.2	Transformer-Based	145

Introduction

Chapter content

1.1	General Introduction	1
1.2	Rationale and Manuscript Organization	2
1.2.1	Chapter 3: TheoremKB Project	2
1.2.2	Chapter 4: Line-Based Extraction	2
1.2.3	Chapter 5: Unimodal Approaches	3
1.2.4	Chapter 6: Multimodal Approaches	3
1.2.5	Chapter 7: Sequential Paragraph Approaches	3
1.2.6	Chapter 8: Discussions and Conclusions	4
1.3	Other Contributions	4
1.4	Closing Remarks	4

This thesis contributes to the TheoremKB project [Sen19], which aims to convert scientific papers – specifically PDFs – into a navigable knowledge base of proofs and theorems, pivotal for scholarly research. This endeavour is intended to enhance literature review processes and address foundational questions, such as evaluating the impact of discredited results on related studies. Such a knowledge base would enable authors to easily track how their results are utilized in subsequent publications, an essential feature given the complexity of mathematical papers that often include numerous proofs and theorems.

The project involves two main stages: initially extracting proofs from various scientific papers and then linking these findings to relevant existing results. This thesis focuses on the extraction phase, using real-world datasets from papers available on arXiv, but, crucially, without requiring L^AT_EX sources at the point of inference. This approach ensures broader applicability, given that PDFs are a standard format for storing academic content. Unlike current search systems like Google Scholar¹, Microsoft Academic² (terminated in 2021), or AMiner³, which rely on keyword-based searches, this knowledge base aims for a contextual understanding, linking documents that directly pertain to specific scientific inquiries, such as all studies addressing the NP-hardness of the vertex cover problem, rather than merely containing related terms.

1.1 General Introduction

This thesis presents a novel approach using a Sliding Window Transformer (SW Transformer) within a multimodal framework, where font, bitmap images, and text are utilized as primary modalities. This work is distinctive because it automates the labelling process and eschews manual feature engineering, setting it apart from typical Document AI applications [XLC⁺20, XXL⁺21, HLC⁺22, WJD22] that often rely on Optical Character Recognition (OCR) or pretraining with specialized loss functions to handle various modalities.

The thesis stands out by:

Automating the labelling process: This allows for an efficient, scalable method to handle large datasets, exemplified by the 0.5 million paragraphs used for testing.

¹<https://scholar.google.com>

²<https://www.microsoft.com/en-us/research/project/academic/>

³<https://www.aminer.org/>

Avoiding OCR: By bypassing OCR, the method maintains high fidelity to the original document formats and avoids common OCR errors, making it robust across diverse document types.

Modular integration of modalities: Instead of using a single pretraining phase that tries to blend This approach uses modular integration for all modalities. This strategy enhances the model’s adaptability and effectiveness by focusing on each modality’s strengths.

Handling long documents: Unlike many Document AI systems that assume all relevant information fits on a single page, this method is adept at handling long documents, such as scientific papers where proofs may span multiple pages and include several page breaks.

The combination of these elements showcases the approach’s versatility and depth and underscores its potential applicability in academic and professional settings where document analysis is crucial. This work could significantly influence future research and applications in knowledge extraction from documents similar to scientific articles.

This thesis aims to establish a foundational pipeline for extracting theorems and proofs from scientific documents, emphasizing modularity, efficiency in inference time, and minimizing parameter count – critical factors for deploying models in production settings. By focusing on modularity, the approach facilitates updates and improvements, allowing for integrating advanced, more capable models as they become available and ensuring compatibility with various deployment hardware. Additionally, the thesis explores the provision of post hoc explanations in simple use cases to verify that the models capture the intended features effectively, enhancing the interpretability and trustworthiness of the models in practical applications.

1.2 Rationale and Manuscript Organization

To enhance the accessibility and appeal of my thesis, I have incorporated numerous visual elements, such as architectural diagrams, while deliberately avoiding complex mathematical expressions. This approach aims to make the thesis comprehensible and engaging to a broader audience, extending beyond those with a deep technical background. The methodologies and models discussed are designed to serve as practical abstractions for readers interested in applying similar techniques or those involved in comparable tasks who prefer to avoid developing models from scratch. For individuals interested in utilizing these models through a transfer learning approach, I have made all related resources accessible via a GitHub repository. This allows users to apply the models to their projects without a detailed understanding of their underlying mechanics.

The thesis is structured into several chapters, each focusing on different aspects of the project and providing detailed insights into the methodologies employed.

1.2.1 Chapter 3: TheoremKB Project

Chapter 3 provides a comprehensive overview of the TheoremKB project and its significance, serving as a foundational piece published at the 4th Workshop on Scholarly Document Processing (SDP, 2024) at ACL 2024 [MBD⁺24]. This chapter delves into the project’s core objectives, addressing crucial questions regarding its aims, importance, and potential impact. Additionally, it highlights the collective contributions of various individuals to the project. It underscores the central role of the work presented in this thesis in shaping and advancing related endeavours within the group.

A key focus of Chapter 3 is contextualizing the project by discussing the broader challenges and preprocessing requirements involved. It explores essential tools such as Grobid [L⁺24a] and pdfalto [L⁺24b], shedding light on their functionalities and potential limitations, including failure cases. By doing so, the chapter lays the groundwork for understanding the motivation behind the project. It introduces readers to the overarching task and its significance within the scholarly document processing domain.

1.2.2 Chapter 4: Line-Based Extraction

Chapter 4 of the thesis showcases a proof of concept for the extraction task by employing various deep learning methodologies for line-level prediction, utilizing well-known architectures from Computer Vision and NLP. This work was presented at the DocEng conference in 2021.

The core objective of this chapter is to evaluate straightforward, established deep-learning approaches such as fine-tuning large language models (LLMs) on text classification tasks and employing computer

vision-based object detectors to identify proofs and theorems directly from raw bitmap images of scientific papers. This exploration serves as a precursor and provides foundational insights that motivate the transition towards a multimodal approach that integrates both textual and visual data for more robust theorem and proof detection.

1.2.3 Chapter 5: Unimodal Approaches

Chapter 5 of the thesis is dedicated to developing unimodal backbones for different modalities, which serve as feature backbones for the extraction task. This chapter primarily focuses on creating robust language, vision, and font modalities models. A significant highlight of this section is the introduction of a pre-trained model developed specifically for scientific texts. This model, trained on a corpus that is one tenth the size but highly relevant compared to a generic English corpus, achieves performance levels comparable to those of models trained on broader English data. This development emphasizes the effectiveness of domain-specific training in achieving high accuracy with relatively smaller and more focused datasets.

Additionally, the chapter offers detailed insights into the convergence behaviours of various unimodal approaches, discussing how much data is necessary for practical training and stabilization of these models. It also extensively reviews related work for laying the groundwork for these unimodal foundations, situating the discussion within the broader context of existing research in document processing and machine learning on each modality.

This section’s contributions have been documented and shared in a preprint available on arXiv since February 2023 [MGS23], broadening access to the methodologies and findings presented in this significant segment of the thesis.

1.2.4 Chapter 6: Multimodal Approaches

Chapter 6 showcases the integration of unimodal approaches, as detailed in Chapter 5, to achieve multimodal fusion. Unlike many document AI models that rely on joint training modalities with unique pretraining losses, this section emphasizes feature-level fusion without altering the foundational loss functions, sticking to standard cross-entropy loss. The primary aim is to explore various deep-learning strategies that enhance the primary method of simple concatenation, often considered the most rudimentary form of fusion. These advanced methods significantly outperform simple concatenation without substantially increasing the number of parameters.

The chapter uses models directly from Chapter 5, which are kept frozen during experiments to ensure no weight updates. This approach guarantees a fair comparison and consistency across tests, which departs from some practices in document AI research where backbones and loss functions are frequently modified concurrently. By focusing on the interoperability of features from distinct modalities – such as text, vision, and font – this chapter illustrates how integrating these can enhance the overall model performance by capturing complex intermodality relationships effectively.

1.2.5 Chapter 7: Sequential Paragraph Approaches

Chapter 7 considers enhancing the model’s capability to capture long-term dependencies across paragraphs. It discusses the potential of utilizing the context provided by preceding and subsequent paragraphs to predict the label of the current paragraph. The chapter begins with a CRF-based probabilistic model as a baseline for experiments and then transitions to exploring more complex architectures suited for handling sequential data.

Transformers are highlighted as a core focus, with specific attention given to the Sliding Window Transformer and standard transformer architectures as primary candidates. Additionally, Hierarchical Transformers [CDF⁺22], akin to models such as Longformer [BPC20] and BigBird [ZGD⁺20], which have proven effective in unimodal text settings, are considered for adapting to handle extensive documents in a multimodal context. These hierarchical models are especially relevant for integrating multimodal features from Chapter 6, aiming to enrich the model’s understanding of interrelated paragraph-level data.

This chapter and Chapters 5 and 6 are part of a comprehensive article submitted for publication at the date of the final version of this thesis. Updates, including the potential acceptance and publication of these findings, will be added to the original arXiv preprint upon final publication.

1.2.6 Chapter 8: Discussions and Conclusions

Chapter 8 synthesizes the contributions of the thesis and addresses the limitations and ethical considerations of applying the research in real-world scenarios. In appendix is provided a comprehensive classification report that outlines per-class F_1 scores and accuracies, offering a clear view of each model’s performance.

The chapter concludes with reflections on the thesis’s contributions to the multimodal domain, aiming to provide a robust foundation for future researchers and practitioners. The intent is to present engaging insights that are informative and accessible without compromising the depth of content, facilitating further exploration and application in this field.

1.3 Other Contributions

Throughout my PhD, I had the opportunity to present my research on the TheoremKB project at various workshops and conferences, allowing me to gather feedback and insights on how to enhance my work. My research was showcased in various formats, such as posters, conference talks, and workshop sessions. Notable venues include the PAISS summer school, the Oxford Machine Learning Summer School, the MDD Summer School, the EEML Summer School, the 3IA Doctoral Workshop, as well as significant conferences like NeurIPS-in-Paris in 2022, SIGKDD 2023, and DocEng 2021. These presentations and forums provided valuable platforms for discussing my project with other experts in the field, receiving constructive criticism, and learning about recent advancements in related research areas.

We plan to develop a demo (and publish a corresponding demo paper) to showcase the final proposed model’s capability to extract proofs and theorems from raw PDFs via a user-friendly RESTful API. The demo is intended to demonstrate the model’s applicability on a much larger corpus, extending beyond the 200,000 papers used in training.

1.4 Closing Remarks

The thesis, enriched by extensive experimentation across various modalities and feature backbone designs, achieves significant progress in automatically extracting proofs and theorems from a substantial dataset of about five hundred thousand paragraphs derived from roughly 4000 PDFs. Despite the considerable size of the validation dataset, it remains a sample of a larger potential corpus, highlighting a limitation in scope due to computational constraints that restricted the experiments to smaller models and a finite number of documents. Future work aims to expand the dataset significantly to include millions of papers, potentially covering the entire scope of arXiv and beyond, to validate the robustness of the findings across a broader range of documents.

Manual labelling played a vital role in this research. However, it proved challenging due to the complex nature of the task, suggesting that while the results from automated labelling and modelling are promising, they could be refined with minimal human intervention.

The practical aspects of the thesis also provided a valuable learning opportunity in utilizing various tools like TensorFlow, Grobid, and pdfto, supported by the computational power of the IDRIS Jean Zay supercomputer and the Inria CLEPS cluster, blending theoretical and practical elements effectively.

Although there is potential for more significant performance improvements with more advanced models, the scope of this thesis was limited to foundational models due to the vast scale of experiments across multiple modalities. The rapidly evolving field of deep learning presents a challenge in maintaining cutting-edge methodologies; however, the flexible design of the proposed pipeline should facilitate future updates and integrations of newer technologies and models.

During my PhD, I collaborated with several Master’s students – Vikash Kulhari, Yacine Brihmouche, Antoine Gauquier, and Postdoctoral researcher Shufan Jiang – on various aspects of the TheoremKB project. The models and training code I worked on also accelerated their development and helped them avoid some of the initial challenges I encountered.

Throughout my thesis, I benefited greatly from discussions with Pierre Senellart, whose guidance was instrumental in exploring my field of interest. I am also grateful for the valuable feedback from Jean Ponce and Eric de la Clergerie, which oriented some of my research efforts.

Attending summer schools, conferences, and workshops allowed me to engage personally with renowned researchers like Edouard Grave, Yann LeCun, and Michael Bronstein just to name a few. This invaluable exposure provided insights that significantly enriched my understanding and approach to the project.

This work, heavily reliant on recent advancements in computing and algorithms, would not have been feasible a few years ago. Witnessing part of the AI revolution firsthand has been a remarkable experience.

All training codes and models from this project are publicly available under a Creative Commons license and can be accessed at <https://github.com/PierreSenellart/theoremkb>.

Introduction en français

Contenu du chapitre

2.1	Introduction générale	7
2.2	Justification et organisation du manuscrit	8
2.2.1	Chapitre 3 : Le projet TheoremKB	8
2.2.2	Chapitre 4 : Extraction ligne par ligne	9
2.2.3	Chapitre 5 : Approches unimodales	9
2.2.4	Chapitre 6 : Approches multimodales	9
2.2.5	Chapitre 7 : Approches séquentielles sur les paragraphes	10
2.2.6	Chapitre 8 : Discussions et conclusions	10
2.3	Autres contributions	10
2.4	Remarques finales	10

Cette thèse contribue au projet TheoremKB [Sen19], qui vise à convertir des articles scientifiques, plus particulièrement au format PDF, en une base de connaissances navigable de preuves et de théorèmes, essentielle à la recherche scientifique. Cet effort vise à améliorer les processus d’analyse de la littérature et à répondre aux questions fondamentales, comme évaluer l’impact de résultats qui s’avèrent faux sur d’autres résultats. Une telle base de connaissances permettrait aux auteurs de suivre facilement la manière dont leurs résultats sont utilisés dans les publications ultérieures, ce qui est essentiel compte tenu de la complexité des articles mathématiques qui comprennent souvent de nombreux théorèmes et preuves.

Le projet comporte deux étapes principales : dans un premier temps extraire énoncés mathématiques et preuves depuis les articles scientifiques, et dans un deuxième temps relier preuves et théorèmes (en particulier, déterminer quel théorème est utilisé dans la preuve de quel résultat). Cette thèse se concentre sur la première phase d’extraction, en utilisant des jeux de données du monde réel provenant d’articles disponibles sur arXiv, mais, de manière importante, sans se baser sur les sources (typiquement au format L^AT_EX) des articles au moment de l’inférence. Cette approche garantit une applicabilité plus large étant donné que les articles sont souvent disponibles uniquement sous la forme de documents PDF. Contrairement aux moteurs de recherche comme Google Scholar¹, Microsoft Academic² (arrêté en 2021) ou AMiner³, qui s’appuient sur des recherches par mots-clés, cette base de connaissances vise à une compréhension contextuelle, reliant les documents et leurs résultats se rapportent directement à des sujets donnés, comme par exemple toutes les études incluant des théorèmes de NP-difficulté de variante du problème de couverture par des sommets, plutôt que simplement les articles contenant les termes associés.

2.1 Introduction générale

Cette thèse présente une nouvelle approche en employant un transformeur à fenêtre coulissante dans un cadre multimodal, où les fontes (variantes de polices de caractère), les images bitmap et le texte sont utilisés comme modalités principales. Ce travail se distingue en automatisant le processus d’étiquetage sans nécessiter de construction manuelle de *features*; il se distingue également d’approches standard d’IA documentaire [XLC+20, XXL+21, HLC+22, WJD22] qui reposent souvent sur la reconnaissance optique

¹<https://scholar.google.com>

²<https://www.microsoft.com/en-us/research/project/academic/>

³<https://www.aminer.org/>

de caractères (OCR) ou sur un pré-entraînement avec des fonctions de pertes spécialisées pour gérer les diverses modalités.

Plus précisément, la thèse se démarque par :

Automatisation du processus d'étiquetage : Cela permet une évaluation efficace qui passe à l'échelle sur de grands jeux de données, ce que l'on illustre par les 0,5 million de paragraphes utilisés pour les tests.

Éviter l'OCR : En contournant la reconnaissance optique de caractères (OCR), la méthode maintient une haute fidélité qui suit le format des documents originaux et évite les erreurs d'OCR courantes, ce qui le rend robuste sur divers types de documents.

Intégration modulaire des modalités : Au lieu d'utiliser une seule phase de pré-entraînement qui tente de mélanger toutes les modalités, cette approche utilise une intégration modulaire. Cette stratégie améliore l'adaptabilité et l'efficacité du modèle en se concentrant sur les atouts spécifiques de chaque modalité.

Gestion de documents longs : Contrairement à de nombreux systèmes d'IA documentaire qui supposent que toutes les informations pertinentes tiennent sur une seule page, cette méthode est adaptée au traitement de documents longs, tels que des articles scientifiques où les preuves peuvent s'étendre sur plusieurs pages et inclure plusieurs sauts de page.

La combinaison de ces éléments met non seulement en valeur la polyvalence et la profondeur de l'approche, mais souligne également son applicabilité potentielle dans les milieux universitaires et professionnels où l'analyse de documents est cruciale. Ces travaux pourraient influencer de manière significative les recherches et applications futures sur l'extraction d'information depuis des documents similaires aux articles scientifiques.

Cette thèse vise principalement à établir un *pipeline* de base pour la tâche d'extraction des théorèmes et des preuves à partir de documents scientifiques, en considérant tout particulièrement la modularité, l'efficacité en terme de temps d'inférence, et en minimisant le nombre de paramètres, facteurs clés pour le déploiement de modèles dans les environnements de production. En se concentrant sur la modularité, l'approche facilite les mises à jour et les améliorations, permettant l'intégration de modèles avancés et plus performants au fur et à mesure qu'ils deviennent disponibles et garantissant la compatibilité avec diverses implémentations matérielles. De plus, la thèse explore la fourniture d'explications *post hoc* dans des cas d'utilisation simples pour vérifier que les modèles capturent efficacement les fonctionnalités prévues, améliorant ainsi l'interprétabilité et la fiabilité des modèles dans des applications pratiques.

2.2 Justification et organisation du manuscrit

Dans le but d'améliorer l'accessibilité et la lisibilité de ma thèse, j'ai incorporé de nombreux éléments visuels tels que des schémas d'architecture tout en évitant délibérément les formalismes mathématiques complexes. Cette approche vise à rendre la thèse compréhensible et engageante pour un public plus large, au-delà de lecteurs avec une solide expérience technique. Les méthodologies et modèles discutés sont conçus pour servir d'abstractions pratiques aux lecteurs intéressés par l'application de techniques similaires ou qui sont impliqués dans des tâches comparables mais préfèrent ne pas développer de modèles à partir de zéro. Pour toute personne intéressée par utiliser ces modèles, par exemple pour une approche d'apprentissage par transfert, j'ai rendu toutes les ressources accessibles via un dépôt GitHub. Cela permet aux utilisateurs d'appliquer les modèles à leurs propres projets sans avoir besoin d'une compréhension détaillée de leurs mécanismes sous-jacents.

La thèse est structurée en plusieurs chapitres, chacun se concentrant sur différents aspects du projet et fournissant des informations détaillées sur les méthodologies utilisées.

2.2.1 Chapitre 3 : Le projet TheoremKB

Le chapitre 3 donne un aperçu complet du projet TheoremKB et de son importance, servant de vision de haut niveau, dont est tirée une publication au 4ème atelier sur le traitement des documents scientifiques (SDP 2024), un atelier ACL 2024 [MBD⁺24]. Ce chapitre approfondit les objectifs fondamentaux du projet, abordant des questions cruciales concernant ses objectifs, son importance et son impact potentiel. De plus, il met en lumière les contributions collectives de diverses personnes au projet et souligne le rôle central du travail présenté dans cette thèse pour façonner et faire progresser les efforts connexes au sein du groupe.

Un objectif clé du chapitre 3 est de contextualiser le projet en discutant des défis plus larges et des besoins nécessaires en terme de pré-traitement. Il explore des outils essentiels tels que Grobid [L+24a] et pdfalto [L+24b], mettant en lumière leurs fonctionnalités et leurs limites potentielles, y compris les cas d’erreurs. Ce faisant, le chapitre jette les bases pour comprendre la motivation derrière le projet et présente aux lecteurs la tâche principale abordée dans cette thèse et son importance dans le domaine du traitement des documents scientifiques.

2.2.2 Chapitre 4 : Extraction ligne par ligne

Le chapitre 4 de la thèse présente une preuve de concept pour la tâche d’extraction en employant divers méthodologies d’apprentissage profond pour la prédiction au niveau de chaque *ligne* du document PDF, utilisant des architectures bien connues des domaines de la vision par ordinateur et du TAL. Ce travail a été accepté et présenté à la conférence DocEng en 2021.

L’objectif principal de ce chapitre est d’évaluer des approches d’apprentissage profond simples et établies, tels que l’ajustement de grands modèles de langage (LLM) sur les tâches de classification de texte ; et l’utilisation de détecteurs d’objets de vision par ordinateur pour identifier les preuves et les théorèmes directement à partir d’images bitmap brutes des articles scientifiques. Cette exploration sert de travail précurseur et fournit des informations fondamentales qui motivent la transition vers une approche multimodale intégrant à la fois des données textuelles et visuelles pour une détection de théorème et de preuve plus robuste.

2.2.3 Chapitre 5 : Approches unimodales

Le chapitre 5 de la thèse est consacré au développement d’approches unimodales pour différentes modalités. Ce chapitre se concentre principalement sur la création de modèles robustes pour les modalités de langage, de vision et de fonte. Un point fort de cette section est l’introduction d’un modèle pré-entraîné développé spécifiquement pour les textes scientifiques. Ce modèle, créé à partir d’un corpus de taille dix fois plus petite mais avec un contenu spécifique au domaine par rapport à un corpus générique en anglais, atteint des niveaux de performances comparables à ceux des modèles formés sur des corpus plus larges. Ce développement met l’accent sur l’efficacité du pré-entraînement spécifique à un domaine pour atteindre une grande précision avec des jeux de données relativement plus petits et plus ciblés.

De plus, le chapitre offre un aperçu détaillé des comportements de convergence de divers approches unimodales, discutant de la quantité de données nécessaires à un entraînement efficace et une stabilisation de la performance de ces des modèles. Il passe également en revue de manière approfondie les travaux connexes essentiels à la base de ces systèmes unimodaux, plaçant la discussion dans le contexte plus large des recherches existantes dans le traitement de documents et de l’apprentissage pour chaque modalité.

Les contributions de cette section ont été documentées et partagées dans une prépublication disponible sur arXiv depuis Février 2023 [MGS23], pour faciliter l’accès aux méthodologies et aux résultats présentés dans cette partie importante de la thèse.

2.2.4 Chapitre 6 : Approches multimodales

Le chapitre 6 se penche sur l’intégration des approches unimodales détaillées au chapitre 5 pour parvenir à une approche multimodale à base de fusion. Contrairement à de nombreux modèles d’IA documentaires qui reposent sur un entraînement joint des différentes modalités avec des fonctions de pertes de pré-entraînement communes, cette partie met l’accent sur la fusion au niveau des *features* sans modifier les fonctions de perte classiques, en s’en tenant à la perte d’entropie croisée standard. L’objectif principal est d’explorer diverses stratégies d’apprentissage profond qui améliorent la méthode de base de concaténation simple, qui est souvent considérée comme la forme de fusion la plus rudimentaire. Il a été démontré que ces méthodes avancées surpassent considérablement la simple concaténation, sans nécessiter une augmentation substantielle du nombre de paramètres.

Le chapitre utilise des modèles directement issus du chapitre 5, qui sont conservés avec poids gelés pendant les expériences. Cette approche garantit une comparaison équitable et une cohérence entre tests, ce qui s’écarte de certaines pratiques de recherche sur l’IA documentaire où les *backbones* et les fonctions de perte sont fréquemment modifiées simultanément. En misant sur l’interopérabilité des *features* de modalités distinctes, telles que le texte, la vision et la fonte, ce chapitre illustre comment l’intégration de celles-ci peut améliorer les performances globales du modèle en capturant des relations intermodales complexes de manière effective.

2.2.5 Chapitre 7 : Approches séquentielles sur les paragraphes

Le chapitre 7 se penche sur l'amélioration de la capacité du modèle à capturer les dépendances à long terme entre les paragraphes. Il discute de la possibilité d'utiliser le contexte fourni par les paragraphes précédents et suivants pour prédire l'étiquette du paragraphe actuel. Le chapitre commence par un modèle probabiliste basé sur les champs aléatoires conditionnels (CRF) comme référence pour les expériences, puis passe à l'exploration d'architectures plus complexes adaptées à la gestion des données séquentielles.

Les transformeurs sont utilisés de manière principale, avec une attention particulière accordée au transformeur à fenêtre coulissante et en utilisant des architectures standards de transformer comme principaux candidats. De plus, les transformeurs hiérarchiques [CDF⁺22], semblables à des modèles tels que Longformer [BPC20] et BigBird [ZGD⁺20], qui se sont révélés efficaces dans les cadres textuels unimodaux, sont envisagés pour être adaptés à la gestion de documents volumineux dans un contexte multimodal. Ces modèles hiérarchiques sont particulièrement pertinents pour intégrer les fonctionnalités multimodales du chapitre 6, visant à enrichir la compréhension du modèle des données interdépendantes au niveau des paragraphes.

Ce chapitre, ainsi que les chapitres 5 et 6, est soumis pour publication à la date de rédaction de la version finale de cette thèse. Son contenu est également destiné à être intégré à la prépublication arXiv déjà citée.

2.2.6 Chapitre 8 : Discussions et conclusions

Le chapitre 8 synthétise les contributions de la thèse et aborde les limites et considérations éthiques de l'application aux scénarios du monde réel. En annexe est disponible un rapport de classification complet qui décrit les scores F_1 et précisions par classe, offrant une vue claire des performances de chaque modèle.

Le chapitre se termine par des réflexions sur les contributions de la thèse au domaine multimodal, visant à fournir une base de travail aux futurs chercheurs et praticiens. L'intention est de présenter des informations engageantes, à la fois informatives et accessibles, sans compromettre la profondeur du contenu, facilitant ainsi une exploration et une application plus approfondies dans ce domaine.

2.3 Autres contributions

Tout au long de mon doctorat, j'ai eu l'occasion de présenter mes recherches sur le projet TheoremKB lors de divers ateliers et conférences, me permettant de recueillir des commentaires et des idées sur la façon d'améliorer mon travail. Mes recherches ont été présentées sous divers formats tels que des posters, des exposés et des ateliers. Les lieux notables incluent l'école d'été PAISS, l'Oxford Machine Learning Summer School, la MDD Summer School, l'EEML Summer School, l'Atelier Doctoral 3IA ; j'ai également participé à de grands événements et conférences comme NeurIPS-in-Paris en 2022, SIGKDD 2023 et DocEng 2021. Ces présentations et forums ont fourni de précieuses plateformes pour discuter de mon projet avec d'autres experts du domaine, recevoir des critiques constructives, et en apprendre davantage sur les avancées récentes dans des domaines de recherche connexes.

Nous travaillons actuellement à préparer une démonstration, avec publication d'article correspondant, visant à présenter l'utilisation et les performances du modèle final proposé, avec la possibilité d'extraire des preuves et théorèmes à partir de PDF bruts via une API RESTful facile d'usage. La démonstration est destinée à démontrer l'applicabilité du modèle sur un corpus plus large, s'étendant au-delà des 200 000 articles utilisés dans cette thèse.

2.4 Remarques finales

La thèse, enrichie par une expérimentation approfondie à travers diverses modalités et divers modèles de *backbones* de *features*, réalise des progrès significatifs dans l'extraction automatique de preuves et de théorèmes à partir d'un ensemble de données substantiel d'environ cinq cent mille paragraphes dérivés d'environ 4 000 PDF. Malgré la taille considérable de l'ensemble de données de validation, cela reste un échantillon d'un corpus potentiel plus large, mettant en évidence une limitation de la portée de notre travail due aux contraintes de calcul qui ont limité les expériences à des modèles plus petits et à un nombre borné de documents. Les travaux futurs visent à élargir l'ensemble de données de manière significative pour inclure des millions d'articles, couvrant potentiellement toute la portée d'arXiv et au-delà, pour valider la robustesse des résultats sur un plus large éventail de documents.

L'étiquetage manuel a joué un rôle essentiel dans cette recherche. Cependant, cela s'est avéré difficile en raison de la nature complexe de la tâche, ce qui suggère que même si les résultats de l'étiquetage et de la modélisation automatisés sont prometteurs, ils pourraient être affinés par une intervention humaine minimale.

Les aspects pratiques de la thèse ont également fourni une précieuse opportunité d'apprendre à utiliser divers outils tels que TensorFlow, Grobid et pdfalto, soutenus par la puissance de calcul du supercalculateur IDRIS Jean Zay et du Pôle Inria CLEPS, mêlant efficacement éléments théoriques et pratiques.

Bien qu'il existe un potentiel d'amélioration des performances plus significatif avec des modèles plus avancés, la portée de cette thèse était limitée à des modèles de référence en raison de la vaste échelle d'expériences dans de multiples modalités. Le domaine en évolution rapide qu'est l'apprentissage profond représente un défi pour maintenir des méthodologies de pointe ; cependant, la conception flexible de la chaîne de traitements proposée devrait faciliter les futures mises à jour et l'intégrations de technologies et de modèles plus récents.

Durant mon doctorat, j'ai collaboré avec plusieurs étudiants en Master – Vikash Kulhari, Yacine Brihmouche, Antoine Gauquier – et la chercheuse postdoctorale Shufan Jiang sur divers aspects du projet TheoremKB. Les modèles et le code d'entraînement sur lequel j'ai travaillé ont aussi accéléré leur développement et les a aidés à éviter certains des premiers écueils que j'ai rencontrés.

Tout au long de ma thèse, j'ai grandement bénéficié des échanges avec Pierre Senellart, dont les conseils ont été déterminants pour explorer mon domaine d'intérêt. Je suis également reconnaissant pour les précieux retours de Jean Ponce et Eric de la Clergerie, qui ont en partie orienté mes efforts de recherche.

Participer à des écoles d'été, des conférences et des ateliers m'a permis de côtoyer personnellement des chercheurs de renom comme Edouard Grave, Yann LeCun et Michael Bronstein pour n'en nommer que quelques-uns. Ces découvertes inestimables ont considérablement enrichi ma compréhension et mon approche du projet.

Ce travail, fortement tributaire des récents progrès de l'informatique et des algorithmes, n'aurait pas été réalisable il y a quelques années. Être témoin d'une partie de la révolution de l'IA a été une expérience remarquable.

Tous les codes et modèles d'apprentissage de ce projet sont accessibles publiquement sous licence Creative Commons à l'adresse <https://github.com/PierreSenellart/theoremkb>.

TheoremKB Project

Contenu du chapitre

3.1	What is the TheoremKB Project?	13
3.2	Why is TheoremKB Important?	14
3.3	TheoremKB Proposal	16
3.4	Information Extraction	17
3.4.1	Collaborative Research Insights by TheoremKB team	17
3.4.2	Daria's Preliminary Work	17
3.4.3	Théo's Work on the Graph of Mathematical Results	18
3.4.4	Lucas's Work on Extraction through CRFs	20
3.4.5	Own Preliminary Work published at DocEng 2021	20
3.4.6	Yacine's Work on Connecting Theorems Across Papers	23
3.4.7	Antoine's work on Document Class Inference	24
3.4.8	Own Work on Multimodal Machine Learning	25
3.4.9	Ilyas's work on Embedding Equation Structures	26
3.4.10	Shufan's work on Extracting Definienda	27
3.4.11	Own Work on Modular Multimodal Extraction	28
3.5	How is the Extraction Task Linked to other Tasks?	29
3.6	Dataset Construction and Preprocessing Tools	32
3.6.1	Collecting and Filtering Dataset	32
3.6.2	Using L ^A T _E X Sources for Ground Truth Labeling	36
3.6.3	Producing an Annotated XML File	37
3.6.4	Related Datasets	39
3.6.5	Problems with Extraction	42

3.1 What is the TheoremKB Project?

There are nearly 2.5 million papers submitted to arXiv¹ (as of 15th March, 2024), a significant portion (around 30% every month) includes mathematical information such as Theorems and Proofs (see Figures 3.1 and 3.2).

These papers span a wide range of fields, not limited to mathematics but extending to economics, computer science, physics, and many other domains. While Figure 3.1 showcases papers only from arXiv, many other sources indicate a number of several orders of magnitude higher of such papers, indicating arXiv is just the tip of the iceberg. Crossref [Cro24], a publication database that is fed from publishers directly, includes 150M metadata records about research articles. Sci-Hub, a pirate Web site whose address frequently changes, includes around 88M papers as of 15th March 2024.

A fundamental challenge for researchers is keeping abreast of this ever-expanding body of literature, a task made daunting by the sheer volume of publications. This challenge sets the stage for the motivation behind the **TheoremKB** project.

¹<https://arxiv.org/>

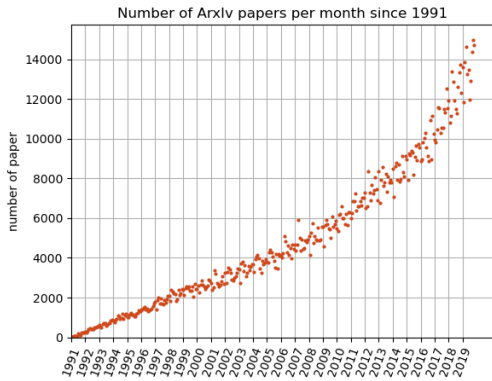


Figure 3.1: Total number of papers published on arXiv until 2019 [Del20]

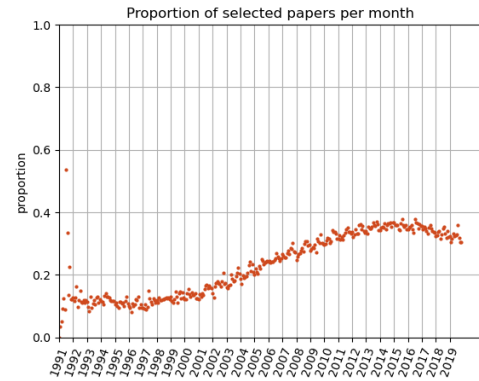


Figure 3.2: Papers with mathematical information such as Theorem, a Lemma or a Proposition [Del20]

TheoremKB, as the name implies, aims to establish an open knowledge base for proofs and theorems, specifically to create a comprehensive database of mathematical findings, drawing from a wide range of scientific papers in mathematics. This database will be enhanced with semantic details regarding the content and relationships among various results, using both automated processes and contributions from users. Users will have the ability to perform diverse and sophisticated searches within this knowledge base, which will deliver results quickly and provide information about the source and reliability of the data returned.

However, to appreciate the value of TheoremKB [Sen19], it is crucial first to recognise the shortcomings of the current systems. Presently, when an author seeks information on a specific proof or theorem, they typically turn to academic search engines like Google Scholar, AMiner, or MathSciNet. These platforms allow searches by keywords, author names, and venues and even permit navigation through the citation graph of papers, offering insights into related results. However, this approach needs to be revised. The fundamental issue is that these search engines, by design, prioritise papers as the basic unit of information, focusing on metadata that, while helpful for distinguishing between numerous relevant documents, does not directly represent the scientific results themselves. Consequently, researchers seeking specific scientific outcomes are often overwhelmed with papers that may not be directly relevant. This is because these search engines operate primarily at the level of linking papers rather than the mathematical results within them. The citation graph, thus, connects papers rather than the intrinsic scientific findings they contain.

TheoremKB seeks to address this issue by pivoting from the conventional practice of connecting paper to a more focused approach of linking mathematical results directly.

3.2 Why is TheoremKB Important?

Why is the creation of such a knowledge base important? To illustrate, let us explore three real-world scenarios that draw from experiences shared by my PhD supervisor, situations that many researchers in theoretical disciplines might find familiar:

- **Discovering Pre-existing Knowledge:** A colleague of my supervisor dedicated significant time to proving a particular result regarding the decidability of termination for linear TGDs in the core chase. Only after this investment of time and effort did he realise his findings were essentially a reiteration of conclusions drawn in an earlier work by Hernich in 2012 [Her12]. Unfortunately, his preliminary review of the literature did not catch this prior work, likely because it had not garnered much attention and had not specifically mentioned the core chase, a central concept in his research.
- **Correcting Publicised Errors:** Within the realm of probabilistic databases, a seminal work by Dalvi and Suciu in 2007 [DS07] has been praised for establishing a dichotomy for evaluating conjunctive queries – distinguishing between computationally intensive ($\#P$ -hard) and those that are not (PTIME). While widely cited and acknowledged, the original proof presented was flawed. This discrepancy was well-known among database theorists yet remained unaddressed in the publication for an extended period. Eventually, the authors rectified the proof and even expanded upon the

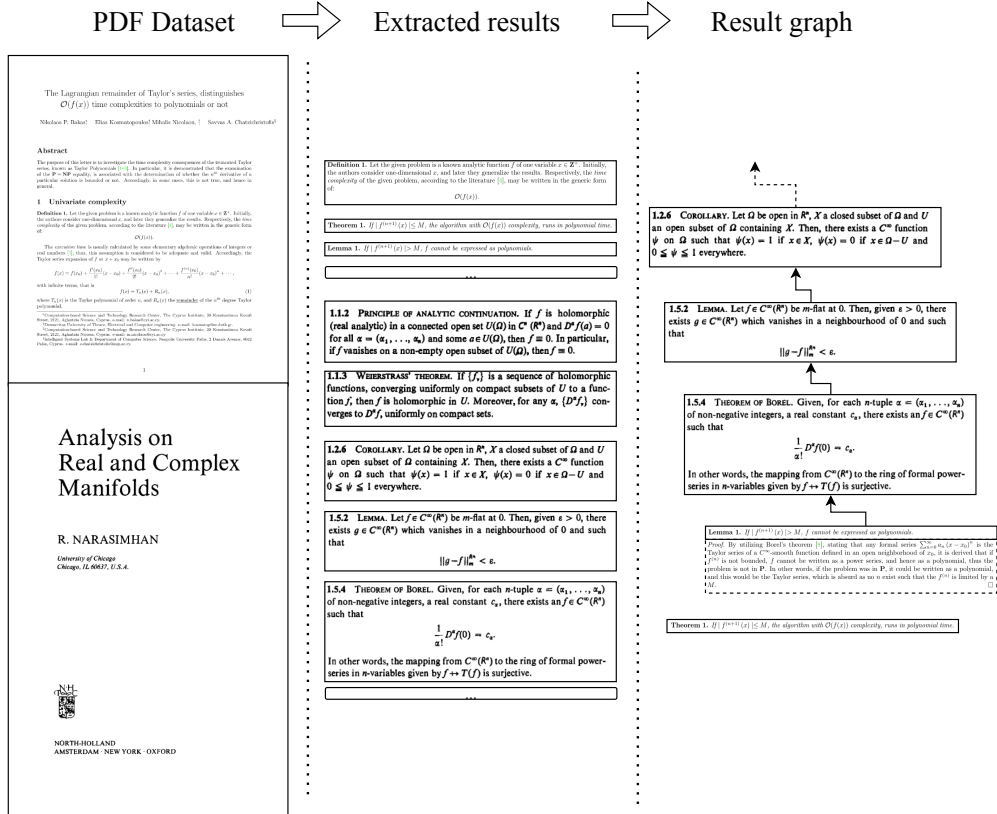


Figure 3.3: Two Step process for building a knowledge graph of papers: (i) extraction step; and (ii) linking step

initial findings in a subsequent publication in 2012. Nevertheless, with its flawed proof, the original document remains accessible, posing a potential source of confusion for those not privy to the correction.

- Managing Complex Dependencies:** In one of my supervisor’s submissions for publication [ABMS19], grappling with 80 enumerated elements – from definitions to theorems – presented a formidable challenge, not just for him but also for his fellow researchers. The complexity was amplified when proofs of some results depended on subsequent findings within the same work. Maintaining a clear, non-circular structure for our arguments was imperative yet strenuous, as also noted by an anonymous reviewer. They highlighted the problematic nature of forward references, which seemed to loop back in a circular logic – one theorem’s proof awaiting the establishment of another, which leaned on further lemmas, making it difficult to discern whether earlier theorems were being presupposed.

The root of these problems is the reliance on PDFs as the primary format for the scientific literature, which lacks an explicit structure for automation and interconnectivity, except through vague citation links. Unlike this static format, research articles inherently contain structured information like concepts, results, and proofs, which are the most accurate and valuable units for scientists.

TheoremKB envisions a transformation of scientific literature in mathematical domains from a collection of unstructured PDFs into a structured, openly accessible database of mathematical results. This database would not only link results to their proofs, definitions, and related findings but also to the original articles, without aiming for a fully formalised representation of each mathematical statement. Instead, TheoremKB strives to create a more usable and informative knowledge base that integrates textual content, structure, and semantic annotations.

This shift in paradigm would revolutionise how researchers search through literature, evaluate the impact of specific results, and analyse research articles, enabling them to address complex queries that are currently impossible to answer.

- What variants of the vertex cover problem are polynomial-time?

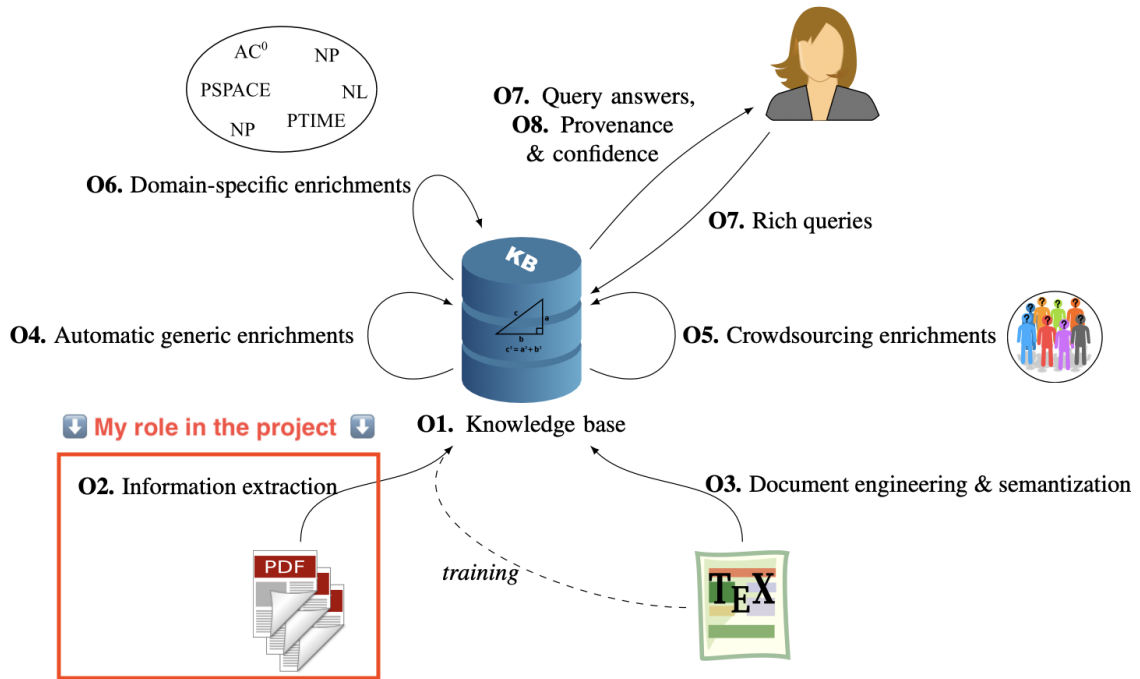


Figure 3.4: Schematic view of the TheoremKB architecture

- What does the graph of results used to prove other results given research article look like in a given research article?
- How many published results depend on a given theorem, and what would be the impact if this theorem were discovered wrong?

3.3 TheoremKB Proposal

The TheoremKB project encompasses a variety of components crucial to its overall success. This thesis will focus specifically on aspects related to my contribution: information extraction. To ensure a comprehensive understanding for the reader, I present in Figure 3.4 a diagram that illustrates these components, with further explanations below.

While my main focus will be on the second objective, which is **Information extraction**. I will later cover in great detail in Section 3.4 the specificities involved in Information extraction and what it means. However, for now, I will focus on the high-level objective. To understand each of the objectives well, I have provided a summary of tasks to be done in each objective:

1. Design and Development of a knowledge base architecture for storing information on mathematical findings, alongside creating a public web platform for accessing and querying this database.
2. **Development of methodologies and tools for extracting content, boundaries, and meta-information from mathematical results in PDF research articles for knowledge base integration, possibly using \LaTeX sources.**
3. Creation of strategies and tools for document engineering and semantization, extracting information from \LaTeX sources and providing authors with tools to generate semantically enriched data.
4. Automatic enhancement of the knowledge base with semantic information about content and connections between results, including categorising reasons for citing theorems and clarifying references.
5. Establishment of a community annotation method and platform allowing contributions to the knowledge base's content enhancement, ensuring a balance between contributor input and data quality.

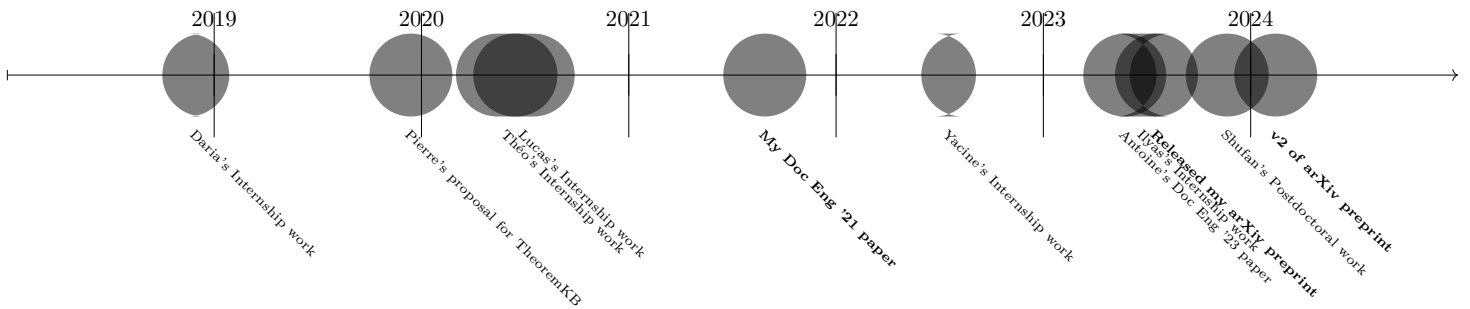
6. Development of specialised methodologies and tools for domain-specific semantic enrichments in the knowledge base, linking terms to ontologies of specific research areas.
7. Designing a rich query language for diverse informational needs from the knowledge base, assessing and implementing an efficient query evaluation system.
8. Formulation of a robust methodology for managing provenance and confidence metadata, ensuring efficient query response without compromising data integrity.

3.4 Information Extraction

3.4.1 Collaborative Research Insights by TheoremKB team

In the previous section, I have demonstrated the overall proposal and components within the TheoremKB project. To enhance comprehension of the chronology and evolution of the TheoremKB project, readers must grasp the precise timeline of developments. This timeline not only marks significant milestones but also highlights necessary updates and integrations of previous work. To clarify the significant advancements within TheoremKB, I will present a detailed timeline showcasing the projects that emerged within TheoremKB that directly covers the **Information Extraction** aspect of TheoremKB.

I am proud to have a direct or indirect connection to several of these milestones. Below, you will find a milestone chart that chronologically outlines important updates within the TheoremKB project:



Now, I will offer a concise introduction to all the projects conducted under the information extraction umbrella of the TheoremKB project.

3.4.2 Daria's Preliminary Work

Daria Pchelina was among the pioneering contributors to the TheoremKB project during her research project [Pch19], where she developed handcrafted features to identify the location of theorems and proofs, such as word count, average word length, italicisation, formula representation, heading detection, proof identification, boldness of the first word, and capitalisation of the first letter. These features were crucial for distinguishing between more straightforward class labels such as headings (i.e., the first line of theorems), body text (of theorems), post-body sections, and textual regions within PDF documents, providing a comprehensive overview of the document's structure. Her methodology relied on extracting features at the line level, which was then input into a Conditional Random Field (CRF) algorithm. This algorithm predicted labels for each line, effectively mapping out the document's layout.

An intriguing aspect of her work was the creation of a simple algorithm that identified body, text, headings, etc., based solely on the first word, followed by a sequence of italic characters. Surprisingly, this naive approach achieved an F_1 score of 0.94, see table 3.1, nearly matching the CRF method's performance, which scored 0.95. It is essential to highlight that despite the naive algorithm's comparable performance, the CRF approach offered the advantage of feature-level learning. This means it could assign weights and significance to various features, even if they were handcrafted over multiple lines, illustrating a deeper, more nuanced understanding of the document's structure.

While Daria's work laid a strong foundation for the TheoremKB project and showed promising results, it also presented significant challenges that complicated its direct application in the TheoremKB setting. Here are some of the problems encountered with her approach:

- **Using simpler class labels:** Daria employed class labels such as *body*, *heading*, and *text*, rather than directly addressing the *proof/theorem identification* task. These class labels are easier to identify because they rely heavily on font-level features. For instance, a heading can typically be

Region	Precision	Recall	F ₁ -score
Bayes Method			
after body	0.39	0.20	0.27
body	0.36	0.58	0.45
heading	0.39	0.92	0.55
text	0.97	0.94	0.95
avg/total	0.93	0.91	0.91
CRF Algorithm			
after body	0.48	0.30	0.37
body	0.68	0.45	0.54
heading	0.87	0.80	0.83
text	0.97	0.99	0.98
avg/total	0.95	0.95	0.95
Naive Algorithm			
after body	0.26	0.22	0.24
body	0.74	0.30	0.43
heading	0.71	0.91	0.80
text	0.96	0.99	0.97
avg/total	0.94	0.94	0.94

Table 3.1: Daria’s reported [Pch19] performance on a combined evaluation of Bayes, CRF, and Naive algorithms.

Weight	Tag	Feature
4.638508	heading	heading_first_word
2.194456	heading	boldness_first_word
1.868191	after-body	proof_first_word
0.802440	body	italics
0.484846	after-body	is_capital_first_letter
0.411067	heading	italics

Table 3.2: Daria’s reported [Pch19] font-level feature importance

recognised by Its boldness and low italics, a fact that is corroborated by the font-level feature importance presented in her report, see Table 3.2.

- **Operating on a much smaller dataset at the line level:** She labelled each line in a modestly sized dataset of only 3673 papers. Extractions were based on the PDFMiner² tool, with predictions and labels assigned to each line.
- **Using hard-coded, manually assigned features:** She utilised hard-coded features (refer to *font.txt*), manually labelling each font in the dataset as bold, italic, etc. This approach is labour-intensive and time-consuming and lacks scalability for fonts not included in the training corpus. Moreover, manual labelling of fonts is susceptible to human error, as the generated features are sensitive to labelling precision.

3.4.3 Théo’s Work on the Graph of Mathematical Results

Théo Delemazure’s report [Del20] presented an innovative approach to constructing a network of interconnected mathematical papers, leveraging theorems as pivotal links. His method hinged on the identification of bibliographical references within the original documents, assigning each paper an arXiv identifier, and thereby creating a graph where each node represents a scholarly work, see Figure 3.5.

²<https://pypi.org/project/pdfminer/>

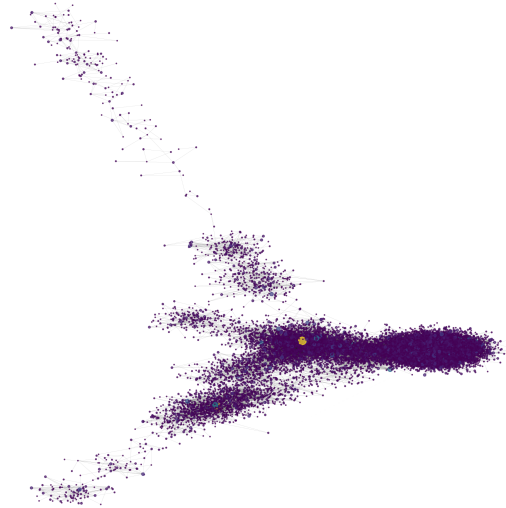


Figure 3.5: Representation of the biggest connected component (paper) of the aggregated graph using Networkx spectral layout [Del20]

Théo skillfully utilised \LaTeX sources to demarcate mathematical statements, utilising environments such as $\text{\begin{theorem}}$ and $\text{\end{theorem}}$ to establish hyperlinks. These links were then manifested in the PDF output, tagging theorem-like statements, including lemmas, propositions, and facts. Upon converting the PDF into an XML format using Grobid, he was able to parse both the results and their associated bibliographical references, the latter courtesy of Grobid’s preprocessing capabilities. The embedded links within the theorem labels clarified which references corresponded to which theorem-like element.

Furthermore, Théo employed the S2ORC dataset to trace the arXiv IDs associated with these citations. By matching the titles found in the theorem citations against the metadata in the dataset, including author names and paper titles, he could accurately link each theorem to its origins. This meticulous process resulted in a comprehensive graph that mapped the intricate relationships between academic papers.

A pivotal component of the methodology is the theorem-matching process, which employs a combination of textual similarity measures (TF-IDF) and deep learning models (autoencoders) to detect and link related mathematical results across different papers. This step is crucial for building a dense network of mathematical knowledge, where each node represents a unique paper, and edges signify a conceptual or direct citation link between results. By analysing the text surrounding mathematical statements, the algorithm assigns context and relevance to each connection, enriching the knowledge base with layers of semantic information, see Figure 3.6.

Théo’s work represents an initial endeavour to associate scientific findings with their respective publications, and it is essential to consider a few key points.

Firstly, **Théo’s methodology presumes the extraction process is flawless, operating under the assumption that \LaTeX sources are available at the time of inference.** His focus was on linking mathematical theorems to relevant literature, with less emphasis on the extraction process itself. Although it is theoretically feasible to integrate Théo’s technique with alternative extraction methods that work directly on PDFs, this could potentially lead to subpar performance due to extraction inaccuracies.

Secondly, Théo’s strategy involved utilising Grobid to identify the target paper. This was achieved by examining the bibliographic references extracted by Grobid and then matching the title from these references against the Semantic Scholar database to retrieve the corresponding arXiv identifier. While this approach successfully connects theorems to their target papers, **it does not pinpoint the specific sections within those papers.** Achieving this level of specificity would significantly enhance its utility by facilitating more precise references to the scientific results.

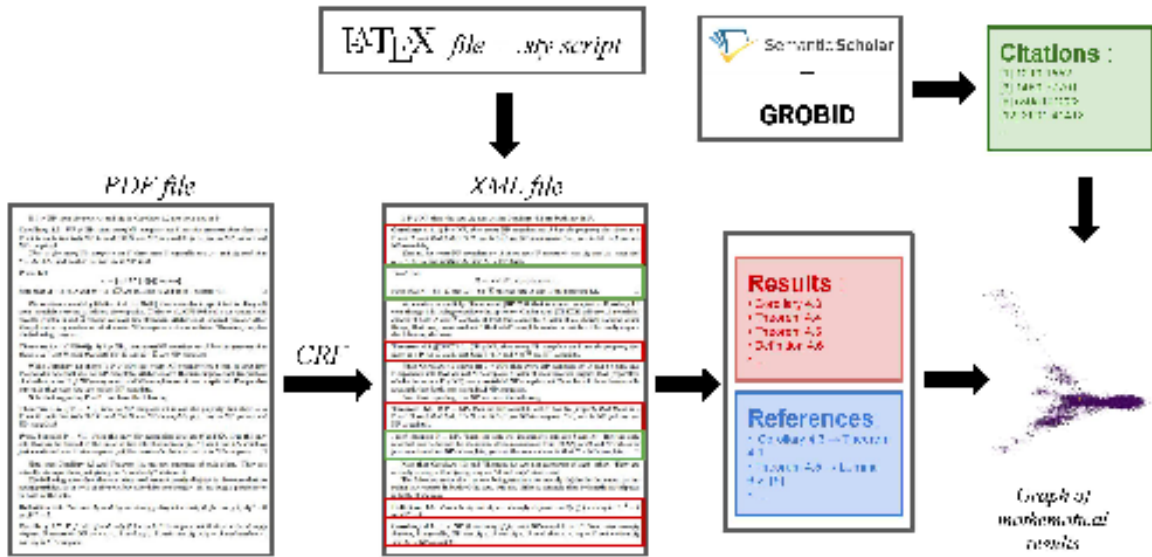


Figure 3.6: An overview of Théo’s approach

3.4.4 Lucas’s Work on Extraction through CRFs

Lucas Pluvineage’s project [Plu20] is dedicated to the automated extraction of theoretical statements, such as theorems, lemmas, and definitions, from scientific PDF articles. The methodology leverages a blend of handcrafted natural language processing (NLP) features and classical machine learning models, specifically Conditional Random Fields (CRFs). The project extracts geometrical features to understand the spatial arrangement of text, lexical features to identify key terms indicative of theoretical content (e.g., “proof”, “theorem”), and formatting features to detect the use of bold or italic fonts. These features are analysed at the line level, allowing for the precise extraction of complex theoretical statements from the structured and intricate layouts of scientific documents.

The classification challenge was defined across ten categories, including proofs, theorems, lemmas, definitions, remarks, and corollaries, achieving a macro F_1 score of around 0.72. While this represents a solid starting point, the approach encounters several issues. Firstly, it relies on the pdfalto tool to generate line-level text, which, when coupled with the linear nature of CRF models, may lead to a loss of contextual information across multiple lines in extended proofs. Additionally, the reliance on handcrafted features, such as specific keywords and formatting cues, limits the model’s ability to grasp the semantic meaning of the text. This approach narrowly focused on specific detection tasks, lacking the ability to transfer knowledge to related but slightly different tasks, a flexibility often found in deep learning approaches.

In essence, while Lucas’s project marks a significant step towards automating the extraction of critical scientific information, it also highlights the challenges inherent in such an endeavour. The reliance on handcrafted features and linear models underscores the need for more sophisticated, semantically aware approaches such as using a deep learning-based approach that not only significantly outperforms a rule-based approach. Thus, the classifier may be used for any other specific task. These insights point towards future directions for research, including the integration of deep learning techniques that could provide a more nuanced and adaptable solution to the complex problem of information extraction from academic documents.

3.4.5 Own Preliminary Work published at DocEng 2021

Our DocEng 2021 paper [MPS21], discussed in detail in Chapter 4 sets out with the objective of building upon and extending the foundational work initiated by Lucas, with a focus on exploring additional deep learning modalities (Vision, Text) that demonstrate relevance in identifying proofs and theorems. At its core, the project remains dedicated to extracting mathematical documents from PDF files. We proposed multiple innovative deep-learning approaches to enhance the efficiency and accuracy of this

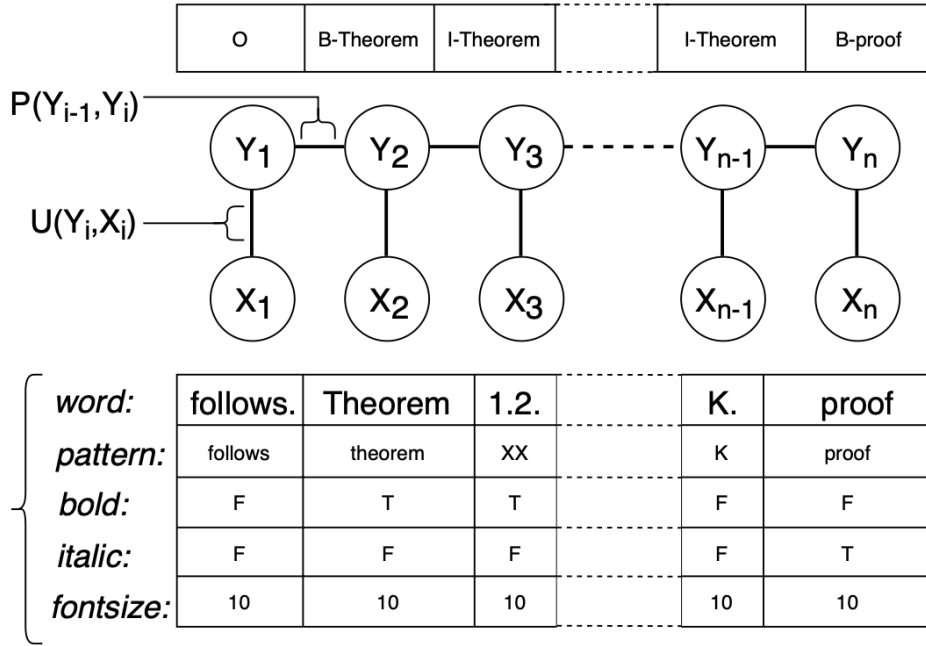


Figure 3.7: Visualisation of Lucas’s approach [Plu20] using handcrafted features applied to a linear-chain CRF.

extraction process, incorporating a variety of classifiers, each finetuned to a specific modality. For example, we refined language models to grasp the semantic intricacies of language structures beyond the rigid rule-based framework established by Lucas. Additionally, we delved into the potential of integrating other modalities subtly suggested in Lucas’s work, such as visual aesthetics—including the identification of bold and italicised text.

One novel approach we experimented with was the application of YOLOv4 [BWL20], a model typically reserved for object detection tasks to precisely locate the bounding boxes containing mathematical results when provided with a bitmap image of a page rather than relying on coordinates extracted by pdfalto. This exploration underscored the potential of visual features to contribute significantly to performance metrics.

Our research addressed several challenges noted in Lucas’s report, moving away from hardcoded features—such as assuming the first word in a sequence to be ‘proof’ or ‘theorem’—and instead leveraging the semantic comprehension capabilities of our models, which may prove beneficial for transfer learning applications. We categorised our findings into three classes: Proofs, Theorems (encompassing lemmas, propositions, definitions, etc.), and a Basic class (neither proof nor theorem), simplifying the categorisation process compared to the multiple subclasses under the theorem category in Lucas’s work. When writing this paper, we focused on building deep-learning approaches as an alternative to employing hard-coded/handcrafted features.

Our Vision-only approach classified objects within a bounding box and accurately identified the box’s coordinates, enhancing its real-world applicability.

While our exploration of language and vision approaches did not entirely replicate Lucas’s methodologies, it is crucial to acknowledge that his work benefited from the sequential information among lines—understanding a line’s label based on the preceding one’s label.

Even though the work lays the foundations for the deep learning approach, there are still major problems with the results:

- **Lack of Standardized Comparison Method:** There was no standardised method to compare detection scores across different approaches. Text-Language model, Vision-YOLOv4, Style-Lucas’s approach.
- **Varied Interpretation of Detection Tasks:** The interpretation of detection tasks varied within each modality; for instance, YOLOv4 focused on identifying bounding box coordinates alongside

proof: 0.36
 T with parameter $\epsilon = \delta/2$, which takes our instance \mathcal{F} and outputs unsighted instance \mathcal{G} . We can apply the α approximation algorithm on \mathcal{G} to obtain some assignment ζ , for which

$$\frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{G})} \geq \alpha.$$

Then, since $\text{Opt}(\mathcal{G}) \geq \gamma$, for the same assignment ζ we have

$$\frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{F})} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G}) + \epsilon} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G})} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G})} \geq \alpha - \frac{\epsilon}{\gamma} \geq \alpha - \delta.$$

which proves the statement of the theorem. \square

The argument from the previous theorem does not work for Min-CSPs, since in this case $\text{Opt}(\mathcal{G})$ can be arbitrarily small. Analogous claim for Min-CSPs was already proved in [16, Lemma 3.11] by using scaling techniques [14, 16]. For the sake of completeness, we give here

theorem: 0.37 *(Min-CSP A, and assume we can approximate the optimal value of A within factor α , a multiplicative factor α . Then, for every $\delta \in (0, 1)$, weighted version of the Min-CSP A can be approximated within a constant $\alpha - \delta$.)*

proof: 0.42 Consider version of CSP A, in which we ask whether there is an assignment $\lambda = (\lambda_1, \dots, \lambda_n)$ such that all the constraints of A are not satisfied. By Schaefer's dichotomy theorem [21], the problem of deciding whether there is an assignment which fulfills all the constraints is either NP-hard or in P. If solving this problem is NP-hard, then both weighted and unweighted versions of Min-CSP A are obviously NP-hard to approximate within any constant. To the contrary, if the problem is in P, then we can check in polynomial time whether $\text{Opt}(\mathcal{F}) = 0$. In case $\text{Opt}(\mathcal{F}) = 0$, we have found an optimal assignment, so it only remains to consider $\text{Opt}(\mathcal{F}) > 0$.

proof: 0.43 Without loss of generality let us assume that the weights of constraints $\{w_i\}_{i=1}^m$ are sorted in descending order, i.e. $w_1 \geq w_2 \geq \dots \geq w_m$. We can find in polynomial time the largest $k \geq 1$ such that there is an assignment fulfilling constraints C_1, C_2, \dots, C_k .

For finally chosen k at least one of C_1, \dots, C_k will be true in any assignment, so we have that $\text{Opt}(\mathcal{F}) \geq w_k$. Also, since there is an assignment satisfying the rest $m - k$ constraints, we have that $\text{Opt}(\mathcal{F}) \leq \sum_{i=k+1}^m w_i$.

proof: 0.31 Let us partition the constraints C_i into the following three groups:

- light: constraints C_i with weight $w_i \leq w_k/m^2$,
- medium: constraints C_i with weight $w_k/m^2 < w_i < w_k/m$,
- heavy: constraints C_i with weight $w_i \geq w_k/m$.

proof: 0.34 Now, \mathcal{F}' by adding medium and heavy constraints C_i from \mathcal{F} . Furthermore, we scale down the weights of heavy constraints to w_k/m^2 in \mathcal{F}' . Finally, we normalize the weights to total weight by multiplying them by some factor $\sigma > 1$. Note that $\text{Opt}(\mathcal{F}') \geq w_k \sigma$, since \mathcal{F}' still has (although with different weights) constraints C_1, C_2, \dots, C_k . Hence, in a completely analogous manner to Theorem 8, we can use the algorithm from Lemma 7 with $\epsilon = \frac{\delta w_k \sigma}{2}$ to get an assignment ζ which gives us an $\alpha + \delta/2$ approximation of the optimal value for \mathcal{F}' . Finding the $\alpha + \delta/2$ approximation can be performed in polynomial time, because $\text{opt}(\mathcal{F}') \geq w_k \sigma$. Let us now see how well ζ approximates the optimal value of \mathcal{F} . We have that following two properties:

proof: 0.60
 T with parameter $\epsilon = \delta/2$, which takes our instance \mathcal{F} and outputs unsighted instance \mathcal{G} . We can apply the α approximation algorithm on \mathcal{G} to obtain some assignment ζ , for which

$$\frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{G})} \geq \alpha.$$

Then, since $\text{Opt}(\mathcal{G}) \geq \gamma$, for the same assignment ζ we have

$$\frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{F})} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G}) + \epsilon} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G})} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G})} \geq \alpha - \frac{\epsilon}{\gamma} \geq \alpha - \delta.$$

which proves the statement of the theorem. \square

The argument from the previous theorem does not work for Min-CSPs, since in this case $\text{Opt}(\mathcal{G})$ can be arbitrarily small. Analogous claim for Min-CSPs was already proved in [16, Lemma 3.11] by using scaling techniques [14, 16]. For the sake of completeness, we give here somewhat more

theorem: 0.92 *(Min-CSP A, and assume we can approximate the optimal value of A within factor α , a multiplicative factor α . Then, for every $\delta \in (0, 1)$, weighted version of the Min-CSP A can be approximated within a constant $\alpha - \delta$.)*

proof: 0.52 Consider version of CSP A, in which we ask whether there is an assignment λ to the variables such that all the constraints of A are not satisfied. By Schaefer's dichotomy theorem [21], the problem of deciding whether there is an assignment which fulfills all the constraints is either NP-hard or in P. If solving this problem is NP-hard, then both weighted and unweighted versions of Min-CSP A are obviously NP-hard to approximate within any constant. Therefore, without loss of generality, we assume that deciding whether all constraints can be fulfilled is in P for A.

Hence, given an instance \mathcal{F} of the Min-CSP A, we can check in polynomial time whether $\text{Opt}(\mathcal{F}) = 0$. In case $\text{Opt}(\mathcal{F}) = 0$, we have found an optimal assignment, so it only remains to consider $\text{Opt}(\mathcal{F}) > 0$.

Without loss of generality let us assume that the weights of constraints $\{w_i\}_{i=1}^m$ are sorted in descending order, i.e. $w_1 \geq w_2 \geq \dots \geq w_m$. We can find in polynomial time the largest $k \geq 1$ such that there is an assignment fulfilling constraints C_1, C_2, \dots, C_k .

For finally chosen k at least one of C_1, \dots, C_k will be true in any assignment, so we have that $\text{Opt}(\mathcal{F}) \geq w_k$. Also, since there is an assignment fulfilling the first $k - 1$ constraints, we have that $\text{Opt}(\mathcal{F}) \leq \sum_{i=k}^m w_i$.

Let us partition the constraints C_i into the following three groups:

- light: constraints C_i with weight $w_i \leq w_k/m^2$,
- medium: constraints C_i with weight $w_k/m^2 < w_i < w_k/m$,
- heavy: constraints C_i with weight $w_i \geq w_k/m$.

Then, we create an instance \mathcal{F}' by adding medium and heavy constraints C_i from \mathcal{F} . Furthermore, we scale down the weights of heavy constraints to w_k/m^2 in \mathcal{F}' . Finally, we normalize the weights to total weight by multiplying them by some factor $\sigma > 1$. Note that $\text{Opt}(\mathcal{F}') \geq w_k \sigma$, since \mathcal{F}' still has (although with different weights) constraints C_1, C_2, \dots, C_k . Hence, in a completely analogous manner to Theorem 8, we can use the algorithm from Lemma 7 with $\epsilon = \frac{\delta w_k \sigma}{2}$ to get an assignment ζ which gives us an $\alpha + \delta/2$ approximation of the optimal value for \mathcal{F}' . Finding the $\alpha + \delta/2$ approximation can be performed in polynomial time, because $\text{opt}(\mathcal{F}') \geq w_k \sigma$. Let us now see how well ζ approximates the optimal value of \mathcal{F} . We have that following two properties:

Figure 3.8: Application of YOLOv4 for Precision Detection of Individual Text Lines: This image demonstrates the YOLOv4 model's capability to accurately detect and highlight individual lines of text within a document, showcasing its effectiveness in isolating distinct textual elements.

Figure 3.9: YOLOv4's Adaptation to Paragraph Detection: Illustrated here is the YOLOv4 model's performance when trained to identify paragraphs by merging adjacent text lines into a single bounding box, highlighting its versatility and adaptability in recognising larger textual structures.

object classification, whereas Language and Style models were centred around classification tasks using coordinates derived from PDF files at the line level (using the pdfalto), thus making the visual task significantly harder.

- **Dependence on Sequential Information:** The Style approach (Lucas’s method) relied on CRF to benefit from line-to-line information and hardcoded rules, a feature absent in the Language and Vision models, which did not consider information across previous lines.
- **Comparatively Poor Detection Performance:** The performance of the detection task was still quite poor compared to a rule-based hardcoded model.

In conclusion, this paper serves as a proof of concept for transitioning from a set of rigid, hard-coded rules, such as those employed by Lucas, to a more flexible and generalisable deep learning model capable of separately processing textual and visual data. This shift not only diversifies the methodologies available for information extraction but also opens the door to more adaptive and nuanced approaches to understanding and extracting mathematical documents.

3.4.6 Yacine’s Work on Connecting Theorems Across Papers

I had the opportunity to work closely with Yacine Brihmouche, building upon Théo’s foundational efforts in connecting mathematical results. During this time, I developed a script for rendering information in CSV format at the paragraph level using GROBID, diverging from the line-level analysis performed with pdfalto. The script generated a CSV file containing paragraphs semantically parsed by GROBID. Text is enclosed in the bounding box, and the font information and the bounding box information are included.

Yacine’s approach [Bri22] aimed at linking specific mathematical results, provided there is a link between the two papers, see Figure 3.10. The work extends to Théo’s work, where he tries to find the target paper for the source mathematical result.

Due to manual examination and construction of the dataset by the manual labelling from papers, Yacine managed to build 45 connected pairs (source and target Result) and evaluate their relevance based on the cosine similarity of features extracted at the paragraph level.

Yacine employed two straightforward methods:

- A TF-IDF word vectoriser that constructs feature vectors for paragraphs based on the characters in the theorem.
- A DistilBERT-based [SDCW19] [CLS] representation for the mathematical result paragraph.

I provided Yacine with the finetuned version of the DistilBERT model on a dataset of mathematical results categorised into proofs, theorems, and basics, following the structure of the language model discussed in the DocEng ’21 paper [MPS21], but this time, it was trained on paragraph-level features generated from GROBID.

The effectiveness of these methods was assessed through their cosine similarity, with only the closest match (K@1) being considered a potential link. **The TF-IDF vectoriser successfully matched 17 out of 45 pairs, while DistilBERT achieved only 7 out of 45 matches.** Several factors could account for these outcomes:

1. Often, a referenced result includes identical keywords or tokens, making TF-IDF a suitable method for feature generation due to its emphasis on word usage frequency rather than the semantic meaning of words.
2. The DistilBERT model, initially trained for the Passage Classification (Grobid) task similar to the Line classification (pdfalto) task described in the DocEng ’21 paper, was not specifically adapted for the theorem matching task. To be precise, instead of utilising a finetuning step (typical for many NLP tasks) such as training on dedicated model heads or employing a Siamese network [CHL05] (to capture similarity) that allows training for additional task-specific weights for the specific theorem matching task could drastically improve the performance. Yacine relied only on raw features from [CLS] token employed for cosine similarity analysis. Task-specific finetuning could have significantly improved the performance of the Language model.
3. Given DistilBERT’s fixed vocabulary and training on general datasets like Google News, it may overlook specific scientific terms, categorising them as unknown tokens ([UNK]). This leads to a more ambiguous representation. Conversely, the TF-IDF vectoriser, trained on the same scientific corpus, ensures no out-of-vocabulary (OOV) tokens are present.

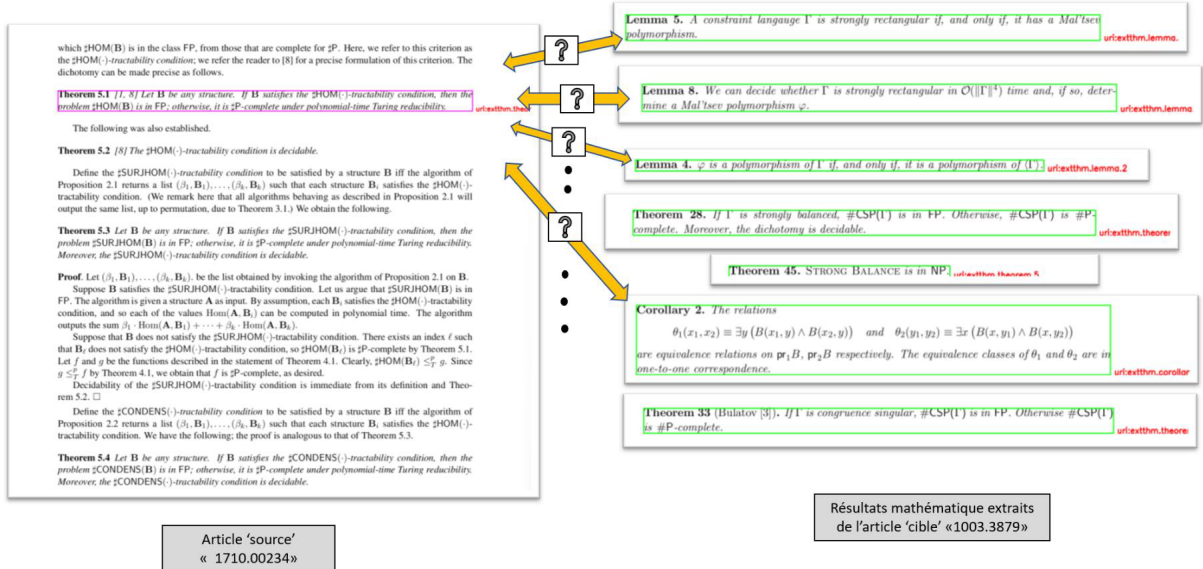


Figure 3.10: Visualising Theorem Connectivity problem [Bri22] explored during Yacine’s Internship

My contribution to Yacine’s report underscored several pivotal advancements beyond the high-level summary:

1. I developed a preprocessing script that generated a CSV file (using Grobid this time rather than pdfalto in the DocEng ’21 paper), capturing both visual and textual features, as well as font-level characteristics extracted by pdfalto. This laid the groundwork for comparing and benchmarking models trained on various modalities. During Yacine’s Internship, he utilised my script on textual features only ³.
2. I also provided a finetuned DistilBERT model to Yacine, Where we successfully employed the same frozen language model (trained on extraction task) in a transfer learning approach (using CLS token) to identify matches in theorems, even when the training task differed and the performance was poor. This underscores the significance and shift towards a fully deep learning-based methodology.
3. I created a visualisation tool using OpenCV, which significantly aided Yacine in the annotation task serving as a critical component to acquire manually labelled Ground truths.

3.4.7 Antoine’s work on Document Class Inference

Antoine Gauquier’s research [GS23] focuses on developing a method to determine the type of scientific papers automatically. This work is essential because it can help find out which papers are freely available and make it easier to pull out information from them. The study uses two main techniques to achieve its goal.

Firstly, the research looks into using specific characteristics of papers, like the size of margins, the width of text columns, and the types of fonts used. The idea is that different types of scientific papers have different ways of formatting, and by looking at these formatting features, one can guess the paper’s type. To do this, Antoine turns the content of PDF files into a more straightforward XML format, making it easier to analyse these features. Then, by examining how these features appear across various documents, Antoine uses a Random Forest classifier to predict the paper type. This approach is logical because it assumes that the way a document looks can tell us something about what kind of document it is.

The second technique involves a more visual approach. Here, Antoine converts the first page of each document into an image. The belief is that the first page of a paper contains enough visual clues to identify the document’s class. With these images, Antoine then uses a type of artificial intelligence known as

³It could have been completely possible to test Yacine’s approach on a different modality, if the paragraph level features were available for that modality, for example- it would have been possible to also compare the visual modality by having the visual features extracted from a CNN model and then benchmark it against the text modality even though we do not employ it in his Internship

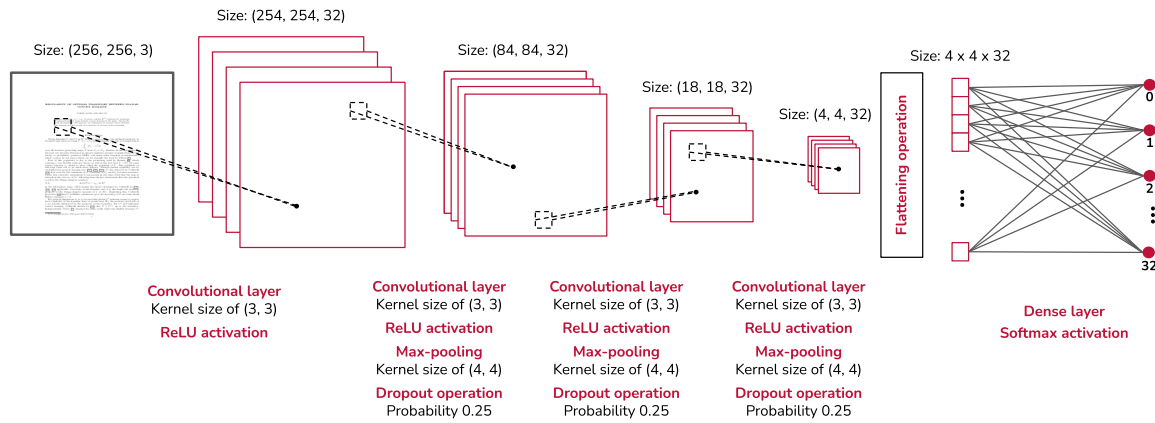


Figure 3.11: Proposed Diagram of Antoine’s [GS23] CNN architecture predicting one of the 33 popular documentclass

Convolutional Neural Networks (CNNs) will classify the documents, see Figure 3.11. CNNs are very good at recognising image patterns, making them suitable for this task. This method is interesting because it is like teaching a computer to recognise the type of paper by looking at its picture, much like a person might.

Both methods have their unique processes for preparing the data. For the first method, converting PDFs to XML allows for extracting specific layout features. For the second method, converting the first page of PDFs into images allows for applying computer vision techniques. In both cases, the goal is to train a computer model to predict the document class based on the extracted features or visual patterns.

The research found that both methods have their merits. The first method, focusing on the document’s layout features, showed promise, especially when using machine learning algorithms to sift through these features and make predictions. The second method, which uses computer vision, also proved effective, demonstrating that visual patterns on a document’s first page can help classify it.

In conclusion, Antoine’s work presents innovative approaches to automatically classifying scientific documents. By combining traditional data analysis with modern artificial intelligence techniques, this research opens up new possibilities for managing and understanding the vast amounts of scientific literature available today. The proposed methods are theoretical and have been shown to work well in practice, offering valuable tools for researchers and libraries alike. This research stands out for its practical applications, offering a step forward in how we handle and access scientific knowledge.

3.4.8 Own Work on Multimodal Machine Learning

Our preprint [MGS23], detailed in Chapter 5, is an extension of earlier work published in the DocEng ’21 conference [MPS21], still focusing on the extraction of proofs and theorems from scholarly articles. The objective of this research is to automate the process of identifying theorem-like environments and proofs within the complex formats of PDF documents, leveraging a combination of text, font features, and bitmap image renderings as distinct modalities. Unlike the work proposed in the Doceng paper, this version aims to unify several modalities, including a sequential approach (taking into account the label of the previous block).

The paper delineates three unimodal models – focusing on text, vision, and font modalities – before combining them into a comprehensive multimodal approach [ASMyGG20] (Gated Multimodal Units). For the text modality, we pre-train a new language model on a scientific corpus, demonstrating its effectiveness in capturing the unique vocabulary and semantic structures of mathematical writing. The vision modality employs an EfficientNetv2 [TL21] architecture to classify bitmap renderings of PDF blocks, focusing on visual cues that indicate theorem and proof environments. The font modality uses an LSTM [HS97] trained on sequences of font names and sizes within document blocks, exploiting typographical features to aid classification.

The paper’s innovative core lies in its multimodal and sequential models. It introduces a late fusion multimodal model that integrates features extracted by the unimodal classifiers, taking into account for the sequential succession of document blocks to improve classification accuracy. Additionally, a Conditional Random Field (CRF)–based approach further refines this model by leveraging the sequential nature of

Table 3.3: Performance comparison of modality and multimodal models [MGS23], with and without CRF; batches indicate training size.

Modality	Model	CRF	Batches	Acc. (%)	Mean F ₁ (%)
Dummy	—	—	—	59.41	24.85
Font	LSTM 128 cells	No	11	65.00	45.00
		Yes	11+1	52.20	50.49
Vision	EfficientNetV2m_avg	No	9	69.43	60.33
		Yes	9+1	74.13	69.82
Text	Pretrained RoBERTa-like	No	20	76.45	72.33
		Yes	20+1	82.70	80.52
Multimodal	GMU	No	10	76.86	73.87
		Yes	10+4	84.38	83.01

paragraphs and the relationships between adjacent blocks, embodying a more nuanced understanding of document structure, see table 3.3.

This work introduces significant enhancements over the initial approach (DocEng) by employing:

- Enhanced Text Processing:** The study introduces a pivotal improvement in the processing and analysis of textual data. Unlike the initial approach that relied on extracting text lines using pdfalto, this extension utilises paragraph-level features extracted by Grobid. This methodology is instrumental in preserving the integrity and completeness of textual information, addressing the limitation where a single line of text might not fully encapsulate a sentence’s meaning or context.
- Uniform Comparison Framework:** A significant advancement in this research is establishing a uniform framework for comparing different modalities. This framework enables a more systematic and equitable assessment across various models engaged in the classification task. Such a structured comparison method provides clearer insights into the effectiveness of each model, highlighting their strengths and areas for improvement more transparently.
- Multimodal and Sequential Approach:** The paper marks a substantial leap forward with the introduction of a multimodal approach, leveraging a Gated Multimodal Unit (GMU) [ASMyGG20] combined with a sequential analysis of unimodal features. This innovative strategy showcases strong performance gains, demonstrating the superiority of combining multiple modalities over-relying on a singular approach. This multimodal and sequential method significantly enhances the model’s ability to classify and extract proofs and theorems from scholarly texts accurately.
- Versatile Pretrained Language Model:** Another noteworthy contribution of this study is developing a versatile pre-trained language model. This model is not confined to the specific task of extracting theorems and proofs but can also be adapted for various other language learning tasks. The flexibility and adaptability of the pre-trained model presents a valuable asset for researchers and practitioners across different domains of natural language processing and document analysis, broadening the scope of its applicability and utility.

3.4.9 Ilyas’s work on Embedding Equation Structures

The project, led by Ilyas Lebleu and titled “Embedding Equation Structure for Theorem Matching,” sets out with the ambitious goal of deepening our understanding of mathematical theorems’ interconnections. It aims to weave a more intricate web of knowledge by integrating equations into the fabric of existing databases, thus enriching the dialogue between disparate pieces of scholarly literature. The project continues the TheoremKB initiative, which seeks to chart the relationships between theorems in a directed graph format, where each theorem is connected based on its foundational role in proving subsequent theorems.

The methodology employed to achieve this goal begins with the challenge of detecting and interpreting equations within scholarly articles. Given that most research articles are compiled into PDF format without direct access to their L^AT_EX source, the project devised a novel approach to identify equations

Definition 2.1. Let V be a vector space over the field F . We say that the collection σ of subspaces is a spread if (1) $A, B \in \sigma$, $A \neq B$ then $V = A \oplus B$, and (2) every nonzero vector $x \in V$ lies in a unique member of σ . The members of σ are the components of the spread.

```

\begin{definition}
Let  $V$  be a vector space over the field  $F$ . We say that the
collection  $\sigma$  of subspaces is a \emph{spread} if (1)  $A, B$ 
 $\in \sigma$ ,  $A \neq B$  then  $V = A \oplus B$ , and (2) every nonzero
vector  $x \in V$  lies in a unique member of  $\sigma$ . The members
of  $\sigma$  are the \emph{components} of the spread.
\end{definition}

```

Figure 3.12: Rendering of a definition from a mathematical scholarly article (Nagy, 2013) accompanied with its \LaTeX source code. The definienda are "spread" and "components".

through their compiled form. This involved creating a training dataset from arXiv submissions, including \LaTeX and PDF versions of documents, enabling the model to learn the positioning of equations within the PDF structure. Key to this process was understanding the inner workings of PDF documents and leveraging the structure of \LaTeX to embed links around equations, facilitating their detection.

Following the detection of equations, the project explores two principal approaches to embedding these equations into a format conducive to comparison and analysis. The first employs direct computer vision techniques to recognise equations based on their visual similarity. This approach is supported by Siamese networks and triplet loss [SJ03] methodologies, which train the model to differentiate and closely associate equations appearing in similar contexts. The second approach, deemed more structural, utilises the intrinsic tree-like structure of mathematical expressions encoded through a specialised neural network architecture that respects the hierarchical nature of mathematical operations.

Despite the promising advances made by the project, it faces inherent limitations. The complexity of accurately detecting and interpreting equations from PDFs, the variability in the presentation of mathematical formulas, and the challenge of adequately capturing the nuanced similarities between different equations are significant hurdles. Moreover, the project's scope is naturally limited by the available data and the chosen methodologies' ability to generalise across the diverse landscape of mathematical literature.

In conclusion, the "Embedding Equation Structure for Theorem Matching" project represents a significant stride towards integrating mathematical equations into the broader context of scholarly communication. By pioneering methods to detect, interpret, and compare equations, it lays the groundwork for more sophisticated analysis of the interconnectedness of mathematical theorems. However, the journey is far from complete. Future work will need to address the outlined limitations, refine the methodologies, and continue pushing the boundaries of what is possible in digital scholarly analysis. The project stands as a testament to the potential of combining classical computer vision techniques with advanced machine learning models to unravel the complex web of mathematical knowledge.

3.4.10 Shufan's work on Extracting Definienda

In the pursuit of enhancing the accessibility and understanding of mathematical literature, this project, spearheaded by Shufan Jiang, sets its sights on the automated extraction of defined terms, known as definienda, from mathematical definitions within scholarly articles [JS23] (see Figure 3.12). The initiative stems from recognising these definitions' critical role in the broader landscape of scientific discovery and knowledge management. By focusing on mathematical texts, the project addresses a unique challenge: The intricate blend of language and mathematical notation.

The approach to this problem is two-pronged. Firstly, a dedicated dataset was crafted from the \LaTeX source of papers in the arXiv's Combinatorics category. This dataset creation involved innovative rule-based methods to identify and extract definienda by exploiting standard formatting practices used by mathematicians in \LaTeX documents. This step was crucial, given the absence of existing datasets suited to the peculiarities of mathematical literature. Secondly, the project harnessed the power of transformer-based models, specifically finetuning pre-trained language models for token classification and employing generative pre-trained transformers (giving prompts to GPT-3.5 and 4) alongside comparing standard Roberta [LOG⁺19] and Pretrained Roberta (provided by me) with token classification head. These models were trained and queried to pinpoint the definienda within the text blocks of mathematical

definitions.

The evaluation of these methodologies revealed promising results. The finetuned transformers demonstrated high precision and recall, effectively identifying the definienda in various contexts. Notably, the experiments with GPT models also showed considerable success, albeit with differences in performance attributed to the models’ capabilities and computational costs. The comparison between the finetuned models and GPT’s answers provided insights into the strengths and limitations of each approach, highlighting the balance between accuracy and resource expenditure.

I contributed by developing an initial model (RoBERTa-like), which Shufan used, and training it from scratch on a collection of scientific texts (11GB/196K papers). This model was designed to identify parts of text without context as it was trained on masked language modelling loss. Shufan refined and evaluated it against the standard Roberta model, which had been trained with 160GB of data. While my model did not perform better than the standard Roberta model on her task, it is essential to recognise the difference in the data used for training the Roberta model. I am pleased that my model was used for comparison in her study, as this encouraged the exchange of ideas and methods we initially aimed for in 2021. This paper shows how a model designed for one task (extraction) can adapt to work on related tasks (Task-specific finetuning). Transfer Learning is possible only for models that capture semantic knowledge, unlike Lucas’s hard-coded method, which is based on a fixed set of rules. Furthermore, Shufan’s work could eventually allow for searching specific terms within the knowledge base once fully built, furthering the overarching goals of the TheoremKB project.

In conclusion, this project represents a significant step forward in the automated processing of mathematical texts. Developing and evaluating models for definienda extraction lays the groundwork for future advancements in mathematical knowledge management.

3.4.11 Own Work on Modular Multimodal Extraction

Our study “Modular Multimodal Machine Learning for Extraction of Theorems and Proofs in the Scientific Literature”, detailed in Chapters 6 and 7, aims to automate the extraction of mathematical statements, such as theorems and proofs from scholarly articles.

The methodology employs a comprehensive approach combining different modalities - Text, vision, and font features - to identify relevant mathematical statements within PDF documents accurately. Initially, a dataset comprising around 200 000 English-language papers from arXiv were curated to develop and evaluate the proposed models.

Three distinct unimodal models were developed to separately analyse text (using a pre-trained Roberta-based [LOG⁺19] language model), visual layouts (using an EfficientNetv2 [TL21] CNN), and font features (using an LSTM [HS97] model). Subsequently, these models were integrated into a singular multimodal system through a late fusion approach, aiming to leverage the complementary strengths of each modality. Various fusion approaches were tested in this section instead of just one, which was GMU in the previous paper [MGS23].

Furthermore, Sequential modelling techniques (based on deep learning) were applied to enhance the identification of theorem-like statements and proofs. These techniques utilise transformers/conditional random fields to incorporate sequential information across paragraphs, thus capturing document narrative flow.

This work builds closely on its predecessor, enhancing each segment’s modularity to facilitate easier component substitution and integration. Specifically, the updates introduced in this version include:

Expanded Multimodal Fusion Comparisons: Beyond employing Gated Multimodal Units (GMU)[ASMyGG20], this iteration also evaluates a variety of fusion-based approaches. These include Multimodal Fusion, EmbraceNet [CL19], Bilinear Gated Mechanism [KGJM18], and Cross-Modal Attention Mechanism [LBPL19], designed to capture relationships between different modalities more effectively.

Expanded Sequential Approach Comparisons: While the previous version focused solely on a Conditional Random Field (CRF) approach [LMP01], the current model broadens this scope. Alongside CRF, we now assess two additional deep learning-based methods: the Sliding Window approach and a Hierarchical Transformer-based approach. Each offers bidirectional context analysis—one concentrating on localised windows of features and the other encompassing the entire document.

Adopting a modular design across all aspects of our study is a notable enhancement in our methodology. This flexibility allows for the interchange of components within the vision, language, font sequence, multimodal, and sequential analysis backbone. Specifically, we transition from a Generalized Multimodal

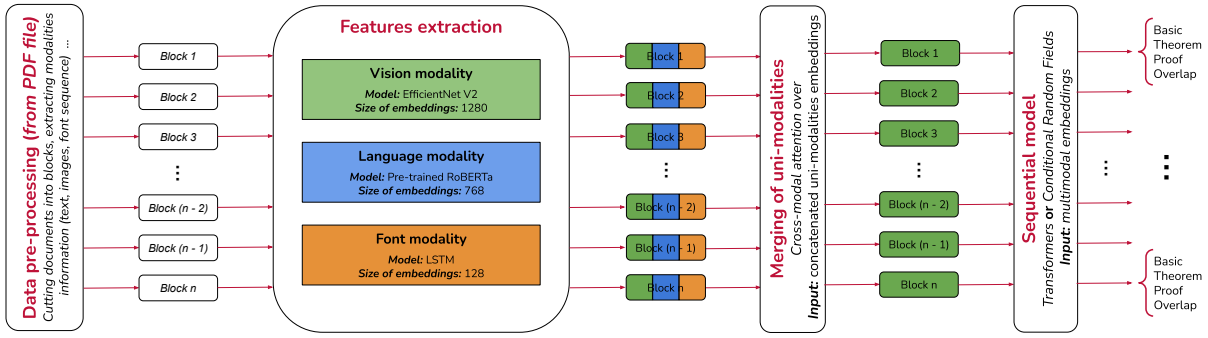


Figure 3.13: Overall model Inference pipeline

Unit (GMU) combined with a Conditional Random Field (CRF) methodology to a strategy that leverages Cross-Modal Interaction with a Sliding Window (SW) transformer. This modification alone accounts for a significant improvement in our results, elevating the model’s accuracy from 84% to 88%. Additionally, we explore the utility of Hierarchical Attention Transformers (HATs) [CDF⁺22] are used to capture long-term dependencies within the data. Despite these efforts, introducing HATs did not yield the anticipated enhancements in performance. Ultimately, the culmination of our research led to the decision to replace the CRF component entirely with an SW transformer in our final approach, integrating several deep-learning models to refine our analytical capabilities further.

Despite these promising outcomes, the study acknowledges several limitations. The complexity of the task means that achieving perfect accuracy is challenging, as even human experts may struggle to distinguish between different mathematical environments within scholarly texts. Additionally, the evaluation was limited to English-language articles, leaving the adaptability of the approach to documents in other languages or formats untested. The reliance on L^AT_EX sources for ground-truth dataset construction may not fully represent the diversity of document formats in the broader scientific literature.

On top of the performance gains observed by the study, several other aspects address the efficiency and practicality of our approach as opposed to many other Document AI models, such as :

- Our approach operates directly on raw PDFs that are multi-page scientific papers, tackling documents’ scientific nature.
- Thanks to Grobid, our approach does not require costly OCR preprocessing or L^AT_EX sources at inference.
- Shows progressive gains without applying any custom pretraining losses to capture cross-modality relationships, making it simple yet effective.

In conclusion, the study presents a novel approach to document AI, offering a modular multimodal machine learning framework for extracting mathematical statements from scholarly literature.

3.5 How is the Extraction Task Linked to other Tasks?

The TheoremKB project is dedicated to constructing a comprehensive knowledge graph that links mathematical findings. To achieve this, we employ a dual-phase approach:

1. The first phase involves deploying a machine-learning model designed to sift through PDF articles and extract theorems and proofs. These elements form the nodes of our knowledge graph.
2. The second phase involves a function capable of evaluating the degree of relatedness between two mathematical results, thus creating the graph’s interconnected edges.

Daria, Lucas, and I focused on the extraction phase during the project. My latest model, which utilises a Sliding Window Transformer, represents an advancement over previous methods. It analyses text at the paragraph level, allowing multimodal context to flow in both directions, a significant improvement over the Conditional Random Fields (CRFs) approach, which only permits unidirectional flow.

Extracting theorems and proofs is pivotal, laying the foundation for linking related findings. Théo’s report depended on L^AT_EX sources for ground truth because Lucas’s work was incomplete at the time. In

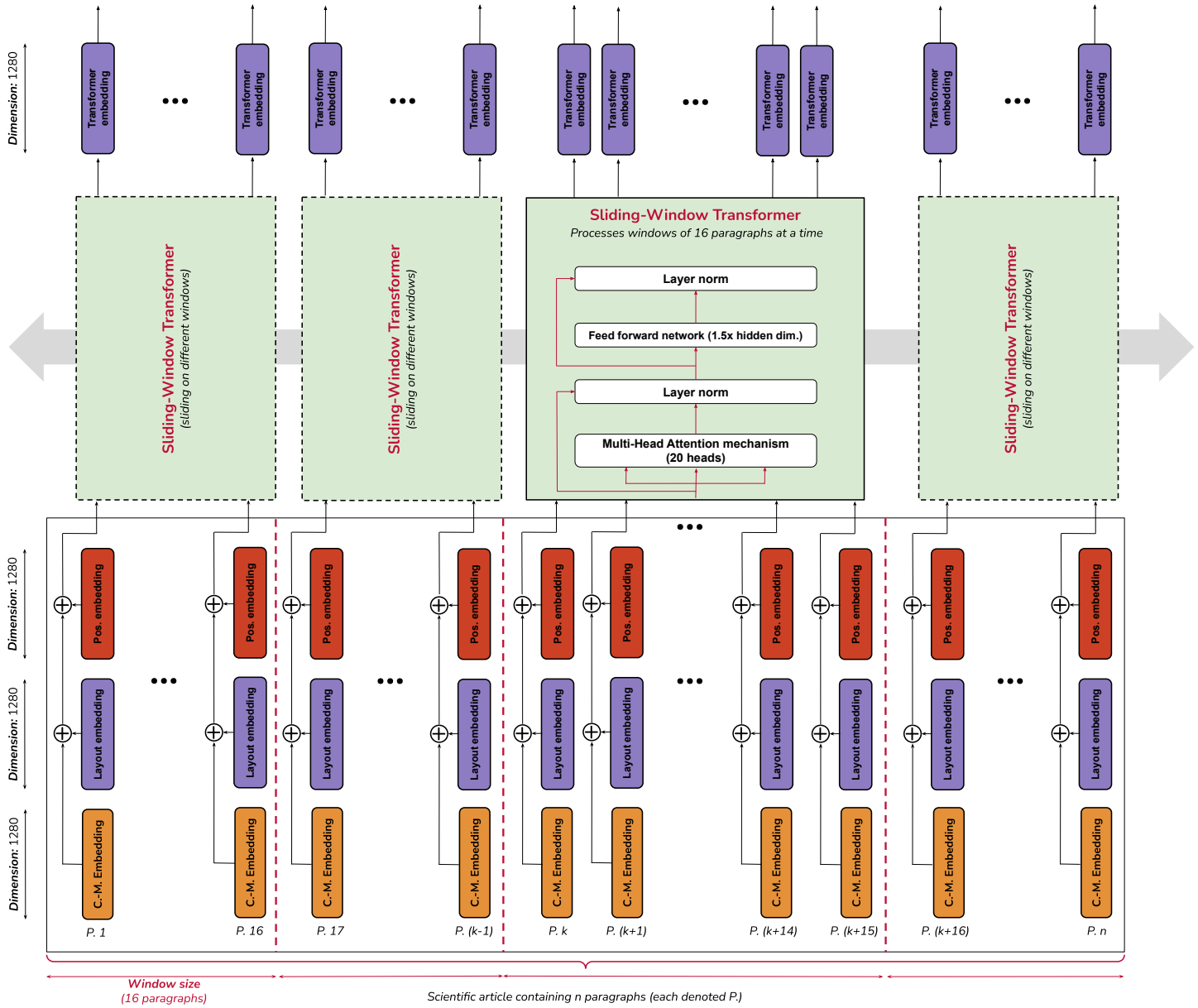


Figure 3.14: Sequential model based on a sliding-window (SW) transformer architecture

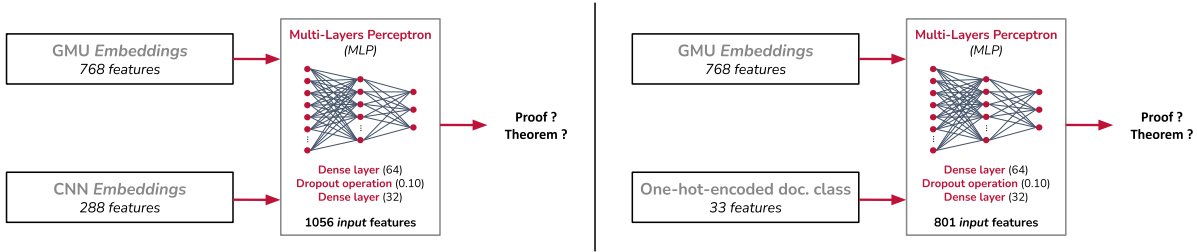


Figure 3.15: Diagram of the two MLP models with document class information merged at paragraph level

Model	Accuracy	F_1 -Score
GMU model (Baseline)	76.86%	73.87%
MLP with 1 056 input features	78.85%	76.11%
MLP with 801 input features	78.92%	76.12%

Table 3.4: Microscopic accuracy and macroscopic F_1 -score for the MLP modelization compared to its baseline

contrast, Yacine’s report built upon the results from my extraction phase, employing a DistilBERT model to analyse paragraph-level features.

Text proves to be a powerful modality in the multimodal setting. Shufan’s comparison between GPT-4 and my pre-trained language model highlighted the potential for knowledge transfer despite the limitations of my model due to restricted training data (only 11GB available).

Antoine’s work on document classification also plays a crucial role. His innovative approach suggests that identifying a document’s class from its first page could be used to predict theorems and proofs within that document class. This insight could significantly enhance the Sliding Window transformer’s performance by incorporating document class information, suggesting a promising direction for future research, as evidenced by Antoine’s preliminary experiments (see Tables 3.4, 3.5) to include document class both at multimodal fusion and CRF level (see Figures 3.15, 3.16).

Antoine’s efforts were directed at integrating document classification data into our model at two distinct levels: at the granularity of individual paragraphs and across entire papers by incorporating document class information at various points.

Incorporating the document class at the paragraph level yielded some positive results, whereas doing so at the level of the entire document marginally improved. This discrepancy can be attributed primarily to the limitations of Simple Conditional Random Fields (CRFs) could struggle to capture complex dependencies (a bunch of paragraphs before and after). At that time, we had yet to fully implement an approach utilising the sliding window (SW) transformer, so we could not provide document class results using the transformer-based approach.

A potentially more effective strategy might involve incorporating the document class as a separate token, perhaps even replacing the traditional [CLS] token’s random initialisation in the multimodal Transformer architecture. Given that the document class essentially encapsulates the core theme of

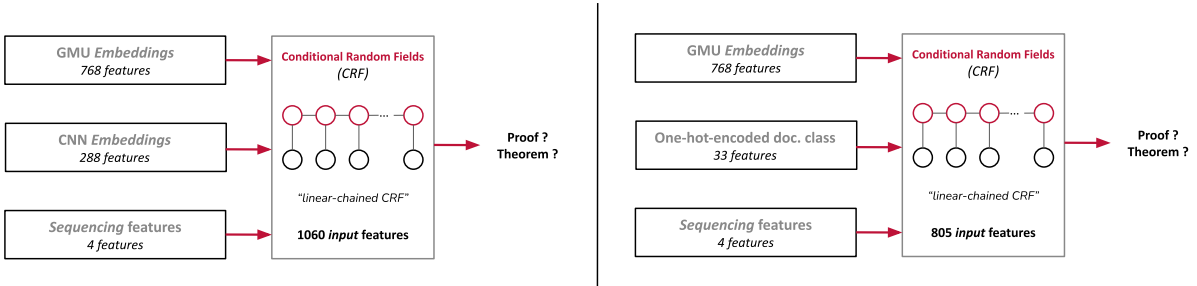


Figure 3.16: Diagram of the two CRF models using document class information operating at the document level, adding document class information at every paragraph

Table 3.5: Microscopic accuracy and macroscopic F_1 -score for the CRF model with document class, compared to its baseline

Model	Accuracy	F_1 -Score
CRF with multimodal (Baseline)	84.38%	83.01%
CRF with 1060 features (CNN embedding)	84.49%	83.13%
CRF with 805 features (document class one-hot-encoding)	84.52%	83.11%

a paper, it seems logical that a transformer model could more accurately map these dependencies by acknowledging the overarching context provided by the document class. This approach could pave the way for a more nuanced and effective integration of document-level information, potentially overcoming the limitations observed with Simple CRFs; such information can also be added to the HAT (Hierarchical Attention Transformer) by replacing the random [CLS] initialisation token with a document class.

3.6 Dataset Construction and Preprocessing Tools

In the absence of a publicly accessible dataset comprising PDF articles alongside their theoretical findings, the initial phase of this endeavour entailed the construction of such a repository. Fortunately, arXiv – a repository harbouring over one million open-access scholarly papers – provides a way to retrieve their entire dataset and source files hosted on Amazon S3⁴.

3.6.1 Collecting and Filtering Dataset

The entirety of the arXiv database, encompassing both PDFs and source materials, was procured. This dataset embodies nearly 1.7 million scholarly documents (all papers on arXiv until May 2020). After removing all that did not contain a theorem or proof, we had 460 thousand papers. After further filtering to remove non-English papers or papers that failed to compile or for which one of the tools used failed, we ended up with a final dataset of 197 thousand papers, which I used for pre-training our language model. The pretraining is applied to the entire corpus to capture most tokens and avoid OOV relationships. However, a separate validation dataset is used for finetuning.

This heuristic entailed the exclusion of any document devoid of theorems, lemmas, or propositions identified through the application of Poppler’s pdftotext [Fre18] and a rudimentary regular expression.

Following this refinement process, the collection was condensed to approximately 500 000 documents, each purportedly encompassing mathematical content. As delineated in Figure 3.2, there is a discernible exponential growth in the annual publication volume of papers. Additionally, it is observed that the fraction of scientific documents featuring mathematical content has been on the rise, peaking in the year 2015. This is likely due to the rise of deep learning, with few theoretical papers.

To Extract labelled information, we rely on a 2-step process:

1. Injecting the labels inside the L^AT_EX sources
2. Producing an Annotation XML (with ground truths of the market environment)

A rough illustration of both the steps is provided below (see Figure 3.17):

In the DocEng paper, to establish a proof of concept, we utilised a deliberately smaller dataset that is formed of 4 400 randomly selected articles from the CS-CC category of arXiv (namely, Computer Science, Computational Complexity) from 2010 to 2020⁵, processed via pdftalto, which generated 1 064 955 lines. Of these, 80% (851964) were used for training and 20% (212991) for validation. Due to the line-based nature of our analysis, lines from the same document were distributed across both training and validation sets, with a stratified condition set to True to ensure similar label distribution.

These PDFs came from diverse backgrounds; It is interesting to see the global research community come to life in such a visual way. We constructed this chart by geo-linking author information from the same 4 400 PDFs, identifying their institutions, extracting author names from Grobid, and then finding the coordinates of the academic institute with the Google Maps API (via a geo-JSON file). When grouped,

⁴https://info.arxiv.org/help/bulk_data_s3.html

⁵See <https://github.com/PierreSenellart/theoremkb> for links to individual papers; we cannot redistribute the entire dataset publicly because of licensing issues.

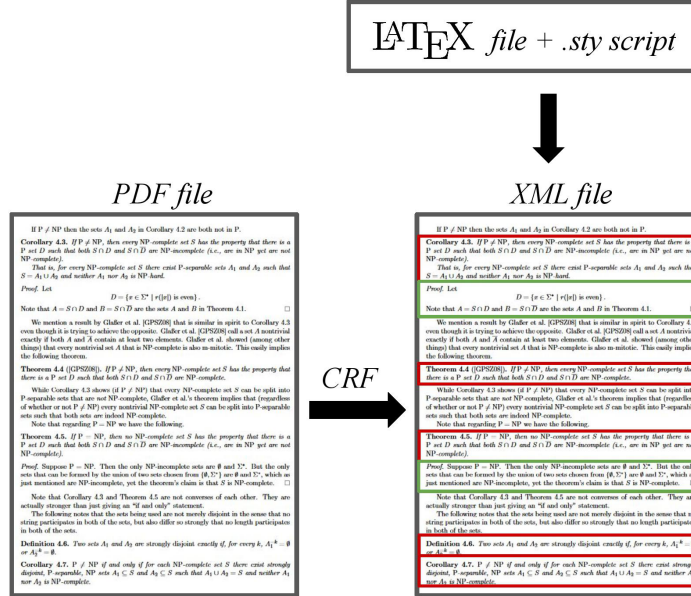


Figure 3.17: Compiling the L^AT_EX Sources to mark ground truths embedded in the PDF

every author’s country information represents the dataset’s underrepresented sections, which also relate to real-life cases. Many countries in Africa are underrepresented in their scientific contributions (see figure 3.18). This method offers a dynamic view of where and how academic work is being generated worldwide.

Our Multimodal dataset, encompassing all arXiv papers up to May 2020, was acquired via arXiv’s bulk data access on Amazon S3. We developed an annotation script to pinpoint theorem-like environments and proofs within these documents, leveraging L^AT_EX sources. This involved crafting a L^AT_EX package to annotate commands like `\newtheorem` for precise identification in the compiled PDFs, illustrated in Figure 3.20. We filtered articles from the dataset to only keep those in English, for which L^AT_EX source is available (according to arXiv’s policy, all those that have been produced using L^AT_EX), that were compilable on a modern L^AT_EX distribution that contained at least a theorem or a proof environment for which none of the tools (our ground-truth annotation package, `Grobid` for extraction of blocks, `pdfalto` for line-by-line font sequences, bitmap image rendering for CNN’s) failed to produce a valid output. We stress that L^AT_EX sources are only used to produce ground-truth annotations; they are not required at inference time. `Grobid` sometimes fails to extract correct paragraphs, i.e., some of the paragraphs identified by `Grobid` overlap blocks of different categories (say, *basic* and *theorem*). We label such paragraphs with the *overlap* class, exclusively used for such outliers. We ran our script(generate data.csv) on about 196k PDFs. We limited the training to several batches due to varying convergence rates across modalities. The order of batches was consistent to facilitate performance tracking. We fed the identical batches sequentially to train sequential models that capture paragraph dependencies to maintain relative order. Here, we also introduce an overlap class where multiple classes overlap (say a proof with basic).

Our validation set comprises approximately 500 000 paragraph blocks from 3 682 randomly selected PDF articles. The remaining articles formed the training dataset, which is used entirely for pretraining our language model after filtering potential personal information such as author names and institutions from `Grobid` extractions to minimise privacy concerns. The dataset is heavily imbalanced, with the number of paragraphs labelled as basic: 314 501, proof: 125 524, theorem: 85 801, and overlap: 3 470. The training involved dividing the dataset into batches of 1 000 PDF articles, incrementally fitting classifiers on these batches until convergence, without exceeding a few dozen batches. Post-training, classifiers’ weights were frozen for integration into the multimodal classifier, subsequently employed as feature extractors for the sequential approaches detailed in Chapter 7.

All experiments were run on a supercomputer with access at any point to 4 NVIDIA (V100 or A100) GPUs. We estimate to 8 000 GPU hours the computational cost of the entire prototyping, hyperparameter tuning, training, validation, and evaluation pipeline.

Reserach organization outputs

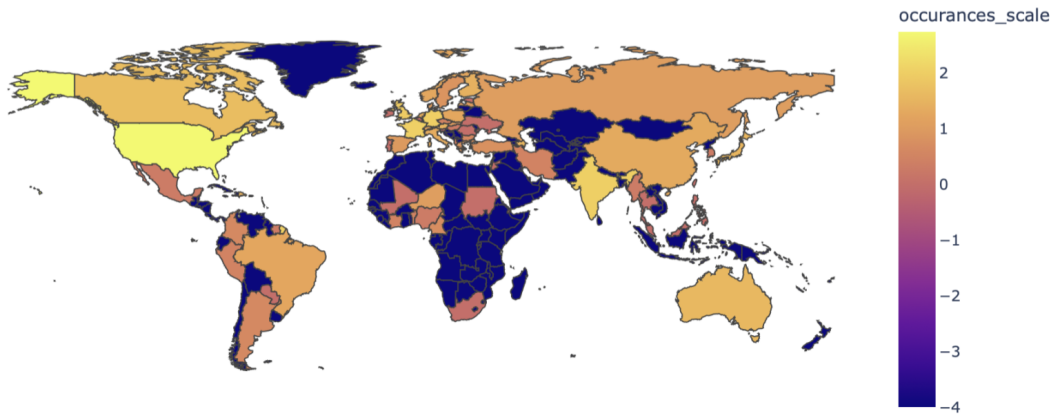


Figure 3.18: Mapping the Academic Pulse: A Choropleth map visualisation of research contributions based on 4000+ PDFs

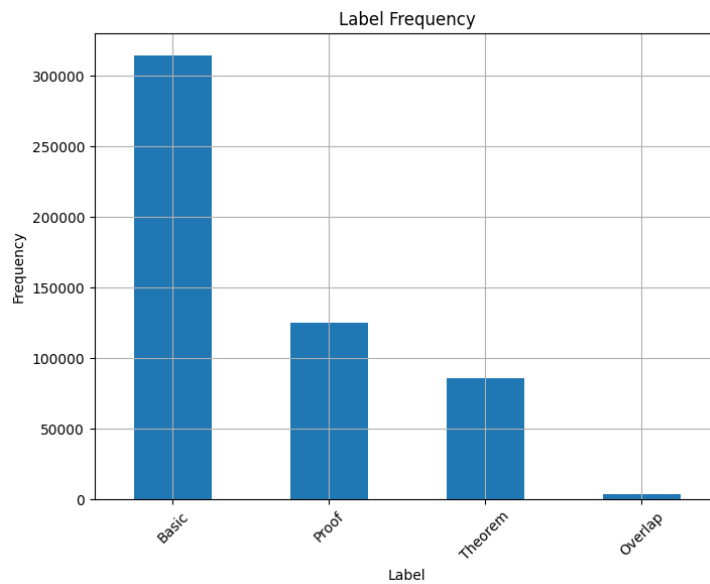


Figure 3.19: Class imbalance among the labels within the validation dataset (multimodal)

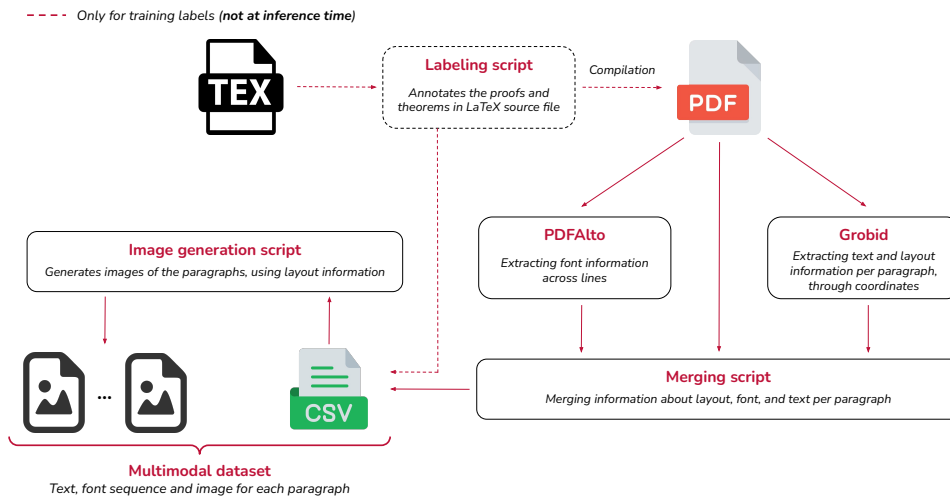


Figure 3.20: An Example of how the annotations in the XML can be visualised

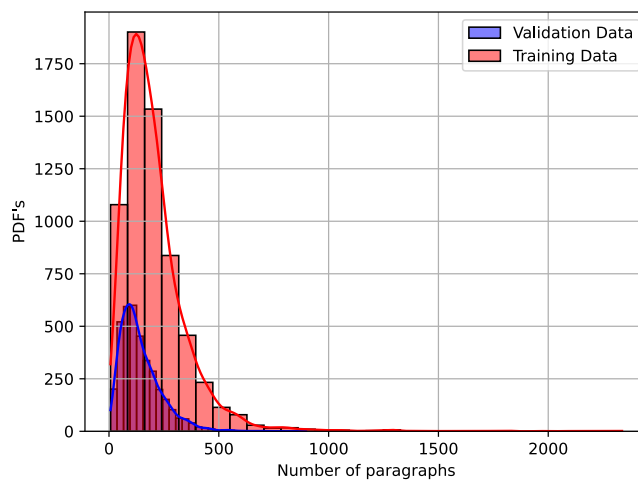


Figure 3.21: Data distribution showing the number of paragraphs in training/validation data (multimodal)

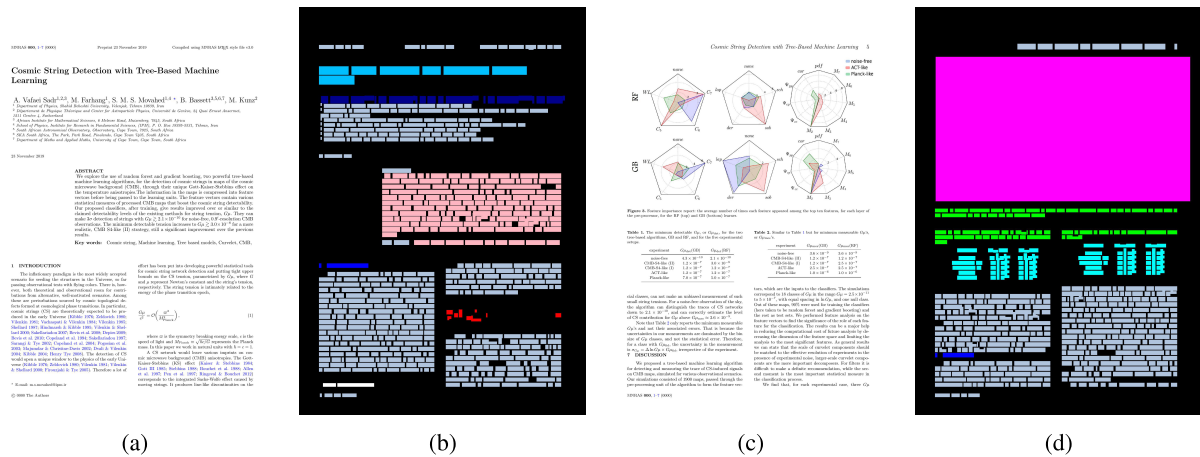


Figure 3.22: Example annotations of the DocBank. The colours of semantic structure labels are Abstract, Author, Figure, Footer, List, Paragraph, Reference, Section, Table, and Title.

3.6.2 Using L^AT_EX Sources for Ground Truth Labeling

L^AT_EX, a comprehensive software and programming language, is engineered to streamline the creation of textual documents by emphasising semantic content while managing formatting intricacies. Specifically, in the context of theoretical findings, L^AT_EX facilitates the use of specialised environments (e.g., `\begin{theorem}` ... `\end{theorem}`) to denote areas requiring distinct formatting, implemented via the `\newtheorem` command. This characteristic permits the identification of theorem sections within scholarly papers.

During Daria’s project, a L^AT_EX plugin was developed to extract all results from given source files. This plugin modifies the `\newtheorem` command to insert a hook that segregates each Result onto a separate page. For optimal function, the plugin must be integrated after the command’s definition before its application. The compilation of extracted results serves as a foundation for machine learning endeavours. However, this technique presents limitations in rapidly determining whether a specific term within the article is associated with a result. It necessitates a sequential review of the PDF and matching its content against the list of extracted outcomes.

Lucas proposed an innovative alternative: adapting the L^AT_EX script to generate bespoke PDF links without altering the original document layout. The revised approach yielded a collection of labelled bounding boxes that delineate the article’s findings. Accompanied by a spatial index, this modification enables swift determination of each token’s classification—whether it is a result.

This methodology was applied to 6000 articles sourced from arXiv within the complexity theory domain. Despite the strategy’s overall success, several challenges were encountered, such as excluding 258 papers not composed in L^AT_EX and compilation errors arising from improper script inclusion. Additionally, the heterogeneity in defining results within L^AT_EX documents posed difficulties in achieving comprehensive extraction. Variations included theorems with unique names, formulations in different languages, or definitions not utilising `\newtheorem`. While such discrepancies may introduce biases, potentially skewing the representativeness of the dataset toward machine learning model training, they were deemed manageable within the scope of creating a sufficiently large dataset.

Introduced in the late 2020s, another significant dataset, known as DocBank [LXC+20], embarked on a mission parallel to ours, focusing on the extraction of document content. Unlike our methodology, DocBank employs a novel approach by integrating a color-coded schema, see Figure 3.22 with the `\color{fontcolor}` L^AT_EX command to delineate specific sections of scholarly papers, Each colour represents a distinct section, such as the author, title, abstract, equation, etc.

This methodological innovation facilitated a visually distinct representation of high-level paper structures, enhancing the dataset’s utility for content extraction and document analysis tasks. Figure 3.22 illustrates the application of DocBank’s colour-based schema, showcasing the Rendering of a paper with clearly marked sections, thereby aiding in identifying and classifying document components for further processing.

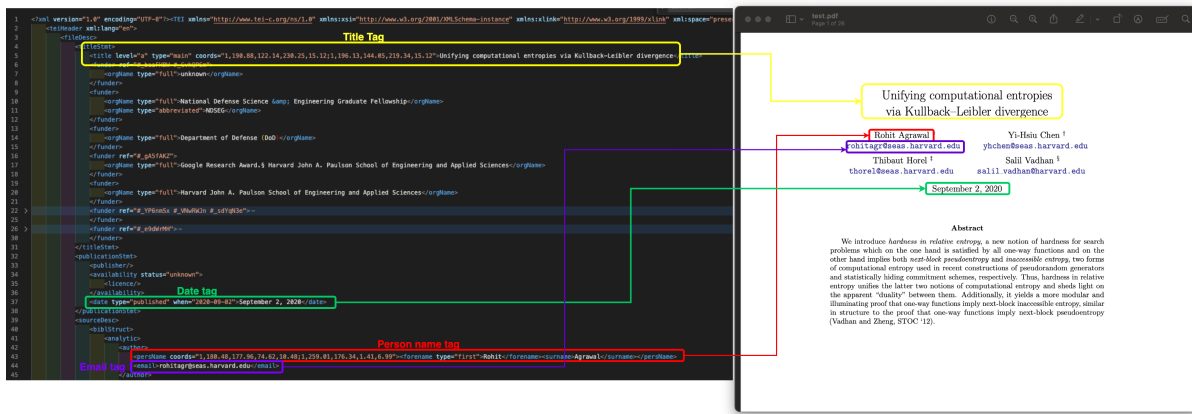


Figure 3.23: Example of Semantically parsed information represented in tags present within the TEI/XML obtained from passing a pdf to GROBID

3.6.3 Producing an Annotated XML File

Grobid Extraction

Grobid (GeneRation Of Bibliographic Data) [L+24a] is an advanced open-source machine learning library designed for extracting, parsing, and restructuring scholarly documents into structured XML/TEI formats. Utilising techniques like conditional random fields and support vector machines, Grobid efficiently processes scientific publications, identifying and segmenting text into metadata, references, figures, and tables. This facilitates the creation of detailed bibliographic metadata and enhances the accessibility of academic research.

A key feature of Grobid is its precise handling of bibliographic references and its ability to preserve mathematical expressions and chemical formulas, making it invaluable for STEM fields. It supports a wide range of applications, from digitising archives to automating manuscript conversion for publishers, significantly contributing to digital library science and the management of scientific knowledge.

Grobid’s scalability and accuracy in structuring unstructured data address the critical challenge of accessing and analysing the vast amount of scholarly content. Transforming documents into machine-readable formats not only aids academic research but also promotes the global dissemination and preservation of scientific information, proving essential in the era of big data and digital scholarship.

Unlike the costly OCR preprocessing often used in document AI tasks for items like bills and receipts, Grobid is explicitly tailored for scientific documents. It not only identifies bounding boxes around characters, including mathematical symbols but also semantically segments blocks into lines and paragraphs, which aids in determining paragraph edges in research. Furthermore, Grobid can parse titles, author names, bibliographical details, and funding information, which could benefit the TheoremKB project. For visualisation, refer to figure 3.23. Grobid achieves a processing speed of about 10.6 PDFs per second⁶ (around 915,000 PDFs daily, around 20M pages daily). A demo of Grobid API is hosted online⁷, with complete installation instructions to run locally provided on its GitHub repository⁸.

pdfalto Extraction

pdfalto [L+24b] is a powerful open-source tool for converting PDF documents into XML, HTML, and text formats. It stands out for its ability to accurately extract text, tables, and images while preserving the original layout and formatting of the document, see Figure 3.24. Developed with a focus on accessibility and interoperability, pdfalto enables users to repurpose and analyse content from PDF files efficiently, making it an invaluable resource for digital libraries, content managers, and researchers. Its straightforward command-line interface facilitates easy integration into automated workflows, allowing for the batch processing of documents and enhancing productivity across various document management and digitisation

⁶<https://github.com/kermitt2/grobid>

⁷<https://kermitt2-grobid.hf.space/>

⁸<https://github.com/kermitt2/grobid>


```

▼<OCRProcessing ID="IdOcr">
  ▼<ocrProcessingStep>
    <processingDateTime>2020-06-13T15:55:07Z</processingDateTime>
    ▼<processingSoftware>
      <softwareCreator>CONTRIBUTORS</softwareCreator>
      <softwareName>pdfalto</softwareName>
      <softwareVersion>0.3</softwareVersion>
    </processingSoftware>
    </ocrProcessingStep>
  </OCRProcessing>
</Description>
▶<Styles>...</Styles>
▼<Layout>
  ▶<Page ID="Page1" PHYSICAL_IMG_NR="1" WIDTH="612.000" HEIGHT="792.000">...</Page>
  ▼<Page ID="Page2" PHYSICAL_IMG_NR="2" WIDTH="612.000" HEIGHT="792.000">
    ▼<PrintSpace>
      ▼<TextBlock ID="p2_b1" HPOS="190.876" VPOS="122.136" HEIGHT="37.0331" WIDTH="230.252">
        ▼<TextLine WIDTH="230.252" HEIGHT="15.1151" ID="p2_t1" HPOS="190.876" VPOS="122.136">
          <String ID="p2_w1" CONTENT="Unifying" HPOS="190.876" VPOS="122.136" WIDTH="59.1074" HEIGHT="15.1151">
            <STYLEREFS="font1"/>
            <SP WIDTH="5.1991" VPOS="122.136" HPOS="249.983"/>
            <String ID="p2_w2" CONTENT="computational" HPOS="255.182" VPOS="122.136" WIDTH="99.0402" HEIGHT="15.1151">
              <STYLEREFS="font1"/>
              <SP WIDTH="5.1991" VPOS="122.136" HPOS="354.222"/>
              <String ID="p2_w3" CONTENT="entropies" HPOS="359.421" VPOS="122.136" WIDTH="61.7069" HEIGHT="15.1151">
                <STYLEREFS="font1"/>
              </TextLine>
            ▼<TextLine WIDTH="219.341" HEIGHT="15.1151" ID="p2_t2" HPOS="196.127" VPOS="144.054">
              <String ID="p2_w4" CONTENT="via" HPOS="196.127" VPOS="144.054" WIDTH="20.5363" HEIGHT="15.1151">
                <STYLEREFS="font1"/>
                <SP WIDTH="5.1991" VPOS="144.054" HPOS="216.663"/>
                <String ID="p2_w5" CONTENT="Kullback-Leibler" HPOS="221.862" VPOS="144.054" WIDTH="116.646" HEIGHT="15.1151">
                  <STYLEREFS="font1"/>
                  <SP WIDTH="5.1991" VPOS="144.054" HPOS="338.508"/>
                  <String ID="p2_w6" CONTENT="divergence" HPOS="343.707" VPOS="144.054" WIDTH="71.7607" HEIGHT="15.1151">
                    <STYLEREFS="font1"/>
                  </TextLine>
                </TextBlock>

```

Figure 3.24: Result of the processing by pdfalto: (A) style block (information about fonts); (B) page block; (C) character string; (D) spacing information

projects. Full instructions on how to install and run pdfalto tool, are available on its GitHub repository⁹.

To extract spatial coordinates for embedded links within PDF documents, PDF Alto is explicitly utilised to target proofs and theorems. This tool generates XML files that outline the coordinates of these elements accurately. The spatial coordinates thus obtained can be applied directly or modified slightly to fit various modalities into classifier development and evaluation. The approaches include:

1. **Spatial Coordinate-Based Classifier Training:** Spatial coordinates are used to train classifiers to recognise and predict the locations of proofs and theorems within page images. This method involves object detection models, such as YOLO or RCNN, with the page image as input.
2. **Line-Level Text Classification via PDF Alto:** PDF Alto is employed to identify text lines within the annotated ground truth boxes. Classifiers are then trained on these lines, offering a detailed yet narrowed field of vision for the model due to the significantly smaller aspect ratio of text lines compared to paragraphs.
3. **Grobid for Paragraph and Sentence Segmentation:** The segmentation capabilities of Grobid extend the focus beyond individual text lines to paragraphs and sentences. Using spatial coordinates from Grobid, paragraphs that fully or partially overlap with the designated proofs/theorems are identified. Labels are adjusted to reflect either a complete fit or an overlap. Overlaps might occur across multiple annotations, such as consecutive proofs and theorems that Grobid misclassifies as a single unit.
4. **Merging Script for Comprehensive Data Compilation:** A merging script combines outputs from PDF Alto, Grobid, and manual annotations into a unified data.csv. This file encompasses structured and segmented information at the proof/theorem level, providing a rich dataset with multimodal insights, including spatial coordinates and paragraph-level font information, for a high level overview refer Figure 3.25.

These approaches facilitate the use of spatial coordinates and textual content in developing and evaluating classifiers, offering various document analysis and processing strategies.

⁹<https://github.com/kermitt2/pdfalto>

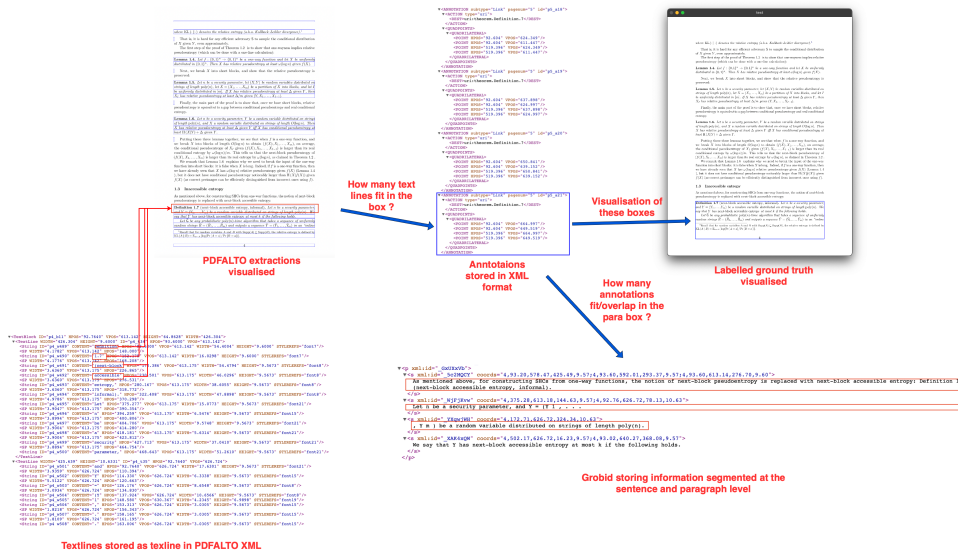


Figure 3.25: An Example of how the annotations in the XML can be visualised

3.6.4 Related Datasets

Our research incorporates the use of Grobid [L⁺24a] for extracting text and bounding boxes from images that display paragraphs, serving as an essential preliminary step in training our models. This effort aligns with the goals set in the Publaynet investigation [ZTJ19], with a particular emphasis on detecting 'Proofs' and 'Theorems.' in academic papers. Our approach differs from Publaynet's method of organising document segments into categories like Text, Tables, and Figures at a broader document scale as we delve deeper into scholarly texts, using L^AT_EX from arXiv submissions to create accurate ground truths, unlike Publaynet, which employs the National Library of Medicine (NLM)¹⁰ framework for categorising journal content.

At its essence, Grobid analyses a variety of document elements such as lists, figures, titles, and references, mirroring the functionalities of Publaynet [ZTY19]. It uniquely identifies and semantically processes text into sentences and paragraphs, pinpointing block coordinates, thereby facilitating a textual modality in our analysis. Importantly, Grobid retains mathematical expressions within texts, a capability not emphasised by Publaynet, which excludes specific XML nodes like `tex-math` and `disp formula`, as indicated in the PMCOA XML documentation (Page 2 of [ZTY19]). This capability highlights our method's suitability for the precise needs of our research and its broader objectives. Whereas Publaynet focuses on visual classifications across a range of labels, including text, Figures and Tables, our study underscores the critical role of text in distinguishing proofs within paragraphs, highlighting the multimodal challenge we face where text analysis is critical.

The introduction of Docbank [LXC⁺20] represents a significant step forward from Publaynet [ZTY19], with its dataset of 500K document images designed for both training and evaluation purposes. Diverging from Publaynet's focus on the medical sciences, Docbank covers a broader spectrum of academic fields such as Physics, Mathematics, and Computer Science, enriching the dataset with various mathematical equations. A notable aspect of Docbank is its dual annotation levels, covering both tokens and segments, making it suitable for various computer vision and natural language processing tasks. However, for projects aimed at identifying proofs, Docbank's extensive annotation coverage, which includes authors, abstracts, titles, equations, and paragraphs, may limit its direct applicability due to the absence of specific labels for proofs or theorems in our focused study area.

An essential attribute of Docbank is its collection of documents from arXiv each tagged with an arXiv ID. This connection allows for the utilisation of L^AT_EX sources from these documents, facilitating the use of our preprocessing script for annotating ground truths (proofs and theorems) within the Docbank dataset, thus enhancing its usefulness. Adopting the `\begin` command for annotations in Docbank aligns with our methodology for delineating structural parts in academic documents, showing a convergent approach in identifying and examining document elements such as proofs and theorems.

Doclaynet [PAD⁺22], in a similar vein to Docbank [LXC⁺20] and Publaynet [ZTY19], focuses on layout

¹⁰<https://dtd.nlm.nih.gov/>

detection across a diverse set of 80K document examples. This collection expands the scope to encompass various types of document layouts, striving for the precision seen in models such as FASTRCNN [RHGS15] and YOLOv5 [Joc20]. An identified complication in Doclaynet is the occurrence of overlapping labels, a situation where blocks can have intersecting categories, a complexity that our study also recognises. To address this, both investigations prioritise the analysis of distinct, non-overlapping blocks for accuracy assessment.

Interestingly, our validation dataset, encompassing approximately 80K images, aligns closely in scale with Docbank 's (around 50K) and exceeds that of Doclaynet (around 6K images), providing a substantial basis for our task. Our findings further reveal that a relatively modest collection of a few thousand PDFs (as depicted in Figures 5.14, 5.38, 5.50) suffices to enhance performance on the validation data, underscoring the efficiency of our unimodal approaches.

Generating Multimodal CSV File

The flowchart crafted serves as a visual guide through a multifaceted pipeline designed to process PDF documents. This elaborate workflow extracts textual and font information from documents and amalgamates these data into an enriched data frame, facilitating a deeper analysis and understanding of the document's content. The following points elaborate on each critical step within this pipeline, highlighting the complexity and thoroughness of the approach:

1. Start: PDF Document

- (a) Represents the initiation of the process with a PDF document as the input.

2. pdfto Processing

- (a) Converts the PDF to an XML format, capturing intricate details like spatial layout and font attributes.
 - i. *XML Conversion*: Transforms PDF into XML for detailed layout analysis.
 - ii. *Attribute Extraction*: Gathers font, text line, and layout information.

3. GROBID Processing

- (a) Extracts structured metadata and semantic annotations from the PDF.
 - i. *Metadata Extraction*: Retrieves bibliographic data and document metadata.
 - ii. *Document Structure Analysis*: Identifies logical document sections and components.

4. Fonts2Vec

- (a) Vectorises font information to enable semantic significance analysis.
 - i. *Attribute Identification*: Determines font characteristics for vectorization.
 - ii. *Vector Transformation*: Converts font attributes into numerical vectors.

5. Merge Using pdfto

- (a) Integrates data from pdfto and GROBID, optionally using Fonts2Vec for a unified document view.
 - i. *Data Alignment and Integration*: Aligns and combines information from multiple sources.
 - ii. *Visual Styling Cues Incorporation*: Adds visual styling information to the dataset.

6. Compute Skip

- (a) Adjusts for page count discrepancies to ensure data alignment.

7. Get Page-wise Text

- (a) Organises text extraction by pages, including their spatial coordinates and associated font information.
 - i. *Text Line Extraction*: Extracts text lines for each page based on XML output.
 - ii. *Font Information Tagging*: Tags each text line with font information extracted from the document.

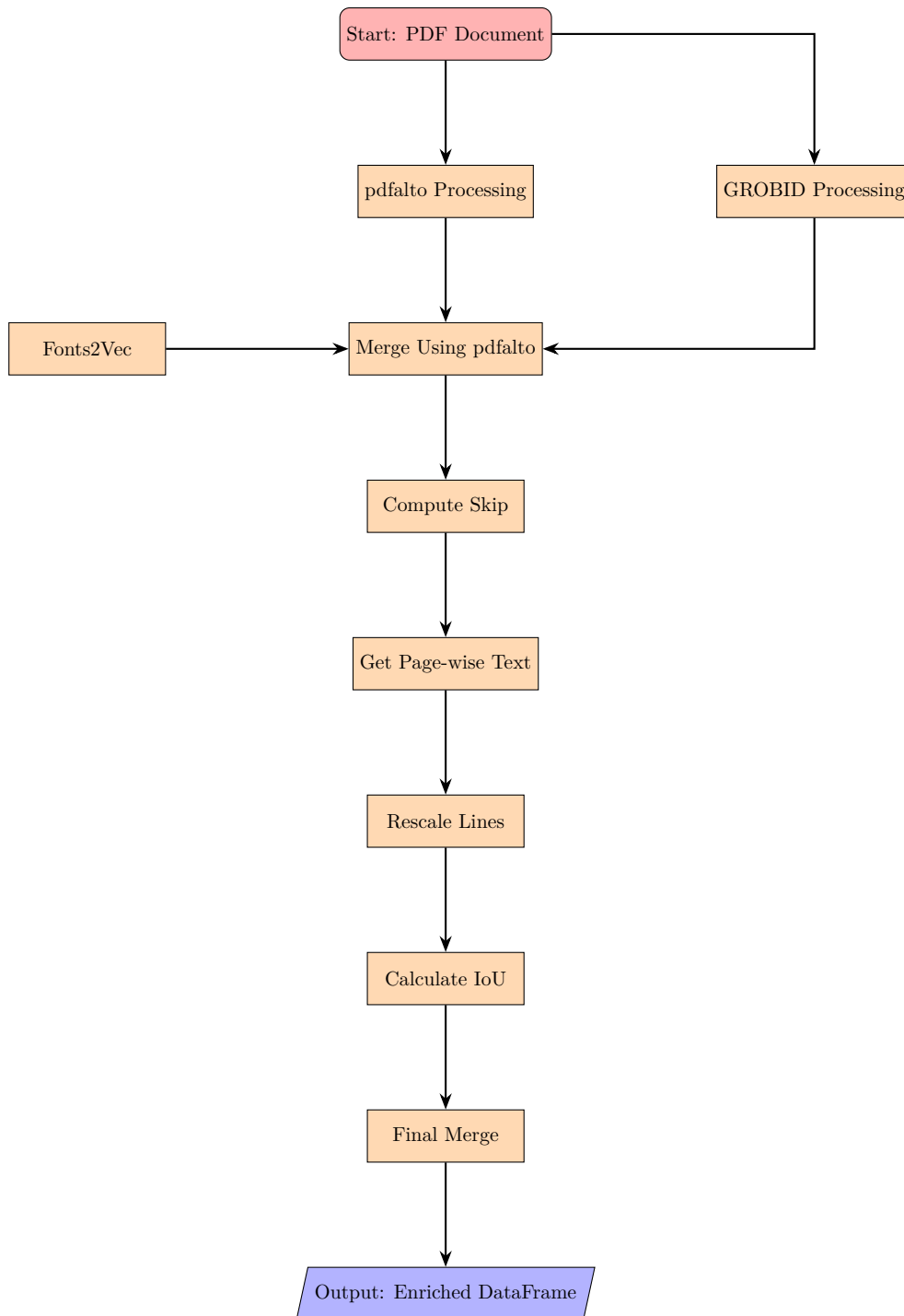


Figure 3.26: High-level code functions involved to generate `data.csv`

8. Rescale Lines

- (a) Adjusts spatial coordinates of text lines according to scaling factors for accurate document layout representation.
 - i. *Scaling Factor Application*: Applies scaling to coordinate data to match the PDF view.

9. Calculate IoU (Intersection over Union)

- (a) Computes overlap between bounding boxes for accurate text segment alignment.
 - i. *BoundingBox Calculation*: Determines bounding boxes for text lines and annotations.
 - ii. *Overlap Computation*: Uses IoU to identify significant overlaps for data merging.

10. Final Merge

- (a) Combines spatial, textual, and font information into a comprehensive data frame.
 - i. *Spatial and Textual Integration*: Merges information based on spatial and textual alignment.
 - ii. *Label Assignment*: Assign labels based on text segments' alignment and overlap.

11. Output: Enriched DataFrame

- (a) Produces a data frame consolidating textual content, structural information, and font styling details.
 - i. *Analysis-Ready Format*: Prepares the DataFrame for easy access, analysis, or further processing.
 - ii. *Visualisation and Processing Ready*: The data is now prepared for potential visualisation or machine learning tasks.

3.6.5 Problems with Extraction

The performance of most trained models is significantly influenced by the data generation step, wherein tools like Grobid and pdfalto are essential. Although these tools are specifically designed for scientific documents and facilitate faster data conversion from PDFs to XML, the discussion remains incomplete without addressing the notable flaws inherent in these approaches.

One major flaw in Grobid and pdfalto is their misalignment in detecting bounding boxes for lines and paragraphs. For instance, pdfalto often misinterprets page numbers as text lines (see Figure 3.27), along with many blank lines. In the DocEng paper, I attempted to rectify this by analysing the number of contours in each text line, disregarding those with few or none. Although these heuristics removed some erroneous blocks, they were not flawless and may have slightly impacted the accuracy. Grobid similarly struggles with misalignment and incorrect text detection, leading to merged paragraphs (Overlaps) or incorrect section identifications, see Figure 3.28. This ongoing issue¹¹ was also discussed with Grobid's creator, Patrice Lopez. For Patrice to investigate and improve Grobid, I shared with him a list of 132 940 problematic paragraphs (Overlaps) from 1 200 PDFs out of the total 197k PDFs to improve system robustness.

The bugs may also originate from the L^AT_EX extraction script. One effective method to identify such errors involves training different classifiers on the noisy data, expecting the model to generalise well on clear cases despite some data outliers. I evaluated the entropy on confidence intervals across three unimodal approaches within the DocEng dataset for more precise analysis, using Grobid to extract paragraphs. This approach helped identify misclassified points where significant disagreement occurred between modalities like text, vision, and font.

Using the Label Studio [Hum24] tool, we could easily visualise problematic paragraphs and apply labels with three trained models at the paragraph level. During a manual review of 4 400 PDFs, We identified two main issues with the extraction script: it failed to capture proofs that were not written with English characters and mislabeled mathematical results within proofs. We revised our script to handle nested results. Due to the vast amount of data, we focused our manual efforts were made on points with high entropy, totalling about 3 000, and I manually labelled around 300 daily over ten days.

We hope that a multimodal system with a sequential component will address some of these bugs, as it relies on a combination of vision, text, and font data. This integrated approach can reduce ambiguities that are difficult to resolve when a single modality is compromised due to bugs in the extraction step.

¹¹<https://github.com/kermitt2/grobid/issues/825>

where $\text{KL}(\cdot \parallel \cdot)$ denotes the relative entropy (a.k.a. Kullback–Leibler divergence).¹

That is, it is hard for any efficient adversary S to sample the conditional distribution of X given Y , even approximately.

The first step of the proof of Theorem 1.2 is to show that one-wayness implies relative pseudoentropy (which can be done with a one-line calculation):

Lemma 1.4. *Let $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a one-way function and let X be uniformly distributed in $\{0, 1\}^n$. Then X has relative pseudoentropy at least $\omega(\log n)$ given $f(X)$.*

Next, we break X into short blocks, and show that the relative pseudoentropy is preserved:

Lemma 1.5. *Let n be a security parameter, let (X, Y) be random variables distributed on strings of length $\text{poly}(n)$, let $X = (X_1, \dots, X_m)$ be a partition of X into blocks, and let I be uniformly distributed in $[m]$. If X has relative pseudoentropy at least Δ given Y , then X_I has relative pseudoentropy at least Δ/m given (Y, X_1, \dots, X_{I-1}) .*

Finally, the main part of the proof is to show that, once we have short blocks, relative pseudoentropy is *equivalent* to a gap between conditional pseudoentropy and real conditional entropy.

Lemma 1.6. *Let n be a security parameter, Y be a random variable distributed on strings of length $\text{poly}(n)$, and X a random variable distributed on strings of length $O(\log n)$. Then X has relative pseudoentropy at least Δ given Y iff X has conditional pseudoentropy at least $H(X|Y) + \Delta$ given Y .*

Putting these three lemmas together, we see that when f is a one-way function, and we break X into blocks of length $O(\log n)$ to obtain $(f(X), X_1, \dots, X_m)$, on average, the conditional pseudoentropy of X_I given $(f(X), X_1, \dots, X_{I-1})$ is larger than its real conditional entropy by $\omega(\log n)/m$. This tells us that the next-block pseudoentropy of $(f(X), X_1, \dots, X_m)$ is larger than its real entropy by $\omega(\log n)$, as claimed in Theorem 1.2.

We remark that Lemma 1.6 explains why we need to break the input of the one-way function into short blocks: it is false when X is long. Indeed, if f is a one-way function, then we have already seen that X has $\omega(\log n)$ relative pseudoentropy given $f(X)$ (Lemma 1.4), but it does not have conditional pseudoentropy noticeably larger than $H(X|f(X))$ given $f(X)$ (as correct preimages can be efficiently distinguished from incorrect ones using f).

1.3 Inaccessible entropy

As mentioned above, for constructing SHCs from one-way functions, the notion of next-block pseudoentropy is replaced with next-block accessible entropy:

Definition 1.7 (next-block accessible entropy, informal). *Let n be a security parameter, and $Y = (Y_1, \dots, Y_m)$ be a random variable distributed on strings of length $\text{poly}(n)$. We say that Y has next-block accessible entropy at most k if the following holds.*

Let \tilde{G} be any probabilistic $\text{poly}(n)$ -time algorithm that takes a sequence of uniformly random strings $R = (R_1, \dots, R_m)$ and outputs a sequence $\tilde{Y} = (\tilde{Y}_1, \dots, \tilde{Y}_m)$ in an “online

¹Recall that for random variables A and B with $\text{Supp}(A) \subseteq \text{Supp}(B)$, the relative entropy is defined by $\text{KL}(A \parallel B) = \mathbb{E}_{a \leftarrow A} [\log(\Pr[A = a] / \Pr[B = a])]$.

Figure 3.27: A visualisation of extraction from pdfalto

ASYMPTOTIC INDUCED MATCHING NUMBER OF HYPERGRAPHS 2

$\subseteq (V_1 \times W_1) \times \dots \times (V_k \times W_k)$.

and we naturally define the power $\Phi^{2n} = \Phi \otimes \dots \otimes \Phi$. The asymptotic subrank or asymptotic induced matching number of the k -graph Φ is defined as

$$Q(\Phi) := \lim_{n \rightarrow \infty} Q(\Phi^{2n})^{1/n}.$$

This limit exists and equals the supremum $\sup_{\psi \in \mathcal{H}} Q(\Phi^{2n})^{1/n}$ by Fekete's lemma (see, e.g., [NS08, No. 98]).

We study the following basic question:

Problem 1.1. Given Φ what is the value of $Q(\Phi)$?

A priori, for $\Phi \in \mathcal{H}_k$ we have the upper bound $Q(\Phi) \leq \min_i |V_i|$ and therefore holds that $Q(\Phi) \leq \min_i |V_i|$, since $|V^{2n}| = |V|^{2n}$.

Problem 1.1 has been studied for several families of k -graphs, in several different contexts: the cap set problem [RE17, Tao18, KSS18, Noo18, Pao18], approaches to fast matrix multiplication [Str01, BCU⁺17a, BCU⁺17b, Saw17], arithmetic removal lemmas [SIS18, FLS18], property testing [FK14, HX17], quantum information theory [VUC15, VC17], and the general study of asymptotic properties of tensors [SIS18, CVZ18a, CVZ18b]. We finally mention the related result of Razborov and Semerád [RS18] which says that the largest subset $E \subseteq \binom{[n]}{k}$ such that $E \times E \cap \{(a, b), (b, c), (c, a)\} : a, b, c \in [n]\}$ is a matching, has size $n^{2-\epsilon(n)}$ when n goes to infinity [RS18]; see also [AS06, Equation 2].

1.2. Result. We solve Problem 1.1 for a family of k -graphs that are structured but nontrivial. For $k \geq 2$ let $\lambda = (\lambda_1, \dots, \lambda_k) \vdash k$ be an integer partition of k with n nonzero parts, that is, $\lambda_i \geq \lambda_j \geq \dots \geq \lambda_n > 0$ and $\sum_{i=1}^k \lambda_i = k$. We define the k -graph

$$\Phi_\lambda := \{x \in [n]^k : \text{type}(x) = \lambda\}$$

where the expression $\text{type}(x) = \lambda$ means that x is a permutation of the k -tuple

$$(\underbrace{1, \dots, 1}_{\lambda_1}, \underbrace{2, \dots, 2}_{\lambda_2}, \dots, \underbrace{n, \dots, n}_{\lambda_n}).$$

For example, the partition $\lambda = (1, 1) \vdash 2$ corresponds to the 2-graph

$$\Phi_{(1,1)} = \{(2, 1), (1, 2)\} \subseteq [2] \times [2]$$

and the partition $\lambda = (2, 2) \vdash 4$ corresponds to the 4-graph

$$\Phi_{(2,2)} = \{(2, 2, 1, 1), (2, 1, 2, 1), (2, 1, 1, 2), (1, 2, 2, 1), (1, 2, 1, 2), (1, 1, 2, 2)\} \subseteq [2]^4.$$

It was shown in [CVZ18a] that $Q(\Phi_{(k-1,1)}) = 2^{H(1-(k-1)/k)}$ for every $k \in \mathbb{N}_{\geq 2}$ where H is the Shannon entropy in base 2. As a natural continuation of that work we study $Q(\Phi_{(\lambda_1, 2\lambda_2/2)})$ for even $k \in \mathbb{N}$. Since $\Phi_{(\lambda_1, 2\lambda_2/2)} \subseteq [2]^{2\lambda_2}$ we have $Q(\Phi_{(\lambda_1, 2\lambda_2/2)}) \leq 2$. Clearly, the 2-graph $\Phi_{(1,1)}$ is itself a matching, and so $Q(\Phi_{(1,1)}) = 2$. It was shown in [CVZ18a] that also $Q(\Phi_{(2,2)}) = 2$. Our new result is the following extension:

Theorem 1.2. Let $k \in \mathbb{N}_{\geq 2}$ be even and large enough. Then $Q(\Phi_{(\lambda_1, 2\lambda_2/2)}) = 2$.

In other words, we prove that for every large enough even $k \in \mathbb{N}_{\geq 2}$ there is an induced matching $\Psi \subseteq \Phi_{(\lambda_1, 2\lambda_2/2)}$ of size $|\Psi| = 2^{o(k)}$ when n goes to infinity.

Moreover, we numerically verified that $Q(\Phi_{(\lambda_1, 2\lambda_2/2)}) = 2$ also holds for all even $k \leq 2000$. We conjecture that $Q(\Phi_{(\lambda_1, 2\lambda_2/2)}) = 2$ for all even k . More generally, we conjecture (cf. [VC15] and [CVZ18a, Question 1.3.3]) that $\log_2 Q(\Phi_\lambda)$ equals the Shannon entropy of the probability distribution obtained by normalising the partition λ . We will discuss further motivation and background in Section 1.4.

$f_1(x+y) = 1$. Denote by T-FREE the set of function triples that are triangle-free, and the distance of a function triple to T-FREE is defined as

$$\text{dist}((f_1, f_2, f_3), \text{T-FREE}) := \min_{(g_1, g_2, g_3) \in \text{T-FREE}} \text{dist}(f_1, g_1) + \text{dist}(f_2, g_2) + \text{dist}(f_3, g_3).$$

As the following reduction and Theorem 2 shows, the multiple-function and single-function case are essentially equivalent.²

Lemma 1 (Xie, 2010). Given any function triple $f_1, f_2, f_3 : \mathbb{F}_2^k \rightarrow \{0, 1\}$ which is ϵ -far from T-FREE and contains N triangles, there is a single function $f : \mathbb{F}_2^{2k} \rightarrow \{0, 1\}$ which is $\frac{1}{3}\epsilon$ -far from triangle-freeness and contains N triangles.

Proof. Construct f as follows. For each $x \in \mathbb{F}_2^k$, denote by (a, b, x) the $(k+2)$ -dimension vector whose last k coordinates are given by x . For each $x \in \mathbb{F}_2^k$, let $f(0, 0, x)$ be $f_1(x)$, $f(0, 1, x)$ be $f_2(x)$, and $f(1, 1, x)$ be $f_3(x)$. It is easy to see that any triangle in f has to have its three "vertices" given by entries from f_1, f_2 and f_3 , respectively. The lemma follows immediately. \square

The canonical tester is the naive-looking algorithm that samples $x, y \in \mathbb{F}_2^k$ uniformly at random and returns YES if $f(x) = f(y) = f(x+y) = 1$ and NO otherwise. A tester is said to be one-sided if, whenever the input satisfies the property in question, it outputs YES with probability 1. By the following theorem, it is without loss of generality to consider obfuscating the canonical tester.

Theorem 2 (Bhattacharyya and Xie, 2010). Suppose there is a one-sided tester for T-FREE has query complexity $q(\epsilon)$, then the canonical tester has query complexity at most $O(q^2(\epsilon))$. This holds for both the single-function case (when $f_1 = f_2 = f_3$) and the multiple-function case.

Definition 1 (Perfect-Matching-Free (PMF) Families of Vectors). Let k and m be integers such that $0 < k < m < 2^k$. A (k, m) perfect-matching-free (PMF) family of vectors is a set of vectors $(a_i, b_i, c_i)_{i \in [m]}$, where $a_i, b_i, c_i \in \mathbb{F}_2^k$ and $c_i = a_i + b_i$ for all $i \in [m]$, such that for any permutation triple $\pi_1, \pi_2, \pi_3 \in \text{Sym}([m])$, either $\pi_1 = \pi_2 = \pi_3$, or there exists an $i \in [m]$ such that $a_{\pi_1(i)} + b_{\pi_2(i)} \neq c_{\pi_3(i)}$.

One can permute and then concatenate all a_i 's in a (k, m) PMF family and obtains $m!$ vectors in \mathbb{F}_2^m , the same can be done for b_i 's and c_i 's. By the property of PMF, each new vector obtained from a_i 's forms one and only one triangle with two other vectors obtained from b_i 's and c_i 's, respectively, and they are obtained through exactly the same permutation on $[m]$. This means that to remove all $m!$ triangles in the system, one has to remove at least the same number of vectors. This large ratio between the distance to triangle-freeness and the number of triangles is exactly what is needed to obfuscate a tester. One may go further and take multiple copies of a PMF family and repeats this experiment. An asymptotic calculation would give the following theorem.

Theorem 3 (Bhattacharyya and Xie, 2010). If (k, m) PMF family of vectors exists, then for small enough ϵ and large enough k , there exists a function triple $f_1, f_2, f_3 : \mathbb{F}_2^k \rightarrow \{0, 1\}$ that is ϵ -far from triangle-freeness, but the canonical tester needs $\Omega(\frac{1}{\epsilon})^m$ queries to detect a triangle, where $\alpha = (2 - \frac{\log 3}{\log 2}) / (1 - \frac{\log 3}{\log 2})$.

²We acknowledge Xie (2010) for informing us of the possibility of this reduction.
³All logarithms in this paper are base 2.

Figure 3.28: A visualisation of (Overlap) extraction from grobid

Line-Based Extraction

Contenu du chapitre

4.1	Introduction to the Task	45
4.2	Related Work	47
4.2.1	Computer Vision: Object Detector	47
4.2.2	Natural language Processing: Transformer-Based	48
4.2.3	Styling and Sequence: CRF based	49
4.3	Experimental Setup and Results	49
4.4	Discussion of the Results	55
4.4.1	Conclusions	55
4.4.2	Limitations	57

4.1 Introduction to the Task

Our DocEng 2021 paper [MPS21], that we describe and expand upon in this chapter, aimed to build on the robust groundwork laid by Lucas, streamlining the process of simplifying mathematical results and employing deep learning techniques to scout for effective architectures that yield high performance. Lucas’s approach, which boasted a 72.4% macro F_1 score on a 10-class classification task, set a strong precedent, see Table 4.1.

	Precision	Recall	F_1 -Score
Assumption	0.000	0.000	0.000
Claim	0.628	0.839	0.718
Conjecture	0.903	0.629	0.742
Corollary	0.874	0.847	0.860
Definition	0.835	0.775	0.804
Lemma	0.901	0.874	0.888
Proof	0.781	0.834	0.806
Proposition	0.848	0.828	0.838
Remark	0.714	0.691	0.702
Theorem	0.886	0.872	0.879
Macro avg	0.737	0.719	0.724
Weighted avg	0.800	0.834	0.816

Table 4.1: Results on Lucas’s test set of line-based CRF. It took 5 hours to train and has 37K active features (over 72K). Parameters: minimum frequency of 50, l_1 -regularization of 5.0, l_2 -regularization of 0.1, 300 iterations.

Taking over the TheoremKB project, I continued from where Lucas left off, heeding his recommendation for a deep-learning-based methodology to approach the problem of information extraction. This shift to

	Precision	Recall	F ₁ -Score	Support
Assumption	0.000	0.000	0.000	283
Claim	0.818	0.542	0.652	17031
Conjecture	0.883	0.356	0.508	2825
Corollary	0.882	0.596	0.711	32371
Definition	0.747	0.649	0.695	140873
Lemma	0.892	0.793	0.840	173266
Proof	0.701	0.893	0.786	1723296
Proposition	0.833	0.632	0.719	34270
Remark	0.775	0.252	0.380	28520
Theorem	0.836	0.786	0.810	138122
Macro avg	0.737	0.550	0.610	2290857
Weighted avg	0.733	0.844	0.777	2290857

Table 4.2: Results on Lucas’s test set of word-based CRF. It took 21 hours to train and has 350K active features (over 800K). Parameters: no minimum frequency, l1-regularization of 0.3, l2-regularization of 0.1, 500 iterations.

deep learning was not merely a different approach but promised to capture complex dependencies, offering numerous advantages such as:

1. The paper advances beyond the limitations of Lucas’s methodology, which relied on hard-coded features incapable of capturing semantic meaning. In contrast, at the time of writing, deep learning—particularly in the NLP field—was already renowned for its semantic understanding, which offers potential benefits in a transfer learning context, eliminating the need for manually engineered features.
2. The incorporation of deep learning models presents the opportunity to significantly enhance the accuracy of classification tasks in comparison to traditional, simpler machine learning models.
3. Adopting a deep learning approach allows for integrating and exploring diverse modalities, extending beyond NLP to include computer vision techniques. Such an approach can uncover valuable insights, particularly in detecting and correcting textual extraction errors that may have persisted in Lucas’s methodology.

Our decision to explore the problem using a deep learning framework was driven by the potential of leveraging both text and vision modalities. While Lucas’s methodology integrated handcrafted rules and considered font and sequential information from previous lines, our approach aimed to harness a broader spectrum of information. The intention was not to outperform Lucas’s method but to investigate whether deep learning could achieve comparable results. However, at the time of composing this paper, we had not developed a sequential model (to capture paragraph context) reliable enough to fairly compete with Lucas’s strategy of using CRF to map long term paragraph dependencies.

Before model implementation, we observed that categories such as lemma, corollary, remark, and definition in Lucas’s table could be grouped under the broader term ‘Theorem.’ Simplifying these into a single ‘theorem’ class could streamline the process, making it easier to understand and interpret results. Detecting the ‘theorem’ class could pave the way for a secondary classifier to determine the specific subclass. This approach allows for a more straightforward methodology while still recognising and categorising multiple subclass distinctions.

When writing the paper, given the popularity of object detection frameworks and the availability of bounding box coordinates from pdfalto, I considered adopting these methods. Utilising a visual approach could bypass the need for tools like pdfalto or Groid at inference, allowing direct prediction of bounding boxes on bitmap images along with detection scores of the content. This method also reveals if features such as bold or italic text are detectable, offering further insights.

The second deep learning approach I proposed was akin to Lucas’s but incorporated NLP to grasp semantic meaning. Like Lucas’s methodology, we extracted text from each PDF line using pdfalto. We then processed these lines independently (ignoring sequential information like previous line labels) through

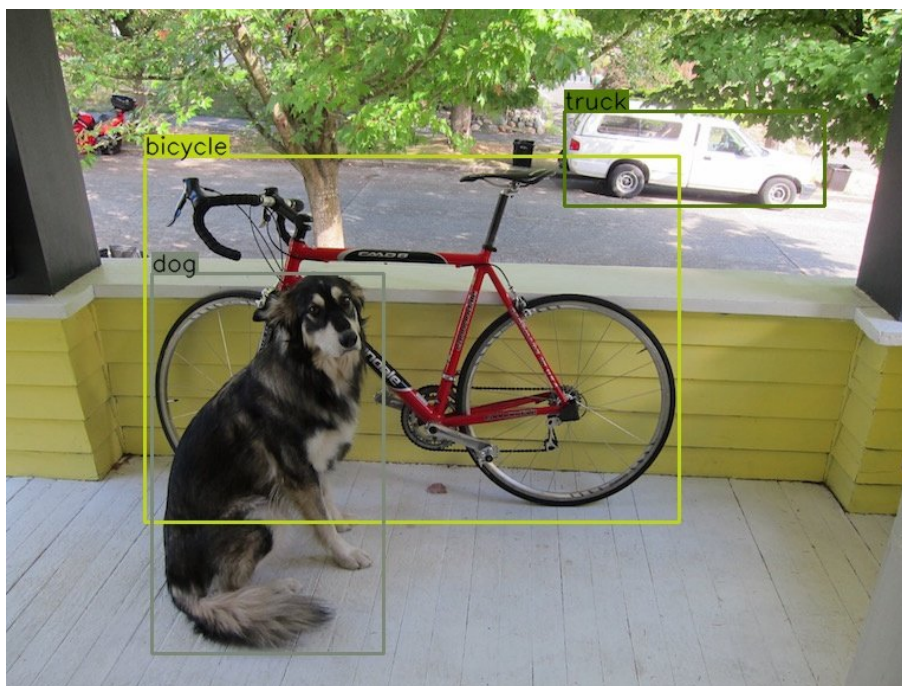


Figure 4.1: A typical Object Detection example [LMB⁺14]

a language model. This model was fine-tuned to perform a three-class classification task, specifically identifying basic, proof, and theorem categories.

The goal was to fine-tune an encoder-based language model to perform a classification task. This approach aimed to utilise the linguistic capabilities of such models for precise categorisation.

4.2 Related Work

4.2.1 Computer Vision: Object Detector

Utilising an Object Detection framework could identify specific regions of interest and label paragraphs accordingly. This approach might eliminate the need for tools like Grobid and pdfto, as the framework would allow direct feeding of bitmap images of the PDF into the model at inference time.

While many object detection frameworks are widely used for identifying everyday objects, such as those demonstrated in the MS COCO dataset [LMB⁺14], assessing their practical applicability to scientific documents is crucial. This involves evaluating how effectively these frameworks can be adapted to scientific text and imagery’s unique characteristics and requirements.

Two significant factors differentiate the application of object detection frameworks like those trained on the MSCOCO dataset for use in scientific documents:

1. **Pixel Value Transition:** In typical object detection scenarios, such as those trained on MSCOCO, the models capitalise on distinct transitions between the background and foreground pixel values. However, in scientific documents, the background and foreground are often visually similar, with only the initial characters potentially indicating the paragraph’s label. This similarity poses a challenge as standard object detectors focus on visual cues rather than textual content.
2. **Spacing and Structure:** Scientific documents usually feature large spacings between paragraphs, unlike typical images that might contain multiple overlapping objects. This structural difference requires a tailored approach, as standard object detectors are designed to handle intersecting objects within a single frame, not discrete blocks of text separated by significant whitespace.

Despite the atypical application of object detection models in document AI, some research has successfully adapted these technologies for document analysis. For instance, the work cited as [6] utilises CNNs and a hybrid CRF model to detect tables, lines, and formulas within documents. This approach

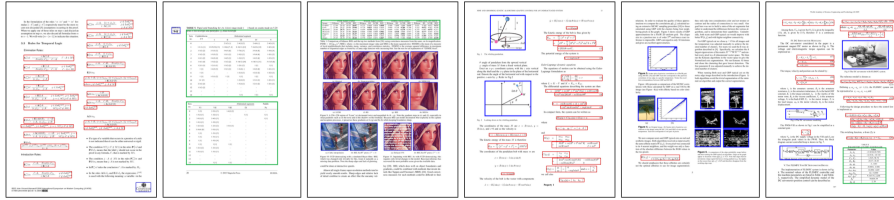


Figure 4.2: Object Detection (formula, table, figures) in Scientific articles [LYL18]

frames these elements as objects in an object detection task, showcasing the flexibility and potential of object detection models in handling structured document content.

This work is significant as the object detectors tested demonstrated high accuracy, around 90%, with an IOU of 0.6, inspiring us to apply these models to our specific needs—distinguishing proofs from theorems. This task presents unique challenges since proofs and theorems do not exhibit clear visual distinctions, unlike figures and formulas used in previous studies, adapting these models to our context is more complex.

During our experimentation phase in February 2021, object detection algorithms such as R-CNN[GDDM14], Fast R-CNN[Gir15], Faster R-CNN[RHGS17], and Yolo[RDGF16] were prevalent. Notably, Yolo4[BWL20] was recently released, showing significant improvements in accuracy and efficiency over its predecessors, measured in frames per second (FPS). Therefore, we chose Yolo4 as our preferred object detector for its superior performance capabilities.

Earlier versions of Yolo utilised the Darknet53 as the backbone network with residual layers to manage issues like gradient vanishing and internal covariance shifts. Yolo4, however, integrates advanced techniques that substantially enhance network performance. These are categorised into "*bag of freebies*"—techniques like Cutmix, DropBlock, Mosaic Data Augmentation, and label smoothing that improve model performance without affecting inference time—and "*bag of specials*" such as Mish activation and Cross-Stage Partial Connections slightly increase inference time but significantly boost detection accuracy.

4.2.2 Natural language Processing: Transformer-Based

Utilising natural language processing to determine the labels of each line extracted via pdfalto could effectively harness the text-based modality, focusing on the semantic meaning of the text. This approach aims to leverage the inherent textual information to improve classification accuracy.

Textual information from each line, extracted using pdfalto, is used to set up a classification task focusing on predicting categories such as Theorem, Proof, and Basic. We evaluated the performance of several autoencoding-based models like Bert base, DistilBERT base, and SciBERT[BLC19] base. These models are mainly chosen for their proficiency in classification tasks due to their ability to encode semantic meaning effectively, unlike autoregressive models like GPT, which are more suited for text generation tasks. This selection underpins our strategy to leverage semantic analysis for improved classification accuracy.

Among the three models evaluated, BERT [DCLT19] provides a standard transformer architecture trained on a 16GB English corpus, making it versatile for general tasks. SciBERT [BLC19], however, is specialised and trained in scientific corpora, including biomedical and computer science texts, making its vocabulary and learning more tailored to scientific tasks. DistilBERT[SDCW19] seeks to replicate BERT's performance through knowledge distillation, acting as a '*student*' to BERT's '*teacher*.' This results in a model with fewer parameters and lower associated inference costs.

In contrast to the transformer-based models, we also developed a simple LSTM model utilising parallel positional encoding with 100 cells. This LSTM [HS97] model serves as a baseline for comparison, focusing on classifying proofs and theorems. It processes inputs as numerical representations of tokens generated by a custom tokeniser that assigns unique IDs to the top 50k tokens in the corpus. Each line is padded to 50 tokens to standardise input dimensions for the LSTM network. This setup allows us to evaluate the relative effectiveness of LSTM against advanced transformer models.

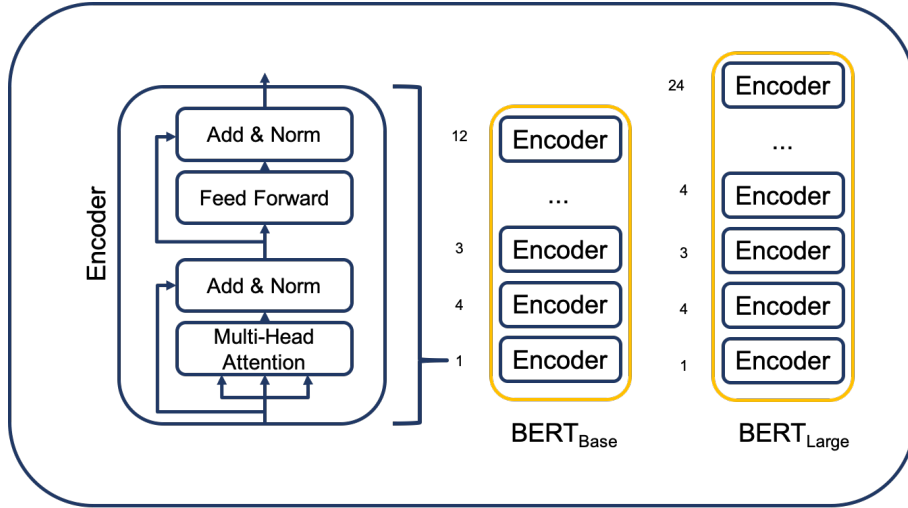


Figure 4.3: Bert Architecture (Encoder block, size)

4.2.3 Styling and Sequence: CRF based

In our styling- and sequence-based approaches, we utilise pdfalto output to extract features from the document is organised into various hierarchical levels. These include page-level features capturing the document’s global position (beginning, inside, end), text block-level features that compute the distances and horizontal paddings between blocks and text line-level features that identify tabulations and recurring text patterns like those found in headers or footers. Additionally, string-level features assess the normalised word, spacing, font style (italic or bold), font size, word position within the sentence, and the presence of digits and special characters. These features are normalised and standardised across the document to allow the model to concentrate on variations rather than fixed values, enhancing its robustness to stylistic changes. We also calculate the difference between current and adjacent token values to identify local changes effectively. To manage the resolution of the document’s analysis, we derive a linear sequence of features, copying broader, coarser-grained features for each line and aggregating finer-grained features using minimum, maximum, first, last, and mean values.

As we obtain a tagged linear sequence of features, it can be used to train a linear-chain CRF. This class of probabilistic graphical models is perfect for model sequences and has efficient inference algorithms. We use sklearn-crfsuite to perform the training, refer figure 4.4. However, when dealing with many features, the training time can remain high, and contrary to neural networks, it does not yet benefit from GPU acceleration. Feature selection methods (such as L1-regularization) is therefore mandatory to reduce training time and reduce overfitting.

Lastly, we implemented a straightforward baseline method that utilises basic document information and sequential data, known as the naïve algorithm. This approach involves searching for specific target words like "Proof" or "Theorem" in bold or italics at the start of a line and assigning labels to this line and all subsequent lines within the same block accordingly.

4.3 Experimental Setup and Results

We initiate the process by marking the ground truths using the labelling file (*extthm.sty*) compiled with the LaTeX source. This compilation injects markers into the PDF, denoting proofs and theorems as embedded links without URLs, pointing to their labels, see figure 4.5.

The marked PDF allows us to extract text lines and their associated labels using pdfalto, which generates an annot.xml file containing the labels (see figure 4.6) and a separate xml file with all the text lines (see figure 4.7).

The dataset consists of text lines from 4,400 PDFs, totalling 5,281,411 lines. After filtering out non-applicable rows, rows with low contours, page numbers, and duplicates, we retained 2,423,064 lines. We then created a balanced dataset of 1,064,955 lines, ensuring each labelled category, including the minority class ‘theorem,’ had an equal representation of 354,985 lines. This dataset was split into training (851,964 lines) and validation (212,991 lines) sets using an 80/20 ratio with a fixed random seed to

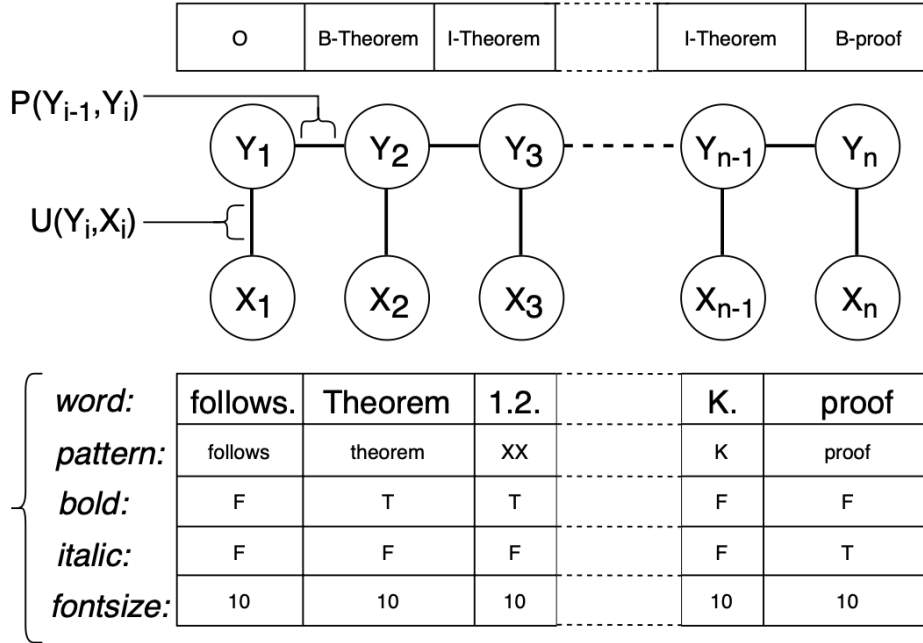


Figure 4.4: Visualisation of Styling and Sequential approach using handcrafted features applied to a linear-chain CRF.

Conjecture 1. Let f be a Boolean function. If m has at least two distinct prime factors, then

$$\deg(f) = O(\text{poly}(\deg_m(f))).$$

Figure 4.5: An example of annotated result (marked as a link - `uri:exthm.Conjecture.1`) in a pdf

```

▼<ANNOTATION subtype="Link" pagenum="2" id="p2_a5">
  ▼<ACTION type="uri">
    <DEST>uri:extthm.Conjecture.1</DEST>
  </ACTION>
  ▼<QUADPOINTS>
    ▼<QUADRILATERAL>
      <POINT HPOS="71.004" VPOS="371.111"/>
      <POINT HPOS="71.004" VPOS="359.422"/>
      <POINT HPOS="540.996" VPOS="371.111"/>
      <POINT HPOS="540.996" VPOS="359.422"/>
    </QUADRILATERAL>
  </QUADPOINTS>
</ANNOTATION>
▼<ANNOTATION subtype="Link" pagenum="2" id="p2_a6">
  ▼<ACTION type="uri">
    <DEST>uri:extthm.Conjecture.1</DEST>
  </ACTION>
  ▼<QUADPOINTS>
    ▼<QUADRILATERAL>
      <POINT HPOS="237.531" VPOS="396.225"/>
      <POINT HPOS="237.531" VPOS="383.324"/>
      <POINT HPOS="374.469" VPOS="396.225"/>
      <POINT HPOS="374.469" VPOS="383.324"/>
    </QUADRILATERAL>
  </QUADPOINTS>
</ANNOTATION>
▼<ANNOTATION subtype="Link" pagenum="2" id="p2_a7">
  ▼<ACTION type="uri">
    <DEST>uri:extthm.Conjecture.1</DEST>
  </ACTION>
  ▼<QUADPOINTS>
    ▼<QUADRILATERAL>
      <POINT HPOS="71.004" VPOS="420.127"/>
      <POINT HPOS="71.004" VPOS="408.438"/>
      <POINT HPOS="72.996" VPOS="420.127"/>
      <POINT HPOS="72.996" VPOS="408.438"/>
    </QUADRILATERAL>
  </QUADPOINTS>
</ANNOTATION>

```

Figure 4.6: Annotation (annot.xml) example obtained from pdftalio in a pdf

```

<TextBlock ID="p2_b7" HPOS="72.0000" VPOS="360.510" HEIGHT="9.6000" WIDTH="452.691">
  ▼<TextLine WIDTH="452.691" HEIGHT="9.6000" ID="p2_t27" HPOS="72.0000" VPOS="360.510">
    <String ID="p2_w356" CONTENT="Conjecture" HPOS="72.0000" VPOS="360.510" WIDTH="60.2281" HEIGHT="9.6000" STYLEREF="font37"/>
    <SP WIDTH="4.1782" VPOS="360.510" HPOS="132.228"/>
    <String ID="p2_w357" CONTENT="1." HPOS="136.406" VPOS="360.510" WIDTH="9.7571" HEIGHT="9.6000" STYLEREF="font37"/>
    <SP WIDTH="4.9847" VPOS="360.510" HPOS="146.163"/>
    <String ID="p2_w358" CONTENT="Let" HPOS="151.148" VPOS="360.543" WIDTH="14.9280" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9020" VPOS="360.543" HPOS="166.076"/>
    <String ID="p2_w359" CONTENT="f" HPOS="169.978" VPOS="360.543" WIDTH="5.3411" HEIGHT="9.5673" STYLEREF="font20"/>
    <SP WIDTH="5.0769" VPOS="360.543" HPOS="175.319"/>
    <String ID="p2_w360" CONTENT="be" HPOS="180.396" VPOS="360.543" WIDTH="9.4800" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9055" VPOS="360.543" HPOS="189.876"/>
    <String ID="p2_w361" CONTENT="a" HPOS="193.781" VPOS="360.543" WIDTH="5.5756" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.8945" VPOS="360.543" HPOS="199.357"/>
    <String ID="p2_w362" CONTENT="Boolean" HPOS="203.251" VPOS="360.543" WIDTH="37.2328" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9055" VPOS="360.543" HPOS="240.484"/>
    <String ID="p2_w363" CONTENT="function." HPOS="244.389" VPOS="360.543" WIDTH="42.3764" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="5.0182" VPOS="360.543" HPOS="286.766"/>
    <String ID="p2_w364" CONTENT="If" HPOS="291.784" VPOS="360.543" WIDTH="7.5524" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9002" VPOS="360.543" HPOS="299.336"/>
    <String ID="p2_w365" CONTENT="m" HPOS="303.237" VPOS="360.543" WIDTH="9.5782" HEIGHT="9.5673" STYLEREF="font20"/>
    <SP WIDTH="3.9038" VPOS="360.543" HPOS="312.815"/>
    <String ID="p2_w366" CONTENT="has" HPOS="316.719" VPOS="360.543" WIDTH="15.6120" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9055" VPOS="360.543" HPOS="332.331"/>
    <String ID="p2_w367" CONTENT="at" HPOS="336.236" VPOS="360.543" WIDTH="9.1996" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9055" VPOS="360.543" HPOS="345.436"/>
    <String ID="p2_w368" CONTENT="least" HPOS="349.341" VPOS="360.543" WIDTH="20.8997" HEIGHT="9.5673" STYLEREF="font23"/>
    <SP WIDTH="3.9055" VPOS="360.543" HPOS="370.241"/>
    <String ID="p2_w369" CONTENT="two" HPOS="374.146" VPOS="360.543" WIDTH="16.4477" HEIGHT="9.5673" STYLEREF="font23"/>
  </TextLine>

```

Figure 4.7: Text lines extracted from pdftalio (contains the text and font level information)


```

<TextStyle ID="font0" FONTFAMILY="cmbx10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font1" FONTFAMILY="cmr8" FONTSIZE="7.970" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font2" FONTFAMILY="cmccsc10" FONTSIZE="7.970" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font3" FONTFAMILY="cmccsc10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font4" FONTFAMILY="cmr10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font5" FONTFAMILY="cmtl10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font6" FONTFAMILY="cmr10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font7" FONTFAMILY="cmml10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font8" FONTFAMILY="cmr7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="subscript"/>
<TextStyle ID="font9" FONTFAMILY="cmsy10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font10" FONTFAMILY="cmml7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="superscript"/>
<TextStyle ID="font11" FONTFAMILY="xyatip" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="fixed" FONTCOLOR="000000"/>
<TextStyle ID="font12" FONTFAMILY="xybtip" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="fixed" FONTCOLOR="000000"/>
<TextStyle ID="font13" FONTFAMILY="cmsy7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font14" FONTFAMILY="freeserifitalic" FONTSIZE="10.435" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="italics"/>
<TextStyle ID="font15" FONTFAMILY="freeserif" FONTSIZE="10.435" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font16" FONTFAMILY="cmml7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="subscript"/>
<TextStyle ID="font17" FONTFAMILY="cmml5" FONTSIZE="4.981" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="subscript"/>
<TextStyle ID="font18" FONTFAMILY="msbm10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font19" FONTFAMILY="cmex10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font20" FONTFAMILY="cmi7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font21" FONTFAMILY="msam10" FONTSIZE="9.963" FONTTYPE="sans-serif" FONTWIDTH="fixed" FONTCOLOR="000000"/>
<TextStyle ID="font22" FONTFAMILY="freeserif" FONTSIZE="10.827" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font23" FONTFAMILY="freeserifitalic" FONTSIZE="10.827" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="italics"/>
<TextStyle ID="font24" FONTFAMILY="freeserif" FONTSIZE="7.038" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="subscript"/>
<TextStyle ID="font25" FONTFAMILY="freeserifitalic" FONTSIZE="10.827" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="italics superscript"/>
<TextStyle ID="font26" FONTFAMILY="freeserif" FONTSIZE="7.038" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font27" FONTFAMILY="freeserif" FONTSIZE="10.827" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="superscript"/>
<TextStyle ID="font28" FONTFAMILY="freeserif" FONTSIZE="63.492" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font29" FONTFAMILY="cmsy7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="subscript"/>
<TextStyle ID="font30" FONTFAMILY="freeserif" FONTSIZE="49.785" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font31" FONTFAMILY="freeserif" FONTSIZE="11.368" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000"/>
<TextStyle ID="font32" FONTFAMILY="freeserifitalic" FONTSIZE="11.368" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="italics"/>
<TextStyle ID="font33" FONTFAMILY="freeserif" FONTSIZE="7.389" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="subscript"/>
<TextStyle ID="font34" FONTFAMILY="cmsy7" FONTSIZE="6.974" FONTTYPE="sans-serif" FONTWIDTH="proportional" FONTCOLOR="000000" FONTSTYLE="superscript"/>

```

Figure 4.8: An example of Font information present in the pdfalto

Table 4.3: Performance: Computer Vision

Approach	Loss	mAP/Accuracy
Yolov4	8.68	0.416

maintain a stratified distribution that preserves the label distribution.

We can utilise the annotation coordinates provided in Figure 4.6 to train the Yolo model. These annotations label the text lines within each box, allowing for practical training of Yolo/BERT. Additionally, Figure 4.7 details each text line with coordinate information and textual content. From pdfalto, specific font information (see Figure 4.8) can be extracted and subsequently used as features for training a Conditional Random Field (CRF) model.

We trained the Yolo model under two different settings, line-based detection (see figure 4.10) and annotation-based detection (see figure 4.9), but only reported the results of the line-based approach, see table 4.3.

This allows for direct comparison with NLP, styling, and sequence-based approaches also trained at the line level. The visualisation of the Yolo model’s loss function is shown in figure 4.11, with the Intersection Over Union (IOU) set at 0.7, mirrors the approach used in paper [LYL18] for table and object detection.

Extending training to a paragraph setting significantly improves performance, as further iterations lead to better generalisation. At the time of writing this paper, I lacked access to the Jean Zay Supercomputer and conducted all training sessions on a Tesla P100 GPU via Google Colab.

We fine-tuned various NLP classifiers on a balanced dataset, testing both cased and uncased settings. We also implemented a simple LSTM model as a benchmark to evaluate the performance of transformer models, see table 4.4. The performance of all the models are quite similar even though they vary significantly in the pretraining data and network size in terms of parameters.

We now present the performance scores of the CRF model trained at the line level, see table 4.5. Unlike the language models, the CRF model leverages sequential and styling information, including details about page breaks obtained from pdfalto across multiple lines. This integration of additional contextual data from the document layout allows the CRF model to perform with a nuanced understanding of the document’s structure.

While each approach aims to capture unique aspects of its learning, it is essential to acknowledge that different models and modalities vary significantly in training times. These differences stem from the number of parameters each model has, memory usage (notably, attention mechanisms can be resource-intensive), and the overall network architecture, see table 4.6. Understanding these factors is crucial for effectively deploying these classifiers in practical settings, as they directly impact the efficiency and feasibility of model training and operation¹.

¹All training was done on a Tesla P100 GPU on Google Colaboratory with 15 GB of memory except for the CRF, trained on commodity hardware.

proof: 0.60
 T with parameter $\epsilon = \delta/2$, which takes our instance \mathcal{J} and outputs unweighted instance \mathcal{G} . We can apply the α approximation algorithm on \mathcal{G} to obtain some assignment ζ for which

$$\frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{G})} \geq \alpha.$$

Then, since $\text{Opt}(\mathcal{G}) \geq \gamma$, for the same assignment ζ we have

$$\frac{\text{Val}(\mathcal{F})}{\text{Opt}(\mathcal{F})} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G}) + \epsilon} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G})} \geq (1 - \epsilon/\gamma) \geq \frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{G})} - \frac{\epsilon}{\gamma \text{Opt}(\mathcal{G})} \geq \alpha - \frac{\epsilon}{\gamma} \geq \alpha - \delta,$$

which proves the statement of the theorem. \square

The argument from the previous theorem does not work for Min-CSPs, since in this case $\text{Opt}(\mathcal{G})$ can be arbitrarily small. Analogous claim for Min-CSPs was already proved in [16, Lemma 3.11] by using scaling techniques [14, 16]. For the sake of completeness, we give here somewhat more involved proof of this claim, using essentially the same techniques.

Theorem 9. Consider a Min-CSP Λ , and assume we can approximate the optimal value of unweighted instances within a multiplicative factor α . Then, for every $\delta \in (0, 1)$, weighted instances of the Min-CSP Λ can be approximated within a constant $\alpha + \delta$.

proof: 0.52 decision version of CSP Λ , in which we ask whether there is an assignment λ to the variables such that all the constraints of Λ are not satisfied. By Schaefer's dichotomy theorem [21], the problem of deciding whether there is an assignment which falsifies all the constraints is either NP-hard or in P. If solving this problem is NP-hard, then both weighted and unweighted versions of Min-CSP Λ are obviously NP-hard to approximate within any constant. Therefore, without loss of generality, we assume that deciding whether all constraints can be falsified is in P for Λ .

Hence, given an instance \mathcal{F} of the Min-CSP Λ , we can check in polynomial time whether $\text{Opt}(\mathcal{F}) = 0$. In case $\text{Opt}(\mathcal{F}) = 0$, we have found an optimal assignment, so it only remains to consider $\text{Opt}(\mathcal{F}) > 0$.

Without loss of generality let us assume that the weights of constraints $\{w_i\}_{i \in [m]}$ are sorted in descending order, i.e. $w_1 \geq w_2 \geq \dots \geq w_m$. We can find in polynomial time the largest $k \geq 1$ such that there is an assignment falsifying constraints C_1, C_2, \dots, C_{k-1} .

For this chosen k at least one of C_1, \dots, C_k will be true in any assignment, so we have that $\text{Opt}(\mathcal{F}) \geq w_k$. Also, since there is an assignment falsifying the first $k-1$ constraints, we have that $\text{Opt}(\mathcal{F}) \leq \sum_{i=k}^m w_i$.

Let us partition the constraints C_i into the following three groups:

- light: constraints C_i with weight $w_i \leq w_k/m^2$,
- medium: constraints C_i with weight $w_k/m^2 < w_i < w_k/m$,
- heavy: constraints C_i with weight $w_k/m \leq w_i$.

Then, we create an instance \mathcal{F}' by adding medium and heavy constraints C_i from \mathcal{F} . Furthermore, we scale down the weights of heavy constraints to w_k/m^2 in \mathcal{F}' . Finally, we normalize the weights to total weight by multiplying them by some factor $\alpha \geq 1$. Note that $\text{Opt}(\mathcal{F}') \geq \alpha w_k$, since \mathcal{F}' still has (although with different weights) constraints C_1, C_2, \dots, C_k . Hence, in a completely analogous manner to Theorem 8, we can use the algorithm from Lemma 7 with $\epsilon = \frac{\delta \alpha w_k}{4}$ to get an assignment ζ which gives us an $\alpha + \delta/2$ approximation of the optimal value for \mathcal{F}' . Finding the $\alpha + \delta/2$ approximation can be performed in polynomial time, because $\text{opt}(\mathcal{F}') \geq \alpha w_k$. Let us now see how well ζ approximates the optimal value of \mathcal{F} . We have that following two properties:

proof: 0.36
 T with parameter $\epsilon = \delta/2$, which takes our instance \mathcal{J} and outputs unweighted instance \mathcal{G} . We can apply the α approximation algorithm on \mathcal{G} to obtain some assignment ζ for which

$$\frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{G})} \geq \alpha.$$

Then, since $\text{Opt}(\mathcal{G}) \geq \gamma$, for the same assignment ζ we have

$$\frac{\text{Val}(\mathcal{F})}{\text{Opt}(\mathcal{F})} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G}) + \epsilon} \geq \frac{\text{Val}(\zeta) - \epsilon}{\text{Opt}(\mathcal{G})} \geq (1 - \epsilon/\gamma) \geq \frac{\text{Val}(\zeta)}{\text{Opt}(\mathcal{G})} - \frac{\epsilon}{\gamma \text{Opt}(\mathcal{G})} \geq \alpha - \frac{\epsilon}{\gamma} \geq \alpha - \delta,$$

which proves the statement of the theorem. \square

The argument from the previous theorem does not work for Min-CSPs, since in this case $\text{Opt}(\mathcal{G})$ can be arbitrarily small. Analogous claim for Min-CSPs was already proved in [16, Lemma 3.11] by using scaling techniques [14, 16]. For the sake of completeness, we give here somewhat more involved proof of this claim, using essentially the same techniques.

Theorem 9. Consider a Min-CSP Λ , and assume we can approximate the optimal value of unweighted instances within a multiplicative factor α . Then, for every $\delta \in (0, 1)$, weighted instances of the Min-CSP Λ can be approximated within a constant $\alpha + \delta$.

proof: 0.45 decision version of CSP Λ , in which we ask whether there is an assignment λ to the variables such that all the constraints of Λ are not satisfied. By Schaefer's dichotomy theorem [21], the problem of deciding whether there is an assignment which falsifies all the constraints is either NP-hard or in P. If solving this problem is NP-hard, then both weighted and unweighted versions of Min-CSP Λ are obviously NP-hard to approximate within any constant. Therefore, without loss of generality, we assume that deciding whether all constraints can be falsified is in P for Λ .

Hence, given an instance \mathcal{F} of the Min-CSP Λ , we can check in polynomial time whether $\text{Opt}(\mathcal{F}) = 0$. In case $\text{Opt}(\mathcal{F}) = 0$, we have found an optimal assignment, so it only remains to consider $\text{Opt}(\mathcal{F}) > 0$.

Without loss of generality let us assume that the weights of constraints $\{w_i\}_{i \in [m]}$ are sorted in descending order, i.e. $w_1 \geq w_2 \geq \dots \geq w_m$. We can find in polynomial time the largest $k \geq 1$ such that there is an assignment falsifying constraints C_1, C_2, \dots, C_{k-1} .

For this chosen k at least one of C_1, \dots, C_k will be true in any assignment, so we have that $\text{Opt}(\mathcal{F}) \geq w_k$. Also, since there is an assignment falsifying the first $k-1$ constraints, we have that $\text{Opt}(\mathcal{F}) \leq \sum_{i=k}^m w_i$.

Let us partition the constraints C_i into the following three groups:

- light: constraints C_i with weight $w_i \leq w_k/m^2$,
- medium: constraints C_i with weight $w_k/m^2 < w_i < w_k/m$,
- heavy: constraints C_i with weight $w_k/m \leq w_i$.

Then, we create an instance \mathcal{F}' by adding medium and heavy constraints C_i from \mathcal{F} . Furthermore, we scale down the weights of heavy constraints to w_k/m^2 in \mathcal{F}' . Finally, we normalize the weights to total weight by multiplying them by some factor $\alpha \geq 1$. Note that $\text{Opt}(\mathcal{F}') \geq \alpha w_k$, since \mathcal{F}' still has (although with different weights) constraints C_1, C_2, \dots, C_k . Hence, in a completely analogous manner to Theorem 8, we can use the algorithm from Lemma 7 with $\epsilon = \frac{\delta \alpha w_k}{4}$ to get an assignment ζ which gives us an $\alpha + \delta/2$ approximation of the optimal value for \mathcal{F}' . Finding the $\alpha + \delta/2$ approximation can be performed in polynomial time, because $\text{opt}(\mathcal{F}') \geq \alpha w_k$. Let us now see how well ζ approximates the optimal value of \mathcal{F} . We have that following two properties:

Figure 4.9: Yolov4 in a paragraph-based setting

Figure 4.10: Yolov4 in a line-based setting

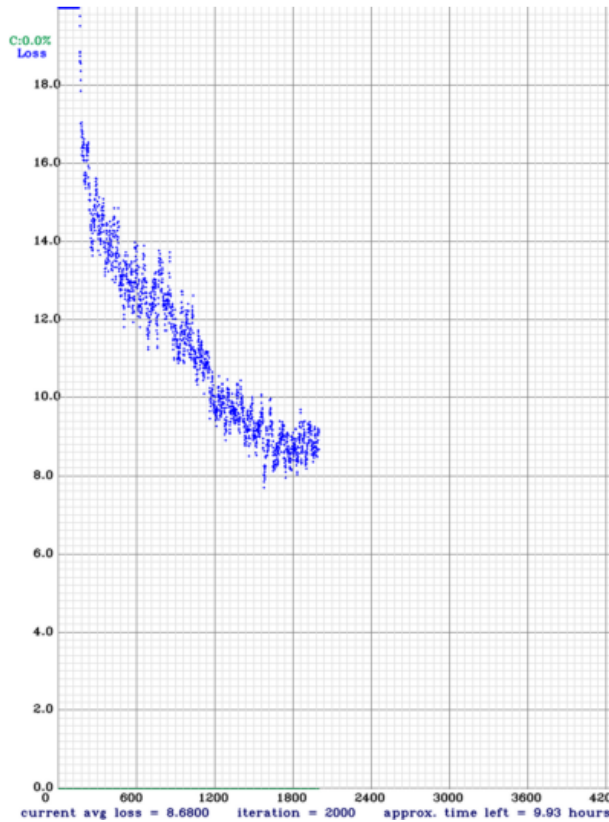


Figure 4.11: Visualisation of Loss function for the Yolo model (line-based, 2000 iterations)

proof: 0.51
 $|VMPSet^*(a, b)| = |VMPSet(a, b)| + |VMPSet^*(a, b)|.$

Since every iteration doubles the length of *VMPs* represented in the *VMPSet*s, after $O(\log(n))$ iterations the entire generating graph $\Gamma(n)$ is reduced to several disjoint sets of *VMPSet*(1, n) representing all *CVMPs*. Summation over all $|VMPSet(1, n)|$ that have $ER = \emptyset$ then gives the total number *CVMPs* and likewise perfect matchings. The summation step is what necessitates keeping *ER* the same for all *VMPSet*s.

Since the problem of counting perfect matchings in any bipartite graph is $\#P$ -complete and the class of parallel algorithms running in $(\log n)^{O(1)}$ on a polynomial number of processors, which is analogous to and commonly referred to as *NC*, is a subset of *FP*, the algorithm Aslam purports to have will be able to solve any $\#P$ problem in polynomial time, meaning $\#P = FP$ and $NP = P$.

2 Refutation

The main flaw in Aslam's reasoning is that the *ER* of every *CVMP* can be preserved during decomposition and subsequent reduction operations over *VMPSet*s. However, the *ER* of a *VMPSet* does not capture the *SE* of the *VMPs* it contains, and ultimately *SE* will determine *ER*. We will show that multiplication and addition of *VMPSet*s does not always give a *VMPSet* with the same *ER* for all *VMPs*. As proof this problem is inherent in all sufficient **theorem: 0.72** apps we give a counter-example in Section 3.

Lemma 1. The product *VMPSet AC* found from multiplying two *VMPSet*s $A = VMPSet(a, b)$ and $C = VMPSet(b, c)$ must be a single set and contain only *VMPs* with the same *ER*.

proof: 0.31 same *ER*.

Proof. The condition on *ER* follows from the definition of *VMPSet* and the inductive result of the algorithm itself. Since after the final iteration all *VMPSet*s with $ER = \emptyset$ will be counted, if multiplication resulted in a *VMPSet* containing some *CVMPs* with $ER = \emptyset$ (valid perfect matchings) and some with $ER \neq \emptyset$ (invalid), then summation would result in an incorrect number of perfect matchings. \square

6

$$|VMPSet^*(a, b)| = |VMPSet(a, b)| + |VMPSet^*(a, b)|.$$

Since every iteration doubles the length of *VMPs* represented in the *VMPSet*s, after $O(\log(n))$ iterations the entire generating graph $\Gamma(n)$ is reduced to several disjoint sets of *VMPSet*(1, n) representing all *CVMPs*. Summation over all $|VMPSet(1, n)|$ that have $ER = \emptyset$ then gives the total number *CVMPs* and likewise perfect matchings. The summation step is what necessitates keeping *ER* the same for all *VMPSet*s.

Since the problem of counting perfect matchings in any bipartite graph is $\#P$ -complete and the class of parallel algorithms running in $(\log n)^{O(1)}$ on a polynomial number of processors, which is analogous to and commonly referred to as *NC*, is a subset of *FP*, the algorithm Aslam purports to have will be able to solve any $\#P$ problem in polynomial time, meaning $\#P = FP$ and $NP = P$.

2 Refutation

The main flaw in Aslam's reasoning is that the *ER* of every *CVMP* can be preserved during decomposition and subsequent reduction operations over *VMPSet*s. However, the *ER* of a *VMPSet* does not capture the *SE* of the *VMPs* it contains, and ultimately *SE* will determine *ER*. We will show that multiplication and addition of *VMPSet*s does not always give a *VMPSet* with the same *ER* for all *VMPs*. As proof this problem is inherent in all sufficient **theorem: 0.65** apps we give a counter-example in Section 3.

Lemma 1. The product *VMPSet AC* found from multiplying two *VMPSet*s $A = VMPSet(a, b)$ and $C = VMPSet(b, c)$ must be a single set and contain only *VMPs* with the same *ER*.

proof: 0.88

Proof. The condition on *ER* follows from the definition of *VMPSet* and the inductive result of the algorithm itself. Since after the final iteration all *VMPSet*s with $ER = \emptyset$ will be counted, if multiplication resulted in a *VMPSet* containing some *CVMPs* with $ER = \emptyset$ (valid perfect matchings) and some with $ER \neq \emptyset$ (invalid), then summation would result in an incorrect number of perfect matchings. \square

6

Figure 4.13: Training Yolo on 6000 iterations

Figure 4.12: Training Yolo on 2000 iterations

Table 4.4: Performance: Natural Language

Approach	Loss	Accuracy (%)	Parameters (millions)
LSTM (unbalanced)	1.2571	41.83	1.65
LSTM	0.9620	46.25	1.65
BERT-base cased	0.8232	63.16	110
SciBERT-base cased	0.8165	63.27	110
DistilBERT-base cased	0.8267	63.02	66
Bert base-uncased	0.8222	63.13	110
SciBERT-base uncased	0.8151	63.57	110
DistilBERT-base uncased	0.8275	62.87	66

All results for natural language techniques were run on under-sampled versions of the input (to avoid class imbalance) unless "unbalanced" is stated. The learning rate used is $2 \cdot 10^{-5}$.

Table 4.5: Performance: Sequence and Styling

Approach	Precision	Recall	F ₁ (Weighted)	F ₁ (macro)
Linear-chain CRF	0.800	0.834	0.816	0.724
Naive Algorithm	0.832	0.370	0.487	0.561

Approach	Training time (h)	Model size (MB)
Yolov4	6.42 (2000 Epochs)	244
LSTM	0.05 (4 Epochs)	19.9
LSTM (unbalanced)	0.12 (4 Epochs)	19.9
DistilBERT-base cased	3.47 (1 Epoch)	263.3
SciBERT-base cased	6.07 (1 Epoch)	440.1
BERT-base cased	6.27 (1 Epoch)	438.2
DistilBERT-base uncased	3.44 (1 Epoch)	263.3
SciBERT-base uncased	4.24 (1 Epoch)	440.1
BERT-base uncased	6.23 (1 Epoch)	438.2
Linear-chain CRF	5 (300 iterations)	0.24

Table 4.6: Training time of each of the classification methods

4.4 Discussion of the Results

4.4.1 Conclusions

The primary objective of this work was to conduct an exploratory analysis of various deep learning approaches aimed at capturing semantic information directly from the data without relying on manually crafted features. This effort is driven by the hope that the insights and methodologies developed could be beneficial in settings where transfer learning is applicable, leveraging the intrinsic capabilities of these models to enhance their performance on related tasks.

The most effective method tested in this study was the linear-chain CRF classifier, which relies primarily on sequence-based styling information at the line level. Results from Table 4.5 highlight that styling-based semantics, such as the type of font used and the presence of bold or italic characters, serve as solid indicators for classification. This is further corroborated by the decent performance of the naïve algorithm.

Surprisingly, as seen in Table 4.4, language models like BERT and SciBERT, which only process information at an individual line level without considering sequential line information, still achieve a validation accuracy of around 0.63 on undersampled data. However, the main drawback of these models is their substantial requirement for training time and computational resources, as indicated in Table 4.6.

A notable alternative explored in this research is DistilBERT, which is approximately 40% smaller than BERT, retains 97% of BERT’s language processing capabilities and operates 60% faster. Our experiments show that DistilBERT performs comparably to BERT-based models in detecting theorems and proofs. Interestingly, using specific vocabularies, such as SciBERT’s scientific corpus-based vocabulary, marginally improves performance on scientific texts, adding an extra layer of information beneficial for classification tasks.

The impact of casing—distinguishing between lowercase and uppercase characters—on model performance was also evaluated. Our findings indicate that casing has minimal effect, suggesting that the emphasis should be more on the semantic and syntactic content rather than text formatting. Due to the significant training demands of these models, only a single epoch of fine-tuning was performed. Extending the fine-tuning Over more epochs, accuracy could potentially be enhanced further. This exploration into different model architectures and their capabilities in handling different text data types underscores the importance of model selection based on specific use cases and available resources.

Deep learning classifiers are not directly interpretable, but visualising some of the examples in both computer vision (refer to figure 4.13) and NLP (see figure 4.14) showcase some relationship is possible. Since this work introduced the task, we could only test a few classifiers and models available in different settings. It is possible to gain better scores by tweaking the hyperparameters or model’s size.

While demonstrating some acceptable raw results from a human observer’s standpoint, the computer vision based approaches yield the least desirable outcomes overall. These methods are particularly challenged by the importance of the Intersection over Union (IoU) score in the evaluation metric, which critically affects the mean Average Precision (mAP). Specifically, for line identification, such as distinguishing proofs and theorems, the precise demarcation of the exact edges around these text elements is essential and often problematic for these techniques, leading to lower performance metrics. Moreover object detection was also the most sensitive modality to the number of training iterations (see figure 4.12

y=Theorem (probability **0.947**, score **3.387**) top features

Contribution?	Feature
+3.355	Highlighted in text (sum)
+0.033	<BIAS>

definition 3.2 (hardness in relative entropy). let (π, γ) be a distributional search problem.

y=Theorem (probability **0.838**, score **1.512**) top features

Contribution?	Feature
+1.482	Highlighted in text (sum)
+0.030	<BIAS>

we say that (π, γ) has hardness (t, Δ) in relative entropy if: $\square\square\square$

y=Theorem (probability **0.332**, score **-0.842**) top features

Contribution?	Feature
+0.070	<BIAS>
-0.912	Highlighted in text (sum)

$k \mid r \square, g \square 1(r \square) \square \square s(\gamma), \gamma > \Delta,$

y=Theorem (probability **0.720**, score **1.238**) top features

Contribution?	Feature
+1.635	Highlighted in text (sum)
-0.398	<BIAS>

for all pairs $(g \square, s)$ of time t algorithms where $g \square$ is a two-block generator supported on π

y=Theorem (probability **0.414**, score **-0.353**) top features

Contribution?	Feature
+0.003	<BIAS>
-0.356	Highlighted in text (sum)

and $r \square$ is uniform randomness for $g \square 1$. similarly, for $\delta \in [0, 1]$, (π, γ) has hardness (t, Δ)

y=Theorem (probability **0.829**, score **1.745**) top features

Contribution?	Feature
+1.876	Highlighted in text (sum)
-0.130	<BIAS>

in δ -min relative entropy Π for all such pairs: $\delta \square \square \square$

Figure 4.14: Visualising LIME [VM23] weights associated with some tokens (text taken directly from pdfs and feed to BERT uncased)

and 4.13) and took the maximum number of training time per epoch, see table 4.6.

Finally, the project represented a significant advancement from Lucas’s original method, which he suggested transitioning from a rule-based approach to a more semantic, deep learning-based approach. This shift aimed to harness the deeper, more nuanced understanding possible with modern AI techniques to improve the analysis and processing of complex data.

4.4.2 Limitations

The evaluation process outlined in the project reveals several challenges with the current approach:

- 1. Text Line Limitations:** The extraction of text lines via pdfalto is confined by the spatial formatting of the PDF, where each line represents all characters within a single line of the document. This format can restrict the attention mechanism of language models, which may not be able to consider tokens in subsequent lines. This limitation can impact the generalisation capabilities of the models, as, unlike humans, these models struggle with fragmented textual inputs. An improved solution could involve redefining the document breakdown from single lines to more logical structures like paragraphs, allowing models to better understand the context and continuity of the text.
- 2. Inconsistency in Approach Comparison:** Comparing the effectiveness of various modalities is challenging due to their differing focus and methodologies. For instance, vision-based approaches analyse entire pages to identify and classify regions of interest, capturing visual semantics. In contrast, text-based approaches may only consider individual lines without the surrounding context. Additionally, CRF-based methods benefit from handcrafted features (e.g., identifying initial keywords like ‘proof’), visual descriptors (like font styles), and sequential learning across multiple lines. To accurately assess the strengths of each modality, it would be beneficial to standardise certain variables, such as analysing the same page or paragraph across all modalities without incorporating sequential modelling. Another potentially valuable strategy might involve exploring a purely font-based approach, leveraging the detailed font attributes available from the pdfalto XML will enhance feature extraction and model performance.
- 3. Sensitivity to Experimental Settings:** Deep learning models, especially in NLP and computer vision, are susceptible to experimental settings. For instance, in NLP, a small adjustment in the learning rate from the standard 2×10^{-5} or changes in batch size can significantly alter the results. This requires maintaining fixed, consistent settings to ensure the reliability and comparability of outcomes.
- 4. Cost Implications of Large-Scale Experiments:** Conducting experiments on large datasets with deep learning models is costly, limiting the scope of experiments to smaller, more controlled samples. This restriction impacts the scalability and the ability to generalise findings from the experimental data.
- 5. Challenges with IOU in Computer Vision:** The accuracy of computer vision models used in this project critically depends on the Intersection Over Union (IOU) metric. Setting a high IOU threshold, such as 0.95, might prioritise more precise learning but also risks missing critical detections if the set cutoff is unmet. This is particularly problematic in document analysis, where even a minor loss of text can severely impact content understanding. For example, missing a few key characters or paragraphs from proofs or theorems can make it difficult for readers to grasp the intended message, unlike in everyday object recognition tasks where minor pixel variations might not significantly alter the recognition of objects like a dog.

These observations suggest the need for a more unified framework or standardised benchmarks that allow for fair and direct comparison of different methodologies, facilitating a clearer understanding of their respective advantages and limitations.

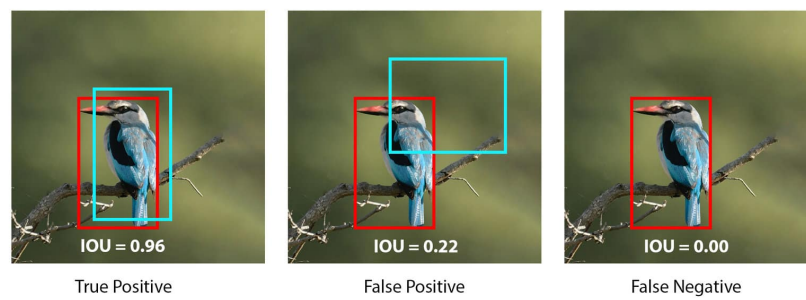


Figure 4.15: How the accuracy changes with different IOU measurements in an object detection task

Unimodal Approaches

Contenu du chapitre

5.1	Introduction	59
5.1.1	From Lines to Paragraphs	60
5.1.2	Related Work (Task)	61
5.2	Font Modality	62
5.2.1	Fonts to Features	62
5.2.2	Exploratory Data Analysis – Fonts	64
5.2.3	Simple ML Classifiers	65
5.2.4	Feature-Level Importance	68
5.2.5	Preliminary Experimental Results – LSTM Model	69
5.2.6	Font Model Evaluation on Large Dataset	71
5.3	Vision Modality	74
5.3.1	Image Preprocessing	74
5.3.2	Evolution of Neural Network Architectures in Computer Vision	77
5.3.3	Preliminary Experimental results – CNNs	83
5.3.4	Evaluation on Large Dataset	87
5.4	Text Modality	88
5.4.1	Related Work	89
5.4.2	Preleminary Work (Finetuning Models, Pretraining Language Model)	95
5.4.3	Evaluation of Language Model on Large Dataset	95

5.1 Introduction

The DocEng paper [MPS21] has laid the groundwork for utilising deep learning classifiers to extract mathematical results from documents. Although the methods for comparing the performance across different modalities were not established, the following steps involve isolating each modality to evaluate their independent benefits. This process will help identify the unique advantages of each approach.

In this section, we concentrate on developing and enhancing powerful unimodal models that utilise distinct modalities, including font, vision, and language. Each modality is uniquely capable of contributing to the overall performance of our models in different ways, and by refining these models separately, we aim to maximise their individual and combined effectiveness.

To rigorously evaluate the performance improvements of these models, we will test them across a large and standardised validation dataset consisting of 0.5 million samples. This substantial dataset allows us to robustly validate the models' capabilities and identify areas for further tuning and improvements. The outcomes from these evaluations will be documented, providing a comprehensive view of how each model performs and contributes to the task at hand. This approach not only helps in fine-tuning the models but also in establishing strong baselines for future research in the field.

5.1.1 From Lines to Paragraphs

In the DocEng project, we utilised the pdfalto tool to extract text lines from documents. However, as discussed in the previous section (refer to the chapter 4), text lines often do not capture complete sentences or fully convey the intended ideas due to document formatting constraints. A more practical and effective approach would be to utilise paragraphs, assuming they can be accurately identified. Using paragraphs helps ensure that the information is not truncated, which often happens with text lines. Most useful local information typically resides within a paragraph, capturing the context for accurate interpretation and analysis.

This approach would necessitate shifting our focus to operate at the paragraph level, where visual, textual, and styling-based font features are more coherent and potentially more informative. This adjustment would allow for a more holistic understanding of the document structure and content, enhancing the effectiveness of our extraction and classification tasks.

Grobid [L⁺24a] provides valuable functionality for parsing scientific papers into semantically segmented sections such as the title, abstract, and author names. By leveraging Grobid, we can effectively isolate and exclude a paper’s header and reference sections, focusing primarily on the body where proofs and theorems are typically found.

Once we have isolated the body section of the paper using Grobid’s output, we continue to use pdfalto under the same configurations as detailed in the DocEng paper. pdfalto is instrumental in identifying the precise locations of annotations related to mathematical results and all the text lines.

The next step involves matching these text lines to their corresponding paragraphs within the body of the paper. Each paragraph Grobid identifies is assessed to determine how it fits within the annotations provided by pdfalto. This fitting could occur in two ways:

1. **Complete Fit:** If a paragraph completely fits within an annotation box, we assign the paragraph the label of that annotation. This direct correlation helps categorise the paragraph under the intended mathematical result or concept.
2. **Partial Fit:** If a paragraph only partially fits within an annotation box, it indicates a potential overlap of labels. In such cases, the paragraph is marked as overlapping, and all relevant overlapping labels identified by the annotation are applied. This situation often arises due to inconsistencies or errors in Grobid’s segmentation.

This methodical approach allows for more structured analysis and categorisation of the content, enhancing the accuracy of information extraction from scientific documents.

For paragraphs that neither completely fit within an annotation box nor are identified as overlapping, we categorise them as *Basic*. A keyword or a range of words is utilised to simplify the labelling process. For example, terms such as *theorem*, *lemma*, and *definition* are all assigned the *theorem* label.

Once the paragraphs have been classified using Grobid, we employ pdfalto to determine how many text lines correspond with each paragraph. This helps identify the font characteristics used within each paragraph. The font size and order indicate the paragraph’s significance and category.

For further processing, we can use the text extracted by Grobid or fit the text from pdfalto within the coordinates of the Grobid paragraph to train language models. For vision-based models, we can extract bitmap images from the exact coordinates provided by Grobid.

We generate a *data.csv* file with detailed font, text, and coordinate information (including page numbers) for each paragraph to streamline this data handling. This excludes data from the title, header, and references, focusing solely on the body content, which is critical for our analysis. This structured approach facilitates a more efficient integration and utilisation of data across different processing modules.

We applied our methodologies to a substantial validation dataset of 3,683 PDFs, culminating in approximately 529,296 samples. This validation dataset is instrumental for comparing and benchmarking models across different modalities. Diverging from the previous training/testing split method described in the DocEng paper, we now base our dataset splitting on entire PDFs, assigning every paragraph from a PDF to the validation dataset. This approach acknowledges the practical reality where documents can vary significantly in length, which creates an imbalance in the dataset size but is a realistic representation of typical user scenarios. The aim is to enhance the model’s performance across various document lengths and types.

Validation Dataset

Multimodal Dataset: Our comprehensive dataset initially comprised 196,000 PDFs. After setting aside 3,683 for validation, we grouped the remaining PDFs into batches of approximately 1,000. These batches are consistently fed to various classifiers in the same sequence to maintain experimental consistency, although the ordering of samples within each batch can be shuffled to facilitate training. This structured approach ensures that each sample is exposed to the model only once, except in cases where the same paragraph appears in multiple documents. Training continues across several batches containing around 200,000 samples until the models converge. This strategy helps prevent overfitting by exposing the model to a more extensive range of samples rather than repeatedly training on a smaller dataset, which could compromise the model’s ability to generalise. All the tables using this dataset report accuracy on 4 class (including the overlap class) along with mean F_1 score for 3 major classes.

During the exploratory phase of our research, we aimed to design the base architectures and conduct preliminary analyses, including building merging scripts and running initial models. During this phase, we often faced challenges, such as missing information from Grobid or pdfto, encountering compilation errors related to \LaTeX extraction, and the extensive time required to evaluate our methods on the sizeable multimodal dataset. We initially conducted experiments with a smaller dataset to manage these issues effectively. Even though they were inconsistent for every modality, we ensured that all necessary dependencies were available. The overall idea was to try different techniques quickly to find the right architecture for every modality. This approach helped streamline our development process.

Once we established confidence in the effectiveness of our architectures, we scaled our efforts to the larger multimodal dataset, training from scratch but ensuring to keep the same training and validation dataset. We also measure the performance incrementally to cut off the training when convergence is reached. All components, including Grobid, pdfto, bitmap rendering, and the merging script, successfully formed our multimodal validation dataset. This allowed us to perform comprehensive comparisons and evaluations. Adopting this iterative approach was crucial in refining our methods and ensuring their robustness before applying them to several batches of 1k PDFs and validating at every step, facilitating a more thorough and practical analysis of our deep learning models.

5.1.2 Related Work (Task)

The work of Ginev et al. [GM20] highlights the importance of structure and explicit markup in scientific documents, particularly those written in \LaTeX and converted to HTML for analysis. Their approach is specifically tailored to leverage the structural cues in such documents. This methodology stands out for its emphasis on classification based on well-defined mathematical and theorem-like environments. This method acknowledges the complex nature of scientific discourse, which often embeds significant information in its layout and formatting.

However, Ginev et al.’s reliance on HTML rendered from \LaTeX introduces significant limitations. Specifically, the approach is constrained to scenarios where \LaTeX source code is available. It is less applicable in broader contexts where direct access to raw PDFs or other source formats might be necessary. Moreover, their evaluation method—focusing only on the **first logical paragraph** within each environment—might overlook the complexity and complete context of the surrounding text, potentially leading to incomplete or skewed analyses. This simplification is understandable from a technical perspective but might not capture the nuances that a more holistic approach would provide.

Furthermore, the requirement to sign a non-disclosure agreement to access the dataset used in their study could hinder the broader research community’s ability to verify and build upon their work. Open access to data is crucial for advancing scientific research, especially in fields like natural language processing and machine learning, where reproducibility and transparency are paramount.

In contrast, our approach in this study seeks to extract theorems and proofs directly from PDFs, offering a more robust framework that does not depend on the availability of source code. By directly interacting with the raw formats most commonly used in academic publishing, we aim to create a more versatile and widely applicable toolset. This method not only broadens the potential applications of our work but also enhances our ability to handle a more comprehensive array of document types and structures, ultimately leading to more accurate and comprehensive data extraction and classification.

Baselines 50-class	F ₁ score	F ₁ (no math)
Zero Rule	0.201	0.206
BiLSTM encoder-decoder	0.67	0.67
Baselines 13-class	F ₁ score	F ₁ (no math)
Zero Rule	0.388	0.369
LogReg	0.30	0.35
LogReg + GloVe	0.77	0.77
Perceptron	0.83	0.83
HAN	0.89	0.88
BiLSTM encoder-decoder	0.91	0.90

Table 5.1: Ginev’s [GM20] BiLSTM based approach (*only on first paragraph*)

5.2 Font Modality

To evaluate the potential of using font characteristics alone to predict paragraph labels, we explored the performance of font features without engaging in sequential learning through paragraphs. This approach is reminiscent of the methodology employed in the DocEng paper, where the CRF model utilised styling cues such as bold, italic, and mathematical fonts. By isolating these font attributes, we aim to determine if such features alone can effectively classify paragraphs.

If it is feasible to accurately classify paragraphs based solely on font attributes like bold, italics, or mathematical notation, we could vectorise these font characteristics and apply traditional machine learning techniques. This approach would involve creating feature vectors from the font data and training classifiers to see if they can effectively leverage this information to make predictions about the text. This method offers a potentially simpler alternative to more complex deep learning models, focusing on straightforward, informative stylistic cues.

5.2.1 Fonts to Features

To effectively build classifiers that leverage font-based information, it is essential to transform the fonts used in paragraphs into features that can be fed into machine learning classifiers. The process begins by extracting all the font names from 4,400 PDFs from the DocEng dataset. This is accomplished using a merging script that integrates output from both Grobid and pdfalto and is set to operate in paragraph mode. From there, we construct a dictionary cataloguing all the unique font names identified across these PDFs to track their frequency of use, see figure 5.1. In this dataset, we identified approximately 407 unique fonts. This comprehensive dictionary of fonts allows us to analyse font usage and frequency, providing a foundational dataset for training our classifiers to recognise and interpret font-based features effectively.

Manual Font Labelling

To enhance our classifiers’ ability to understand font-based information, we utilised a script developed by Lucas that uses regular expressions to identify specific font attributes such as italic, bold, and font type (serif, sans serif, or math). Lucas’s script effectively identified 17 italic, three bold, and 13 math fonts.

Building on this, I embarked on manually labelling all 407 unique fonts identified from the 4,400 PDFs in our dataset. This manual labelling involved examining the first occurrence of each font within its respective PDF to determine its attributes. This process, while meticulous, is subject to potential errors due to inaccuracies in font naming by pdfalto. However, it serves as a sufficient basis to create the initial version of our manually labelled font dataset.

The outcome of this manual labelling process is the creation of four binary features: `is_bold_manual`, `is_italic_manual`, `is_serif_manual`, and `is_math_manual`. To apply these features effectively in our classifiers, we normalise the values for all fonts used in a paragraph, resulting in a composite attribute value representing the overall font characteristics of the entire paragraph. This method harnesses font-based features comprehensively, improving our classifiers’ ability to analyse and interpret text based on visual style cues.

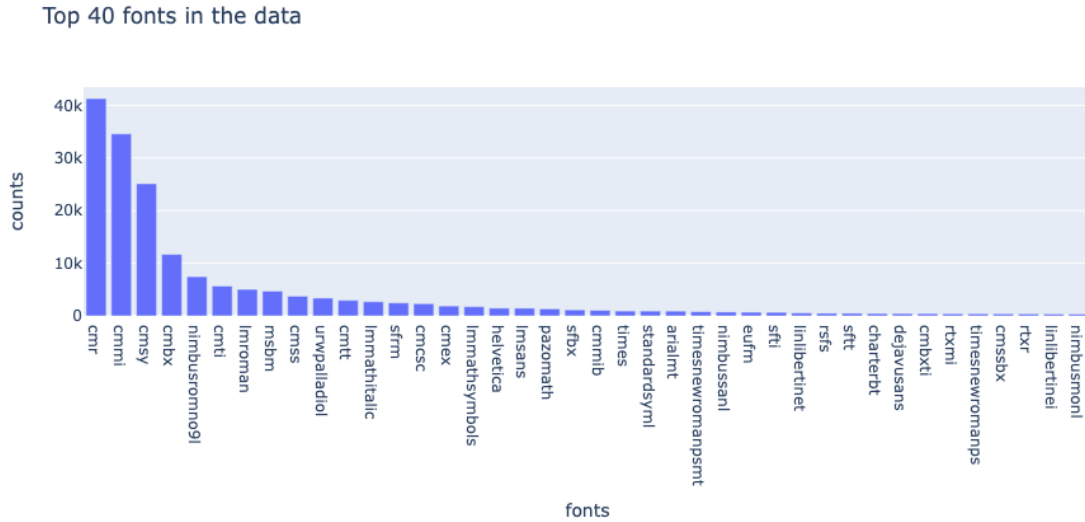


Figure 5.1: Top 40 most used fonts

	id	font_color	font_family	font_size	font_type	font_width	font_style	file_name
count	187708	187708	187708	187708.000000	187708	187708	187708	187708
unique	523	546	638	NaN	2	2	12	4400

Figure 5.2: number of unique occurrences of font features obtained from pdfalto

To see the manual labelling of each of the fonts, a spreadsheet is provided here: [Manual_font_labels.xlsx](#)

Font Labelling from pdfalto

While pdfalto provides some font-based information, such as whether a font is serif or sans-serif, the reliability of this data has proven questionable upon manual inspection, revealing several inaccuracies. Despite these issues, this dataset still holds potential value. Even with its inherent noise, the font information from pdfalto can be utilised to construct preliminary features for our classifiers. Leveraging this data, albeit imperfect, may offer a foundational level of insight into font styles, which could be refined or augmented with additional verification steps to enhance the accuracy of our feature set. This approach allows us to make initial assessments and potentially guide the development of more sophisticated feature extraction methods in future iterations.

Before using font-level features in a machine learning classifier, they must undergo preprocessing to ensure they are suitable for analysis. Here is a breakdown of how these features can be preprocessed, to see a list of all font features from pdfalto, see refer figure 5.2:

- Font Color (3 features):** The colour of the font is often provided in hexadecimal format. This can be converted to RGB (Red, Green, Blue) values, thus generating three distinct features: "*font_color_red*," "*font_color_green*," and "*font_color_blue*."
- Font Size (1 feature):** To normalise the font size, it is divided by 40. This assumes that the maximum font size is around 40, standardising the dataset's size.
- Font Type (1 feature):** This typically involves whether the font is *serif* or *sans-serif*. Label encoding is applied here, where serif might be assigned one value and sans-serif another, effectively converting this categorical data into a numerical format suitable for machine learning algorithms.
- Font Width (1 feature):** The width of the font, whether it is '*proportional*' or '*fixed*' also undergoes label encoding to transform these categories into a machine-learnable form.
- Font Style (5 features):** There are primarily five types of font styles considered: '*Normal*' (where the label is not present), '*superscript*', '*subscript*', '*bold*', and '*italics*'. All other styles are combinations of these basic ones. Each style is treated as a separate feature, where the presence or absence of each style is encoded.

This preprocessing is crucial as it transforms raw font data into a structured format that machine learning models can interpret and learn from effectively.

The comprehensive preprocessing and feature engineering efforts resulted in a total of Fifteen features are derived from both automated tools and manual labelling processes. Here is a breakdown of these features, which include 11 features extracted from pdfalto and four from manual labelling:

1. Normal: Indicates the absence of any specific font style.
2. Superscript: Identifies text that is formatted as superscript.
3. Subscript: Identifies text that is formatted as subscript.
4. Italics: Recognises italicised text.
5. Bold: Detects bold text.
6. is_Proportional: Indicates whether the font width is proportional.
7. is_Serif: Denotes if the font is serif.
8. font_color_red: The red component of the font's colour.
9. font_color_green: The green component of the font's colour.
10. font_color_blue: The blue component of the font's colour.
11. new_font_size: The normalised font size.
12. is_bold_manual: Manually labelled presence of bold text.
13. is_italic_manual: Manually labelled presence of italic text.
14. is_serif_manual: Manually labelled presence of serif text.
15. is_math_manual: Manually labelled identification of mathematical fonts.

Each feature is crucial for the classifier's ability to make accurate predictions based on textual appearance and style. Combining automatically extracted features with manually verified labels provides a robust dataset that enhances the effectiveness of the machine-learning models.

5.2.2 Exploratory Data Analysis – Fonts

I conducted manual investigations into several PDFs to further understand the stylistic nuances within paragraphs at a font level. This examination revealed intriguing patterns in the presentation of proofs and theorems. Notably, many instances of proofs were characterised by initial italics followed by standard text, while others prominently featured bold text throughout (as illustrated in Figure 3.27). Similarly, it was common to observe a sequence where bold text stating "*Definition*" was followed by significant use of italics, particularly in the initial logical paragraphs of theorem-like environments (as depicted in Figure 5.15).

These observations suggest a potentially structured use of font styles within mathematical documents, which could indicate underlying relationships between text presentation and content function. To explore this further, sampling the distribution of these styles across various paragraphs could provide deeper insights. Such an analysis could reveal if these stylistic choices are consistently applied across similar types of content, enhancing our understanding of how visual cues relate to text function (refer to Figure 5.3). This exploration aims to uncover any consistent patterns that could eventually aid in automating the recognition and classification of such text elements in academic papers.

Observing the kernel density estimation plot for the 'is_italic_manual' feature, it becomes clear that paragraphs labelled 'basic' seldom use italic characters. In contrast, the distribution for 'proof' paragraphs reveals a notable second peak, suggesting a modest use of italics, likely corresponding to instances where 'proof' is the first word and is italicised. The distribution for theorem-like paragraphs is more evenly spread, indicating more frequent use of italics, with some paragraphs having over 80% of their content in italics. This aligns with the hypothesis and reflects a distinctive pattern in using italics in mathematical documents, providing valuable insights for automated text classification.

To gain a visual understanding of the multidimensional data, including features extracted from pdfalto, I chose to reduce the dimensionality and project the features into a two- or three-dimensional space. This helps to identify potential clusters or patterns in the data. Among the various techniques for

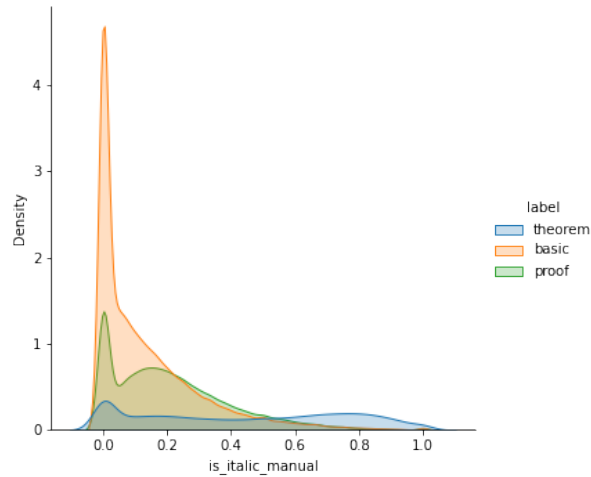


Figure 5.3: Class distribution (KDE) of 'is_italic_manual' over 388519 paragraphs

dimensionality reduction, I concentrated on four: PCA (Principal Component Analysis) [KPF01], t-SNE (t-Distributed Stochastic Neighbor Embedding) [VdMH08], MDS (Multidimensional Scaling) [Mea92] and UMAP (Uniform Manifold Approximation and Projection) [MHM18].

Each of these techniques has its advantages and drawbacks, particularly when it comes to preserving data structure and interpretability. PCA is a linear method often used for simplicity and speed, but it may not capture nonlinear relationships as effectively. t-SNE (*perplexity = 32, n_iter = 10000*) is more adept at revealing clusters or groups within the data but can be computationally intensive. MDS (*max_iter=4000*) It focuses on preserving the distance between points, which can help reflect the similarities or dissimilarities in the data. UMAP (*n_neighbors=20*) is a relatively newer technique gaining popularity due to its effectiveness in maintaining the global data structure and faster computation time than t-SNE.

I applied these dimensionality reduction algorithms to the entire dataset but visualised only 200 randomly sampled points from each category, see figure 5.8. This was done to avoid overcrowding in the graphs and ensure the visualisations remained transparent and interpretable. By sampling points from each category, I aimed to maintain a representation across different data types while simplifying the visualisation. This approach can offer insights into whether the features from different paragraph types form distinct clusters, indicating how well the features might perform in classification tasks.

5.2.3 Simple ML Classifiers

Within a corpus of 385,368 paragraphs (after filtering out the NAN entries), adopting a random sampling strategy allows for a division where 90% of the data is allocated for training (346,831 paragraphs) and the remaining 10% for testing (38,537 paragraphs). The training data comprises 175,405 'basic', 110,709 'proof', and 60,717 'theorem' labelled paragraphs. Correspondingly, the test data comprises 19,490 'basic', 12,301 'proof', and 6,746 'theorem' labelled paragraphs.

Drawing from insights in the DocEng paper, where data balancing showed improved performance (refer to table 4.4), we extend this concept to our paragraph dataset, which offers richer context for each data point. To thoroughly investigate the Impact of data balance, we experiment with three distinct data configurations:

1. **As-Is (Without under/over-sampling):** Models are trained on the original distribution of the training data without any modifications to ensure class balance.
2. **Random Undersampling:** The majority class in the training data is randomly reduced to match the minority class, bringing the count down to 60,717 samples for each class.
3. **Random Oversampling:** Conversely, the minority class is augmented to equal the majority class, increasing its size to 175,405 samples through random replication.

These variations aim to explore how different sampling techniques affect the model's ability to generalise from the training data to unseen data and whether a balanced dataset contributes to better model performance in contexts with extensive textual information, such as paragraphs.

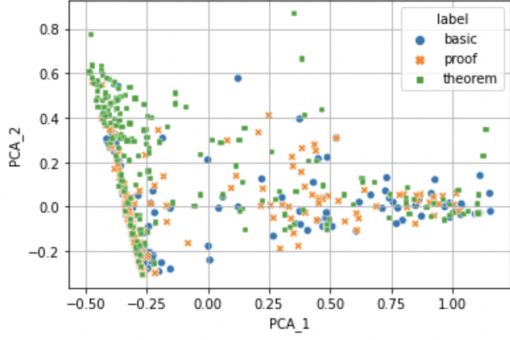


Figure 5.4: (a) PCA visualization

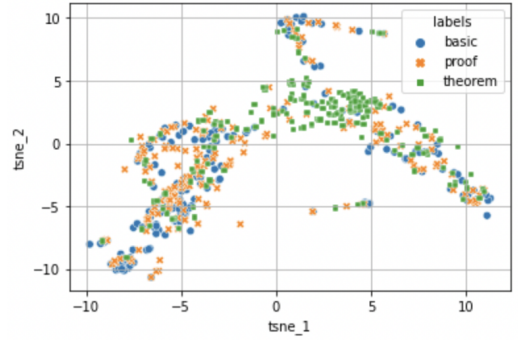


Figure 5.5: (b) Tsne visualization

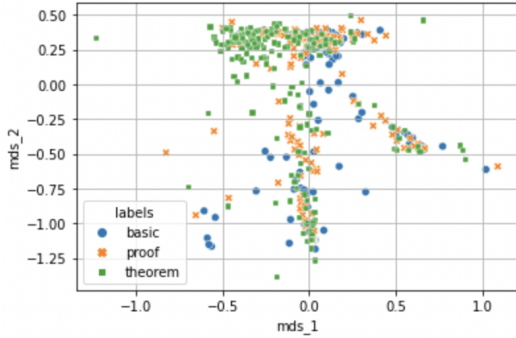


Figure 5.6: (c) MDS visualization

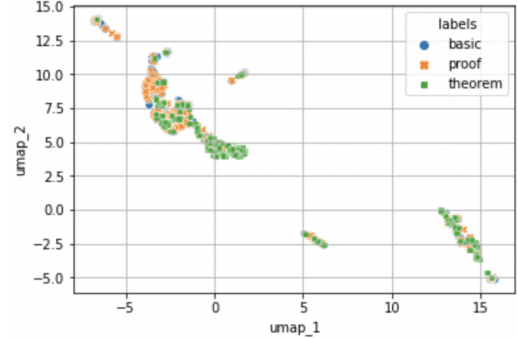


Figure 5.7: (d) UMAP visualization

Figure 5.8: Dimensionality reduction to visualise clusters (based on fonts) of paragraphs

In our study, we explored three straightforward machine-learning classifiers to handle the classification task based on font features within paragraphs:

1. **K Nearest Neighbor (KNN):** The underlying premise of this classifier is that paragraphs with similar font attributes should cluster closely within the feature space and, therefore, have similar labels. KNN is generally robust to outliers but can be sensitive to the effects of under and over-sampling because duplicate points could skew the distance measurements and, subsequently, the voting mechanism of the classifier.
2. **Multinomial Naive Bayes:** This classifier is based on the assumption of conditional independence between features. However, many font features (italic, bold, and math styles) are likely correlated. For example, 'is_italic_manual' might correlate with an automatically extracted 'is_italic' feature. The performance of Naive Bayes can be impacted by class imbalance, as its probability estimates might be skewed towards the more prevalent class.
3. **Linear Model (Logistic Regression):** Logistic regression seeks to establish a linear relationship between features and the log odds of the outcomes. It minimises the logistic loss, where outlier points can have a significant influence since the model aims to find optimal weights that minimise this loss function. Class imbalance can particularly affect logistic regression, as it may lead to biased predictions toward the majority class.

Each of these classifiers offers unique strengths and potential drawbacks, influenced by the distribution and balance of the dataset. Their performance on the task will provide insights into how font features alone can contribute to the identification and classification of different paragraph types in scientific papers, see table 5.2.

We can employ PDF-based sampling, where a 90/10 split is applied at the PDF level. This ensures that all paragraphs from a given PDF are placed in the same bin, either for training or testing. This method mirrors a more practical and realistic approach to splitting data, reflecting how models might be used in real-world applications where they are given all the documents to process, see table 5.3. However, this form of sampling may result in differences between the training and testing Distribution, potentially

Table 5.2: Classifier Performance for Different Sampling Strategies (random split)

Sampling	Classifier	Accuracy	Weighted F ₁
As is	KNeighborsClassifier(n_jobs=4)	0.5677	0.5759
As is	MultinomialNB()	0.3526	0.5059
As is	LogisticRegression(n_jobs=4, random_state=0)	0.5481	0.5832
Undersampled	KNeighborsClassifier(n_jobs=4)	0.5489	0.5483
Undersampled	MultinomialNB()	0.5332	0.5290
Undersampled	LogisticRegression(n_jobs=4, random_state=0)	0.5607	0.5627
Oversampled	KNeighborsClassifier(n_jobs=4)	0.5451	0.5425
Oversampled	MultinomialNB()	0.5325	0.5282
Oversampled	LogisticRegression(n_jobs=4, random_state=0)	0.5621	0.5638

Table 5.3: Classifier Performance for Different Sampling Strategies (PDF-level split)

Sampling	Classifier	Accuracy	Weighted F ₁
As is	KNeighborsClassifier(n_jobs=4)	0.5588	0.5683
As is	MultinomialNB()	0.3413	0.4985
As is	LogisticRegression(n_jobs=4, random_state=0)	0.5439	0.5812
Undersampled	KNeighborsClassifier(n_jobs=4)	0.5578	0.5577
Undersampled	MultinomialNB()	0.5386	0.5343
Undersampled	LogisticRegression(n_jobs=4, random_state=0)	0.5666	0.5695
Oversampled	KNeighborsClassifier(n_jobs=4)	0.5440	0.5419
Oversampled	MultinomialNB()	0.5391	0.5348
Oversampled	LogisticRegression(n_jobs=4, random_state=0)	0.5670	0.5697

Table 5.4: Feature Selection Results for Each Classifier

Feature Name	KNeighborsClassifier	MultinomialNB	LogisticRegression
Normal	✓	×	✓
Superscript	×	×	✓
Subscript	×	×	✓
Italics	×	✓	×
Bold	✓	✓	×
is_Proportional	✓	✓	×
is_Serif	×	✓	×
font_color_red	×	✓	×
font_color_green	×	✓	×
font_color_blue	×	✓	×
is_bold_manual	×	×	✓
is_italic_manual	✓	×	✓
is_serif_manual	✓	×	×
is_math_manual	✓	×	✓
new_font_size	✓	×	✓

leading to overfitting if the model overly adapts to the idiosyncrasies of the training set. Applying PDF-based sampling (without removing duplicates) to our dataset resulted in 352,106 samples for training and 36,413 for testing, providing us with substantial data for robust model training and validation.

In the context of classifier performance about different sampling techniques, it is observed that, except Naive Bayes—which relies on prior class probabilities—the choice of sampling strategy does not generally have a substantial effect on the results. Using the data in its original Distribution (As-is, see table 5.2) often yields satisfactory outcomes. Interestingly, while more aligned with practical, real-world applications, a PDF-wise see table 5.3, split does not significantly diminish accuracy. This suggests that such a splitting method could be advantageous in operational settings without compromising the model’s performance.

5.2.4 Feature-Level Importance

Feature selection is crucial when building classifiers, especially with handcrafted features. Accurately identifying and focusing on the most relevant features can enhance model performance and reduce training times. To examine this, we trained three different classifiers (KNN, Naive Bayes, Logistic Regression) using standard settings, they were subjected to sequential feature selection using a forward selection approach.

Forward Sequential Feature Selection (SFS) is a greedy algorithm that iteratively identifies the best feature to add to the existing feature set based on model performance. Starting with no features, it selects the single feature that maximises the cross-validated score of the model. This process is repeated, adding one feature at a time until the predetermined number of features is reached. For our investigation, we set the target at seven features—approximately half of the original 15 font features. Additionally, we employed 3-fold cross-validation during feature selection to ensure robustness and prevent overfitting.

This methodical approach to feature selection aims to identify a subset of features that contribute most significantly to the predictive power of the classifiers, providing a more efficient and potentially more accurate model, for reference see the table 5.4 on removed features.

We do not observe significant gains in dropping nearly half of the features from 15 to 7 (see table 5.5). This suggests that if we use a deep learning classifier, a few neurons should be able to effectively capture the essential patterns in the data without necessitating a large input feature set. This finding underscores the possibility that the critical font features are sufficiently representative and that additional features may not contribute significantly more to the model’s learning capability. Thus, a streamlined neural network architecture with fewer neurons might be efficient and sufficient for comparable performance.

Table 5.5: Performance of Classifiers Before and After Feature Selection with 7 of 15 Features Kept

Classifier	Acc (Before)	Acc (After)	Weighted F ₁ (Before)	Weighted F ₁ (After)
KNN	0.5746	0.5667	0.5679	0.5595
NB	0.5076	0.5069	0.3531	0.3410
LR	0.5845	0.5832	0.5481	0.5460

5.2.5 Preliminary Experimental Results – LSTM Model

In the previous section, we explored machine learning-based approaches that utilise preprocessing of fonts with pdfalto and manually labelled features to categorise fonts into unique styles such as bold, italic, math, and serif. While these methods help establish a proof of concept, they also present several disadvantages:

1. **Limitations of Simple ML Classifiers:** The methods employed are based on simple machine learning classifiers, which may not leverage the advanced capabilities of deep learning approaches. Deep learning could offer superior performance through more sophisticated pattern recognition and learning from larger, more complex datasets.
2. **Reliance on Manual Labeling:** The accuracy of these classifiers heavily depends on the quality of manual labelling of fonts and the features extracted from pdfalto. Manual labelling is prone to human error and inconsistencies, which can directly impact the effectiveness and reliability of the classifiers.
3. **Lack of Sequential Font Feature Analysis:** The current approach considers overall values for font features but does not account for the text’s sequence or order of font usage. Understanding the sequence could add significant value, such as identifying that the first word is bold and in a specific font. This sequential information might provide crucial contextual cues that enhance the classifier’s ability to understand and interpret the text more effectively.

These drawbacks highlight the need for further refinement of the methodologies and potentially exploring more advanced techniques to automate and improve the accuracy and efficiency of font style classification in documents.

To address the challenges identified with traditional and manual methods of font classification, we propose a deep learning solution that utilises LSTMs to process fonts as sequential inputs. LSTMs, or Long Short-Term Memory networks [HS97], is a special kind of recurrent neural network (RNN) specifically designed to handle sequence prediction problems effectively. This makes them particularly suitable for tasks where understanding the sequence and context of data is crucial.

Why LSTMs are suitable for this task:

1. **Sequential Information Handling:** LSTMs excel in capturing and learning from the sequential nature of data. By using fonts as inputs, where the sequence might carry contextual significance (e.g., the first word in bold), LSTMs can leverage their architecture to recognise and learn these patterns effectively.
2. **Memory Gates:** LSTMs are equipped with internal mechanisms—input, output, and forget gates—that help manage information flow. These gates allow LSTMs to retain important information over long sequences while discarding irrelevant data, which is ideal for processing complex documents where the relevance of information can vary significantly across the text.
3. **Mitigating Vanishing Gradients:** One of the historical challenges with standard RNNs is the vanishing gradient problem, where gradients become too small for effective learning through backpropagation over long sequences. LSTMs address this issue with their gated structure, ensuring stable learning and making them robust for tasks requiring the learning of long-term dependencies.
4. **Proven Effectiveness:** LSTMs have been highly effective in various applications beyond font recognition, including natural language processing, speech recognition, and even complex sequence prediction tasks in financial time series. Their ability to model and predict based on historical sequences makes them a reliable choice for dynamically styled text in documents.

Table 5.6: Impact of Font Size with the LSTM Architecture

Model	Loss	Train Accuracy (%)	Test Accuracy (%)	Inf. Time (ms/step)
LSTM	0.786	62.48	62.50	57
LSTM (with font size)	0.753	64.36	64.40	66

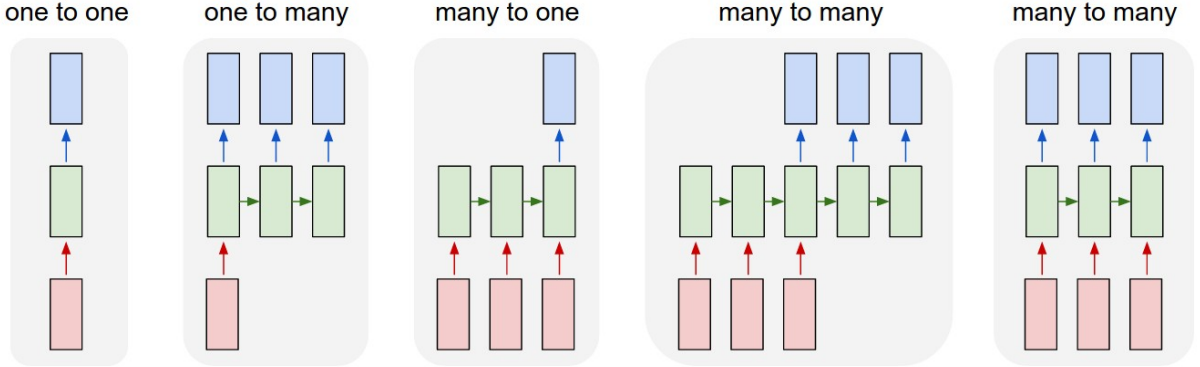


Figure 5.9: Different types of RNN Networks

By applying LSTMs to our dataset, we aim to build a model that learns the specific characteristics associated with different font styles and how these styles are utilised in real document contexts. This approach promises to enhance the classifier’s ability to make nuanced distinctions based on the stylistic and sequential use of text fonts, potentially improving accuracy and reliability in automated document analysis, see table 5.6.

For the task of classifying paragraphs as ‘basic’, ‘proof’, or ‘theorem’ based on the sequence of fonts used, we are employing a “many-to-one” LSTM architecture (see figure 5.9). This model type is well-suited for tasks, a sequence of inputs needs to be condensed into a single output label. Here is how we plan to implement and utilise this LSTM setup:

1. **Input Sequence:** Each paragraph’s sequence of fonts, as used in the text, will serve as the input. This includes both the font family and the size of each font, which can provide critical stylistic information about the paragraph. The sequence of fonts will be extracted using the pdfalto tool, which annotates each text token with its corresponding font details. This tool is particularly adept at handling standard L^AT_EX fonts, such as ‘cmr10’ for Computer Modern Roman in 10-point size.
2. **Encoding Font Information:** To prepare our font data for the LSTM, we first assign a unique token to each font (including its size) in our training corpus. To standardise the input size for the LSTM, we pad these sequences with zeros on the left until they reach a uniform length of 1000 dimensions.
3. **Embedding Layer:** A 32-dimensional embedding layer captures and constructs a latent representation of each font’s order within the paragraph. This layer helps the model understand and interpret the sequential styling cues embedded in the text.
4. **Performance Evaluation:** We compare the performance impact of using just the font names versus using both font names and font sizes as tokens. Preliminary results suggest that including font size in the tokens provides additional context that enhances the model’s accuracy.

This approach allows the LSTM to effectively learn from the stylistic and sequential nuances of the fonts in each paragraph, providing a sophisticated way to classify paragraphs based on their typographic features.

After achieving satisfactory initial results with the LSTM model, which offers lower inference costs than more complex transformer-based variants, we decided to optimise our model further using Bayesian optimisation. This approach aims to find the optimal set of hyperparameters to enhance the model’s performance.

Bayesian optimisation is particularly effective for this purpose as it builds a probabilistic model of the function mapping from hyperparameter values to the objective based on past evaluations. The objective

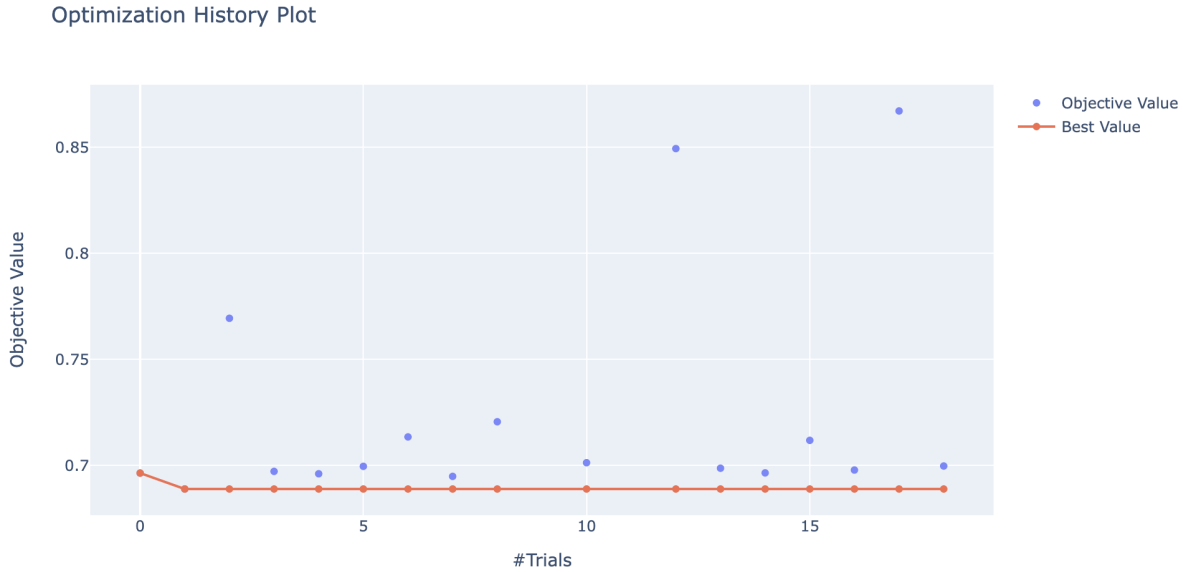


Figure 5.10: Hyperparameter search optimisation (objective function set to minimise the cross entropy loss)

function is the cross-entropy loss, which we use to train the LSTM model. By minimising this loss, Bayesian optimisation helps us iteratively converge on the best hyperparameters, thus refining our model’s training process and potentially leading to improved classification accuracy.

This strategic use of Bayesian optimisation allows us to systematically and efficiently explore the hyperparameter space, reducing the costly evaluations typically required by other optimisation methods and ultimately enhancing the LSTM’s performance in classifying text-based on stylistic and sequential font features.

After setting the hyperparameters such as dropout [SHK⁺14], the number of LSTM layers (stacked), and the number of LSTM cells in each layer, we did not observe significant differences in the model’s performance based on these settings (see figures 5.11, 5.10). We can also visualise the importance of individual hyperparameter, see figure 5.12.

Consequently, we decided to solidify the architecture with 128 LSTM cells per layer. This decision was informed by our aim to efficiently capture and leverage sequential information inherent in font usage across paragraph blocks.

5.2.6 Font Model Evaluation on Large Dataset

We compiled a comprehensive font vocabulary from the training data consisting of 4,031 unique fonts, including their sizes. Each paragraph block is represented as a sequence of font identifiers, ensuring that every textual content is characterised by its stylistic attributes. We applied left padding to each sequence to maintain uniform input dimensions across all training samples, with a maximum length set to 1,000.

The processed data is then fed into a simple LSTM network, which consists of a single layer with 128 LSTM cells. This model, based on the seminal architecture proposed by Hochreiter and Schmidhuber, in 1997, was specifically chosen for its ability to process sequential data effectively. The primary goal is to utilise this LSTM network to capture the sequential relationships among fonts within a paragraph, thus aiding in accurately classifying the paragraph’s label based on its typographic features (see figure 5.13 for architecture diagram). This approach underscores the model’s utility in discerning the nuanced use of fonts as a marker for differentiating between text categories such as *‘basic’*, *‘proof’*, *‘theorem’* and *‘overlap’* in scholarly documents.

We continuously monitor the performance of the LSTM-based font model by evaluating it on the validation data after processing each batch of approximately 1,000 PDFs. This approach allows us to assess the model’s effectiveness in real-time and make necessary adjustments based on its performance with each new batch of data, see figure 5.14.

Parallel Coordinate Plot

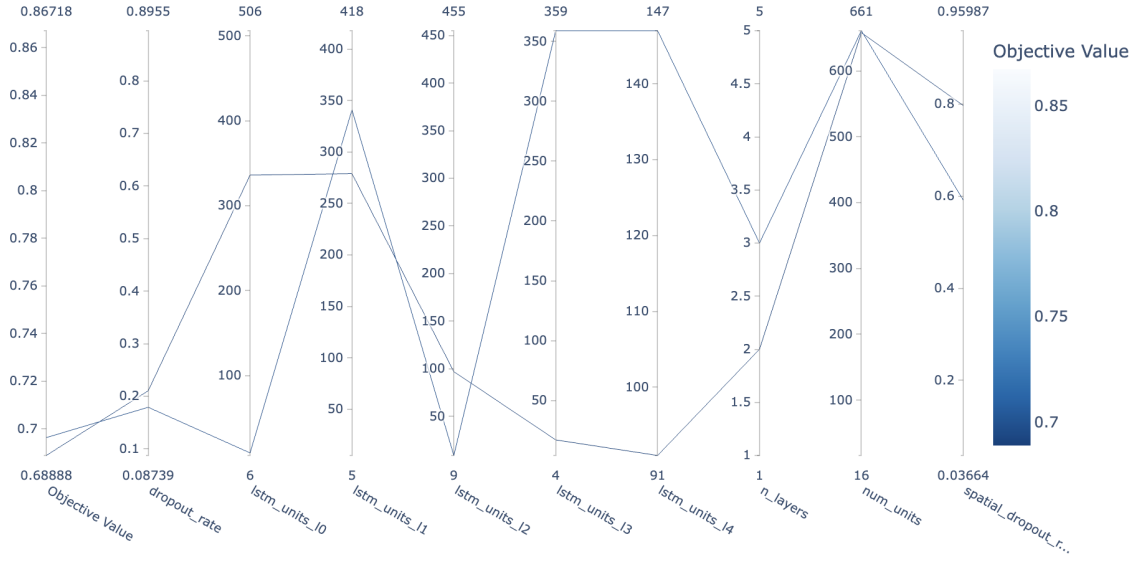


Figure 5.11: Visualising two runs from the search space

Hyperparameter Importances

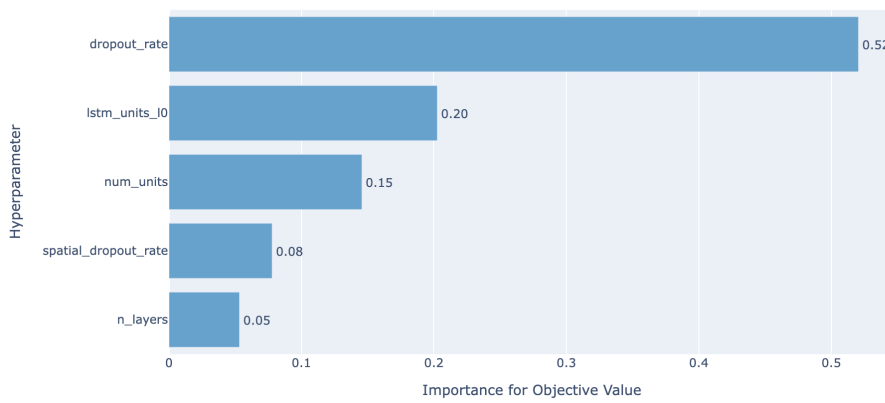


Figure 5.12: Impact of feature when searching for the optimal architecture

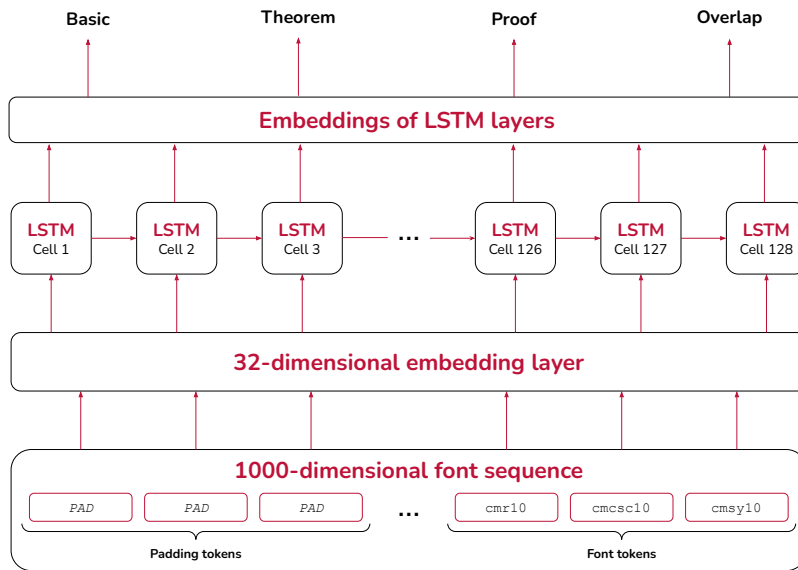


Figure 5.13: Final Architecture (LSTM) to capture font modality

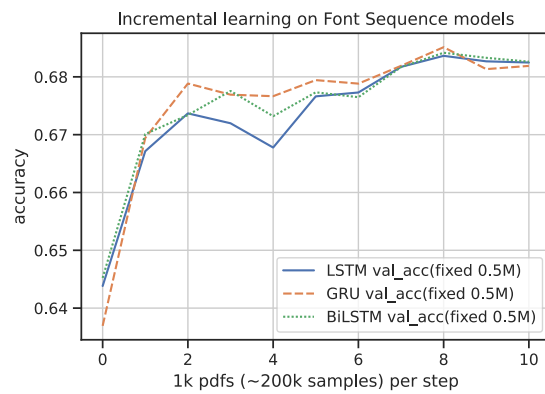


Figure 5.14: Accuracy of font models on fine-tuning task concerning number of batches

Table 5.7: Performance comparison of font models

Model	#Batches	Inf. time (ms/step)	Accuracy (%)	Mean F ₁ (%)	#Params
Dummy (most-frequent)	-	-	59.41	24.85	-
LSTM (128)	11	14	64.93	45.48	1.72M
GRU (128)	11	14	60.59	47.29	1.72M
BiLSTM (128)	11	26	64.71	45.45	1.82M
BiGRU (128)	11	25	64.22	49.15	1.78M

Theorem 1.8 ([HRVW09]). Let $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a one-way function, let X be uniformly distributed in $\{0, 1\}^n$, and let (Y_1, \dots, Y_m) be a partition of $Y = f(X)$ into blocks of length $O(\log n)$. Then (Y_1, \dots, Y_m, X) has next-block accessible entropy at most $n - \omega(\log n)$.

Proof. Since f is (t, ε) -one-way, the distributional search problem $(\Pi^f, f(X))$ where $\Pi^f = \{(f(x), x) : x \in \{0, 1\}^n\}$ is (t, ε) -hard. Clearly, $(f(X), X)$ is supported on Π^f , so by applying Theorem 3.8, we have that $(\Pi^f, f(X), X)$ has witness hardness $(\Omega(t), \log(1/\varepsilon))$ in relative entropy and $(\Omega(t), \log(1/\varepsilon) - \log(2/\delta))$ in $\delta/2$ -min relative entropy. Thus, by Theorem 4.7 we have that $(Y_1, \dots, Y_{n/\ell}, X)$ has next-block inaccessible relative entropy $(\Omega(t \cdot \Delta \cdot \ell^2 / (n^2 \cdot 2^\ell)), \log(1/\varepsilon) - \Delta)$ and next-block inaccessible δ -min relative entropy $(\Omega(t \cdot \delta \cdot \Delta \cdot \ell^2 / (n^2 \cdot 2^\ell)), \log(1/\varepsilon) - \log(2/\delta) - \Delta)$, and we conclude by Theorem 4.9. \square

Figure 5.15: Strong visual indicators

During our analysis, we also explored the potential benefits of enhancing the model with bidirectional LSTM layers designed to capture information from past and future contexts within the data sequence. Despite the theoretical advantages of bidirectional LSTMs in many sequence learning tasks, we discovered that they did not significantly improve the performance in our specific application (see table 5.7 for final scores). This finding suggests that the directional sequence of fonts within paragraphs may not provide additional contextual information necessary for improving the classification results beyond what is achieved with the unidirectional LSTM setup. Thus, we continued with the original LSTM configuration, focusing on optimising other aspects of the model and training process.

5.3 Vision Modality

5.3.1 Image Preprocessing

While in our font modality, LSTM model [HS97] successfully captures the order and sequence of font usage, it may fall short in identifying specific visual relationships and symbols within the text. For example, it might not effectively recognize the end-of-proof symbol, typically denoted by a unique character, unless it is consistently associated with a particular font style (see figure 5.15). Additionally, variations such as subscript, superscript, and special mathematical symbols, which may not have fixed font representations, could be overlooked.

This limitation highlights the need for integrating additional features or modalities that can directly address these visual elements. By expanding the model to include the detection of such specific typographic features, we could enhance its ability to interpret and classify academic text more accurately, especially in mathematical and scientific documents where these elements are crucial for understanding the content.

To begin our analysis, we selected a subset of 4,644 PDFs containing approximately 772,621 paragraphs, all free of overlapping labels. After filtering out entries missing image coordinates, we retained 772,617 paragraph blocks. We then applied a PDF-wise split, ensuring all paragraphs from a single PDF were grouped, resulting in an 80/10/10 distribution for training, validation, and testing sets. This Distribution led to 623,994 paragraphs in the training set, 77,237 in the validation set, and 71,386 in the test set. For this report, we focus solely on the training and validation sets. One of the major tasks involved extracting cropped bitmap images of the paragraphs based on page numbers and extracted coordinates.

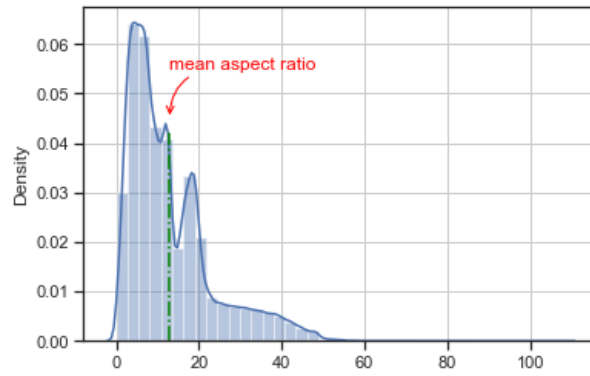


Figure 5.16: Distribution of aspect ratios

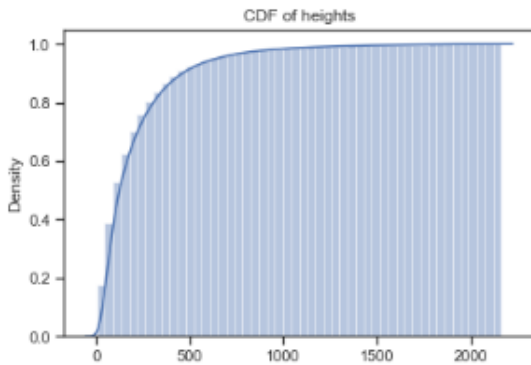


Figure 5.17: (a) Cumulative distribution function of heights of paragraphs

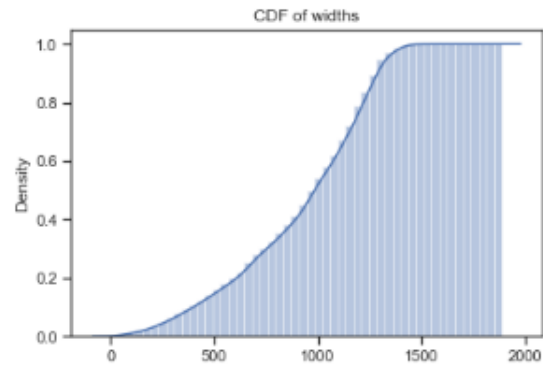


Figure 5.18: (b) Cumulative distribution function of widths of paragraphs

Figure 5.19: CDF of heights and widths of blocks with all aspect ratios

This process is primarily IO-bound, with significant time waiting for the system to write the images to disk. We employed multithreading to expedite the generation of these images, which significantly reduced processing time. On my 2021 M1 MacBook, utilizing about 14 out of 16 cores, it took approximately 28.5 minutes to generate the validation dataset of 77,237 paragraphs. Each cropped image has a resolution that varies depending on the width and height settings used in the PDF. We also analyzed the distribution of aspect ratios across these images to understand the diversity in dimensions we were handling (see figure 5.16).

In the realm of vision-based architectures, there is a notable trend in the literature to utilize architectures designed for fixed-resolution square images, such as those used for ImageNet [DDS⁺09] (224x224), CIFAR (32x32) [Kri09], and MNIST (28x28) [Den12]. These datasets primarily consist of images representing real-world objects formatted to fit these standard dimensions. However, this approach presents a few challenges when adapting these models to different, more practical applications where image sizes vary significantly and may not conform to these small, fixed dimensions.

When adapting vision-based architectures for analyzing text images from scientific documents, two main challenges arise due to the specific nature of these images:

1. **Handling Different Aspect Ratios:** Text images derived from paragraph blocks in scientific papers come in a wide range of aspect ratios, unlike the fixed square resolution (e.g., 224x224 pixels) commonly used in standard vision architectures. To effectively process these images, it is crucial to standardize the input dimensions. One approach to manage this diversity, fix a standard aspect ratio that best accommodates most data. To determine the most appropriate standardization, we can analyze the Cumulative Distribution Function (CDF) of the height and width across all paragraph images, which helps us select an aspect ratio that minimizes distortion and preserves the textual content’s legibility and layout (see figures 5.18, 5.17).
2. **Adjusting Background Color:** Scientific articles typically feature a white background with

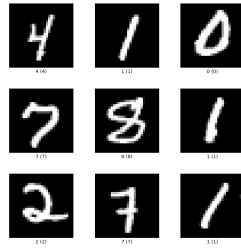


Figure 5.20: Example of MNIST dataset (white on black background)

black text unlike the MNIST dataset, see figure 5.20. This can pose challenge for Convolutional Neural Networks (CNNs), especially in architectures that utilize operations like max pooling. Max pooling is designed to detect strong intensity features within a specified kernel size, which may not perform optimally with light text on a dark background. To address this, we can apply a bitwise NOT operation to invert the image colours, resulting in a format similar to MNIST (i.e., white text on a black background). This alteration enhances the text’s visibility for the CNN and aligns with the training conditions of many pre-trained models, which often expect dark text on a lighter background.

Based on the analysis of the paragraph dimensions from the dataset, we observe that at least 90% of the paragraphs have heights of 600 pixels or less and widths of 1600 pixels or less. This statistical insight allows us to standardize the input size for our neural network processing to 600x1600 pixels. We apply resizing techniques for paragraphs exceeding this resolution to conform to these dimensions. We add vertical and horizontal white padding for those below this threshold to achieve a uniform resolution of 600x1600 pixels across all images. This standardization simplifies the input process and allows the network architecture to adapt its parameters based on this set resolution dynamically.

After applying the necessary preprocessing steps—including patch extraction, resizing or padding, and a bitwise NOT operation to invert image colours—we have successfully processed a total of 622,618 images for the training set and 76,987 for the testing set. This preparation is crucial for effectively training our model, ensuring it can accurately analyze and classify the text data from the scientific papers.

The development of Convolutional Neural Networks (CNNs) has witnessed numerous groundbreaking advancements over the years, each contributing significantly to the Evolution of machine learning, particularly in computer vision. These innovations are often introduced through specific architectures, setting new standards in the industry. Let us delve into some of these major advancements and their contributions:

1. Skip Connections (Residual Learning): Popularized by the ResNet architecture [HZRS16], skip connections help mitigate the vanishing gradient problem by allowing gradients to flow through shortcuts across multiple layers. This facilitates the training of much deeper networks by enabling feature reuse and reducing training time, significantly enhancing model performance.
2. Inception Block: The Inception block [SLJ+15], utilized within the Inception network, handles multi-scale information Efficiently. It contains parallel convolutional paths with varying kernel sizes and a pooling path, simultaneously allowing it to capture spatial hierarchies from fine details to broader features. This reduces dimensionality and computational costs while enhancing the network’s focus and Accuracy.
3. Batch Normalization: Introduced by Ioffe and Szegedy [IS15], batch normalization standardizes the inputs to a layer within a network, helping stabilize and accelerate the training process. This feature, integral to enhancing training efficiency and robustness against various initializations, became a key component of the future Inception network architecture, commonly known as InceptionNet.
4. Depthwise Separable Convolutions [HZC+17]: First seen in the MobileNet architecture, depthwise separable convolutions break down the convolution operation into two layers—one for filtering and another for combining. This approach Reduces both the computational cost and the number of parameters, optimizing the architecture for mobile and edge devices.

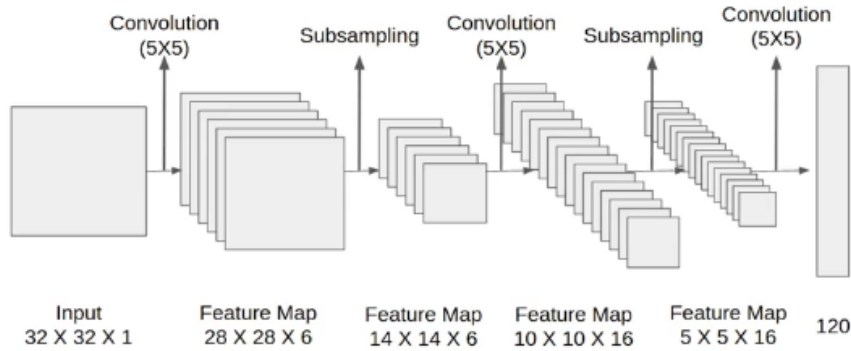


Figure 5.21: LeNet [LBBH98] architecture

5. Dilated Convolutions [YK15]: These are used to increase the receptive field of the network without adding more parameters or computation, making them valuable for segmentation tasks where preserving resolution while capturing larger context is essential. A prime example is the DeepLab architecture, which utilizes dilated convolutions for semantic image segmentation.
6. Neural Architecture Search (NAS): NAS [ZL17] automates the architectural design process of neural networks, optimizing for performance and computational efficiency. This has led to the creation of state-of-the-art architectures tailored for specific performance targets or computational constraints, such as NASNet and EfficientNet.
7. Attention Mechanisms [VSP⁺17]: Initially used in natural language processing, attention mechanisms have also proven effective in vision tasks. They enable models to focus on the most relevant parts of the input data, enhancing performance on tasks requiring contextual understanding. The Vision Transformer (ViT) architecture, which heavily relies on attention mechanisms, has been adapted for various vision tasks.

These advancements illustrate how modern neural network architectures have evolved to incorporate these technologies naturally, often making a base architecture or backbone rich with these advanced features. Thus, when selecting an architecture for a specific application, one often benefits from these built-in advancements without needing to integrate or modify the architecture to include them separately. This integration significantly simplifies the process of model development, focusing efforts on optimizing performance rather than reengineering the architectural framework.

5.3.2 Evolution of Neural Network Architectures in Computer Vision

One of the first CNN architectures, The LeNet architecture (see figure 5.21), was developed by Yann LeCun and colleagues in the late 1980s and early 1990s. It is one of the earliest convolutional neural networks and was pivotal in developing deep learning technologies. Primarily designed for handwritten and machine-printed character recognition, LeNet represents a foundational architecture that includes essential convolutional layers followed by subsampling, now commonly called pooling layers. The architecture typically consists of two sets of convolutional and average pooling layers followed by a flattening convolutional layer, then two fully connected layers and a Gaussian connection layer at the end, which classifies inputs.

LeNet [LBBH98] laid the groundwork for convolutional neural networks (CNNs), influencing subsequent designs, notably AlexNet [KSH12], which emerged as a breakthrough model in 2012. Building on the foundational concepts introduced by LeNet, such as layered convolution and pooling, AlexNet introduced deeper layers, rectified linear units (ReLUs), dropout, and data augmentation to effectively manage more complex and higher-dimensional image data from the ImageNet challenge. AlexNet's architecture featured more convolutional layers and significantly more parameters than LeNet, enabling it to capture a broader range of features at multiple scales (see figure 5.22). It was designed to process higher-resolution images and utilized powerful GPUs for training, addressing the computational and memory limitations that constrained earlier models like LeNet. AlexNet's performance in the ImageNet competition, where it significantly reduced error rates compared to its predecessors, marked a pivotal moment in demonstrating

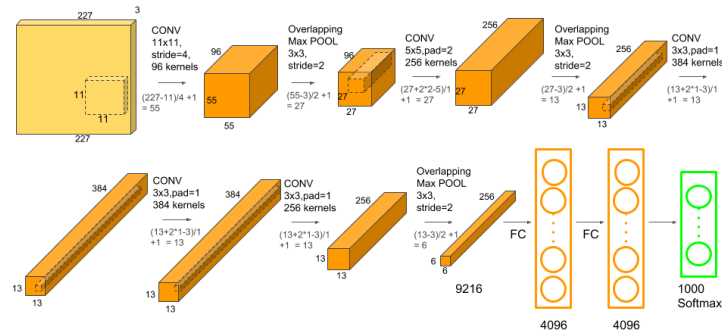


Figure 5.22: AlexNet [KSH12] architecture

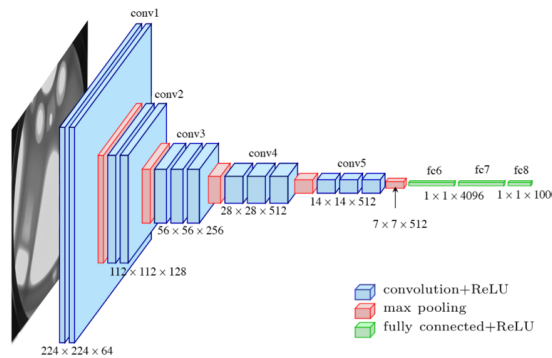


Figure 5.23: VGGNet [Kar14] architecture

the effectiveness of deep learning for large-scale visual recognition tasks, setting new standards for what CNNs could achieve in computer vision.

Following the success of AlexNet, VGGNet [Kar14], introduced by researchers from the Visual Graphics Group at Oxford in 2014, further advanced CNN architectures by emphasizing the importance of network depth. VGGNet is renowned for its simplicity and uniformity. It features an architecture with tiny (3x3) convolution filters throughout its layers, allowing it to go deeper—up to 19 layers than previous models like AlexNet. This depth, combined with its repetitive stacking of convolutional layers followed by a pooling layer, enabled the network to learn more complex features at various scales, thereby significantly improving Accuracy in large-scale image recognition tasks. VGGNet’s architecture (see figure 5.23) enhanced the development of deep learning models by demonstrating that depth is crucial for achieving high performance and set a new standard for how neural networks could be systematically scaled and designed. Its approach to handling large volumes of data and training on multiple GPUs has made it a foundational model for many subsequent innovations in computer vision.

All the above CNN architectures were discussed before falling into the category of plain networks (without skip or residual connection). Many studies [LXT+18], have shown that switching from plain to networks with residual connections prevents the model from getting stuck in the local minima, see figure 5.26. It also helps the model to backpropagate for gradients in a very deep network where the gradient vanishes the more profound the network gets; here is a visualization of the loss function visualized by a plain vs a network with the residual connection.

Building on the depth-focused innovations of VGGNet, Residual Networks [HZRS16] (ResNets) introduced by Kaiming He and colleagues in 2015 further advanced deep neural architectures by addressing the limitations seen in very deep networks. While VGGNet utilized small (3x3) convolution filters to construct deeper networks of up to 19 layers, it encountered vanishing gradients and performance degradation issues, see figure 5.27 directly taken from the ResNet [HZRS16] paper. ResNets effectively countered these challenges by incorporating skip connections that allow inputs to layers to bypass one or more layers and be added directly to their outputs.

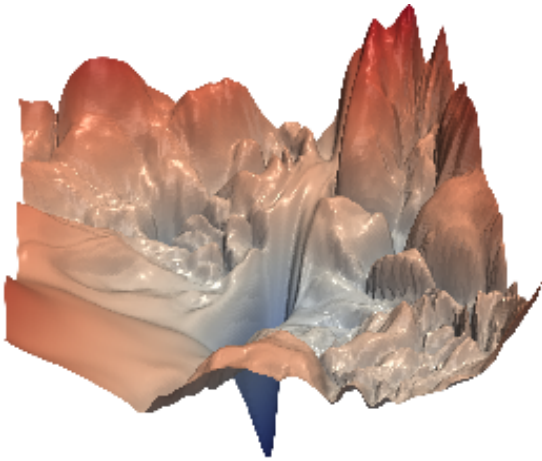


Figure 5.24: (a) Loss surface of a plain network (VGG)

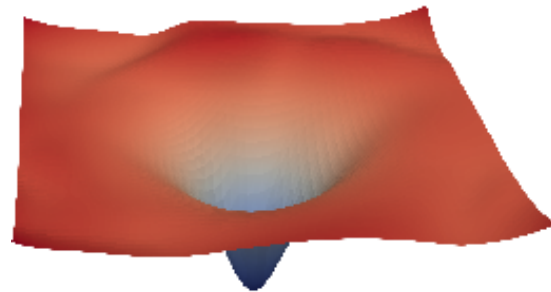


Figure 5.25: (b) Loss surface with a network with skip connections [LXT⁺18]

Figure 5.26: Loss surfaces of ResNet-56 with/without skip connections.

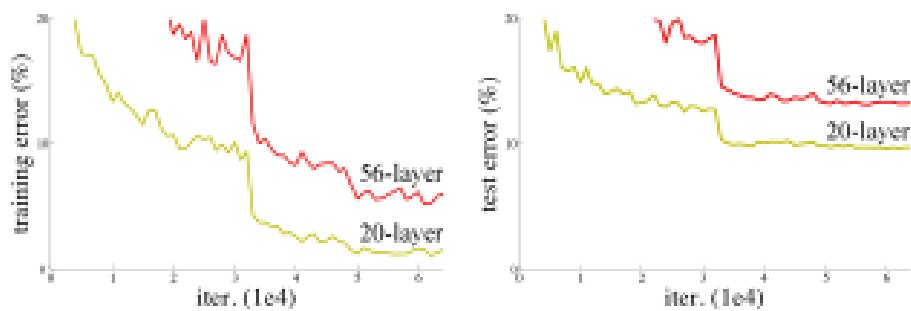


Figure 5.27: Combating gradient vanishing with skip connections

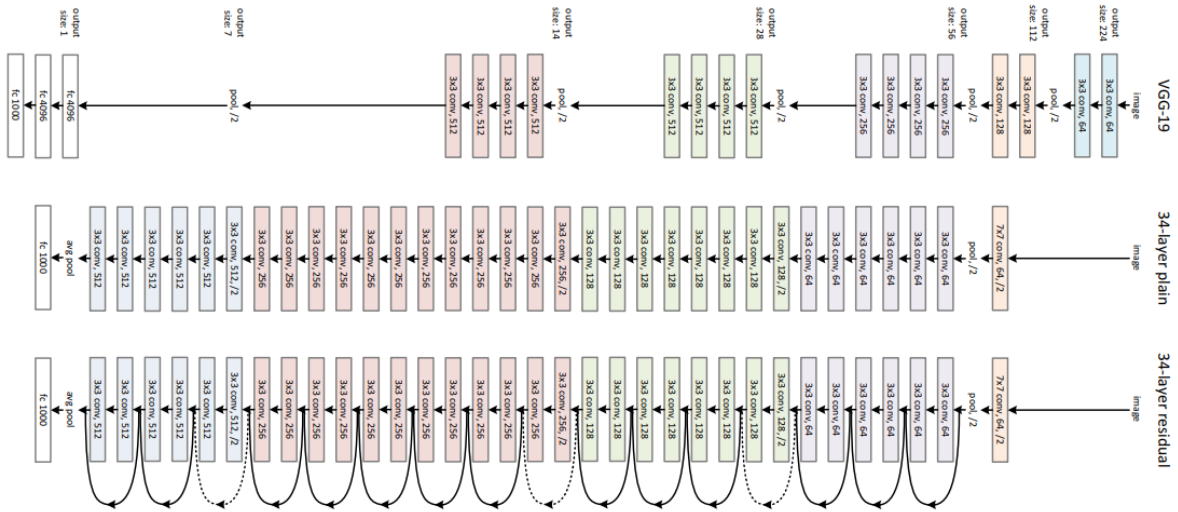


Figure 5.28: Resnet Architecture [HZRS16] (34 layered) incorporating skip connections

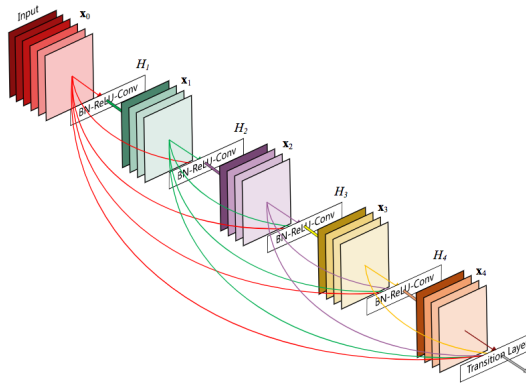


Figure 5.29: DenseNet Architecture [HLVDMW17] (skip connections with every preceding layer)

These skip connections help mitigate the vanishing gradient problem (see figure 5.26) and facilitate the training of networks significantly deeper than VGGNet—extending to over 100 layers. This innovation allows ResNets to train deeper models without increasing training complexity and reducing efficiency. By enabling the network to learn identity functions efficiently, ResNets ensure that adding more layers does not harm training performance, thus addressing the degradation problem prevalent in prior models like VGGNet (see figure 5.27). Consequently, ResNets have demonstrated superior performance on challenging datasets such as ImageNet, showcasing their ability to scale depth effectively while maintaining or reducing parameter count compared to previous architectures (see figure 5.28).

Building upon the architectural innovations introduced by ResNets, Dense Convolutional Networks (DenseNets) [HLVDMW17], developed by Huang et al. in 2017, marked a further advancement in deep neural network design by addressing some inefficiencies observed in ResNet structures. While ResNets uses skip connections to alleviate the vanishing gradient problem and enable the training of deeper networks, DenseNets takes this concept further by connecting each layer to every subsequent layer in a feed-forward fashion (see figure 5.29).

This dense connectivity ensures that each layer receives the collective knowledge of all preceding layers, which enhances feature propagation and reuse across the network. This results in considerable improvements in training efficiency because each layer has direct access to the gradients from the loss function and the original input signal, leading to more accessible and more effective training of deep networks.

DenseNets also demonstrate superior parameter efficiency. Unlike ResNets, where each layer needs to learn new features independently, DenseNets requires fewer parameters. Each layer can leverage the feature maps generated by all previous layers, thus reducing redundancy and enhancing feature diversity

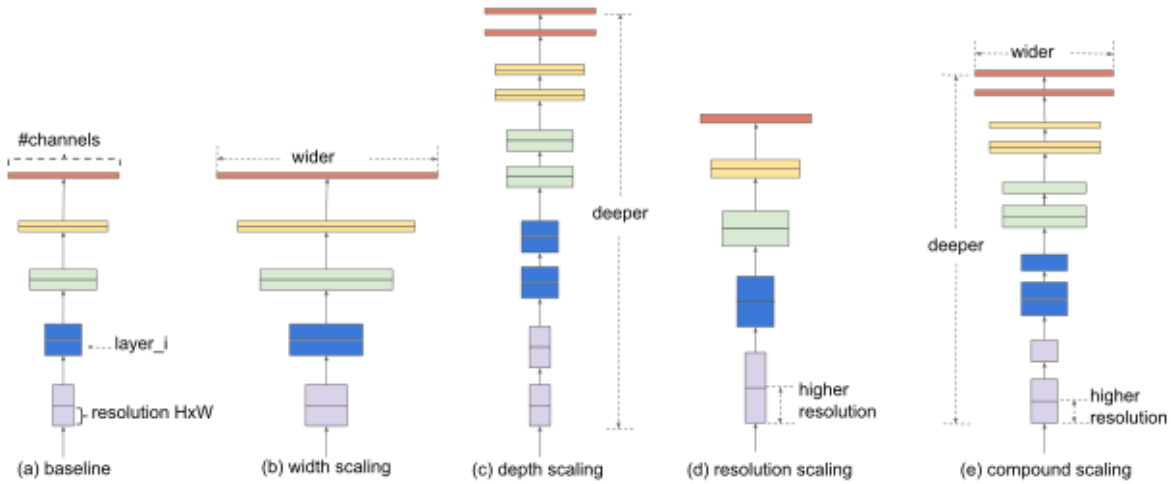


Figure 5.30: Compound Scaling: scaling depth, width, resolution wrt each other [TL19]

across the network. This efficient use of parameters helps combat overfitting and enables DenseNets to scale more effectively in depth and complexity without a commensurate increase in computational burden.

Neural Architecture Search Networks (NASNets) [ZL17] represent a significant advancement in the field of neural networks, introducing an automated approach to designing network architectures. Developed by Zoph et al. At Google, NASNets utilize reinforcement learning to automatically find the most effective network architecture for a given task. This method shifts the burden of architectural decisions from human experts to an automated system, which explores a predefined space of potential architectures and optimizes performance on a specific dataset.

The main advantage of NASNets over previous architectures like DenseNets and ResNets is their ability to tailor architectures to specific tasks, leading to higher efficiency and better performance. The process of architecture search also ensures that NASNets are not just deeper but more thoughtful, optimizing for computational efficiency and parameter usage. As a result, NASNets often outperform hand-designed models in speed and Accuracy, making them highly effective for a wide range of applications in computer vision and beyond.

EfficientNet [TL19], introduced by Mingxing Tan and Quoc V. Le at Google Research in 2019, brings a new perspective on scaling deep neural networks that distinguishes it from previous models like NASNet. EfficientNet employs a novel approach called compound scaling (see figure 5.30), which uses a compound coefficient to systematically scale up CNNs across three dimensions: depth, width, and resolution. This method is a strategic departure from approaches like NASNet, which primarily focuses on optimizing network architecture through a Neural Architecture Search (NAS) without a fixed rule for scaling.

While NASNet [ZVSL18] leverages reinforcement learning to discover the most efficient architecture for a given task automatically, EfficientNet provides a more formulaic and predictable method for scaling network size (see figure 5.31). This compound scaling technique is based on a set of fixed scaling coefficients obtained through a grid search that balances network depth, width, and resolution. The result is a family of models (EfficientNet-B0 to B7) that outperform other architectures like NASNet by achieving higher Accuracy and efficiency on benchmarks such as ImageNet.

The principal advantage of EfficientNet over NASNet is its ability to reach better performance with much fewer parameters and lower computational costs. EfficientNet models can scale to much larger sizes efficiently and are more adaptable to different computational and resource constraints, making them highly effective for a broad range of applications in fields requiring efficient and powerful image processing capabilities.

EfficientNetV2 [TL21], an evolution of the original EfficientNet, further refines the concept of model scaling and efficiency. Introduced by the same team at Google Research in 2021, EfficientNetV2 addresses some of the practical deployment issues faced by its predecessor, particularly around training speed and adaptability across different scales. This new version innovates primarily on making the model faster to train and more robust across a broader range of image sizes.

EfficientNetV2 incorporates advancements in training methodologies, such as progressive learning, where the model starts training with smaller images and gradually increases image size, which speeds up the training process significantly compared to the original EfficientNet. This method reduces overfitting

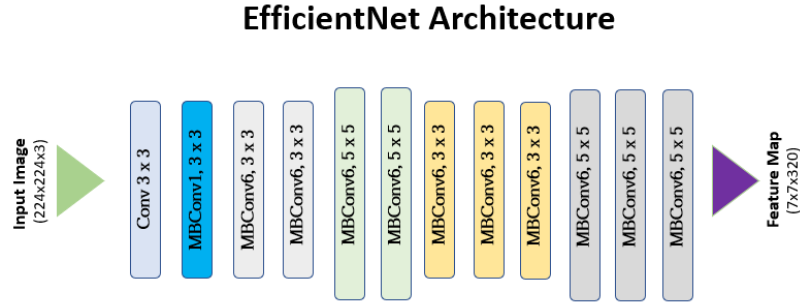


Figure 5.31: Efficientnet [TL19] Architecture

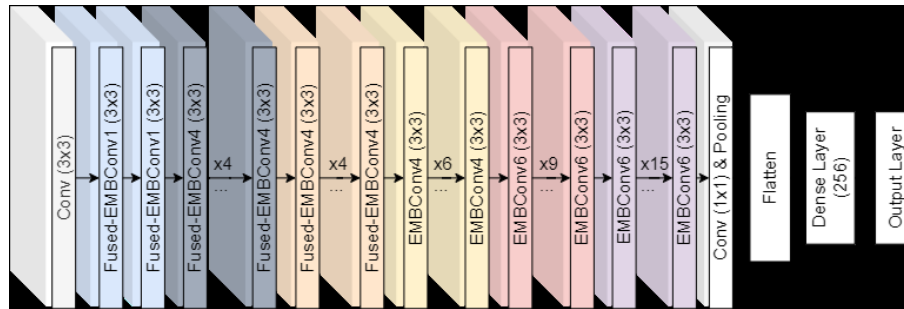


Figure 5.32: Efficientnetv2 [TL21] Architecture

and enhances the model’s ability to handle different input sizes naturally without needing extensive finetuning.

Compared to NASNet and even the first version of EfficientNet, EfficientNetV2 stands out by offering improved efficiency in parameter usage and computational performance and being quicker and less resource-intensive during training. These improvements are achieved through refined architecture (see figure 5.32) adjustments, including fewer but more powerful layers optimized for modern hardware. As a result, EfficientNetV2 sets new standards for Accuracy and efficiency in various computer vision tasks, making it highly suitable for real-world applications where both speed and accuracy are critical.

The paper *"A ConvNet for the 2020s"* [LMW⁺22] reexamines the design and performance of pure convolutional neural networks (ConvNets) in light of the rapid advancements and popularity of Vision Transformers (ViTs) [DBK⁺21] in computer vision tasks. The authors explore how to modernize a standard ResNet to match the design of a vision Transformer (see figure 5.33), uncovering several vital components that contribute to the performance difference between the two. This exploration leads to the development of a family of models named ConvNeXt, which, despite being constructed entirely from standard ConvNet modules, demonstrate competitive performance with Transformers on a variety of vision tasks, including ImageNet classification, COCO object detection/segmentation [LMB⁺14], and ADE20K semantic segmentation.

ConvNeXt models notably retain the simplicity and efficiency of standard ConvNets while also being extremely straightforward to implement due to their fully convolutional nature for training and testing. These models achieve up to 87.8% top-1 Accuracy on ImageNet and surpass Swin Transformers in COCO detection and ADE20K [ZZP⁺17] segmentation tasks. The study hopes to challenge common beliefs about the supremacy of Transformers and reinvigorate interest in the capabilities and importance of convolutions in computer vision.

The researchers present a roadmap of their exploratory process, detailing the systematic transformation from a standard ResNet towards a Transformer-like ConvNet [WDH⁺23], maintaining the network’s simplicity. Key findings include the replacement of ReLU with GELU [HG16] activations, adopting depthwise separable convolutions, and using LayerNorm [BKH16] instead of BatchNorm, among others. Their empirical evaluations on ImageNet and other downstream tasks confirm that ConvNeXt models are efficient, exhibiting comparable or better inference throughputs and requiring less memory than

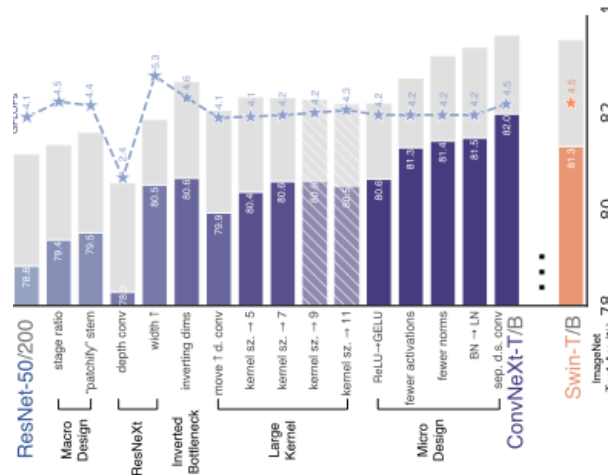


Figure 5.33: ConvNeXt [LMW⁺22]: Searching for various techniques to scale a Resnet architecture to compete with a Swin Transformer

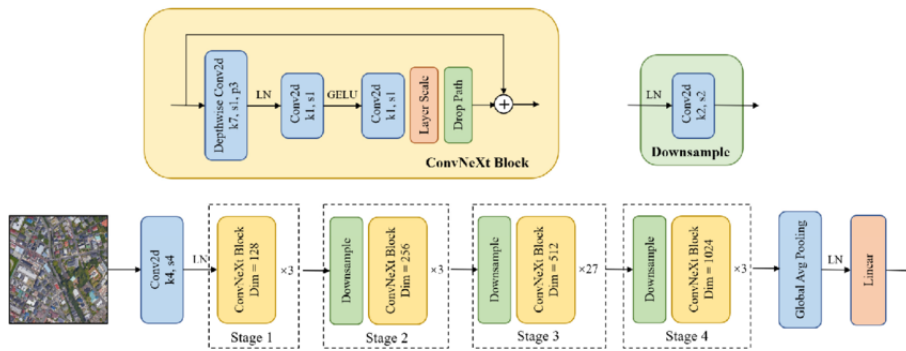


Figure 5.34: ConvNeXt Architecture [WDH⁺23]

Swin Transformers [LLC⁺21] and ViTs [DBK⁺21], especially when considering large-scale training and higher-resolution inputs. ConvNeXt’s robust performance, including on out-of-distribution data, suggests that it is a promising architecture for the future of visual representation learning.

Unlike EfficientNetV2 and previous architectures focusing on optimizing convolutional layers through scaling and neural architecture search, ConvNeXt challenges some long-standing CNN conventions. It incorporates transformer-like elements, such as Layer Scale and larger kernel sizes, alongside simplifying the convolutional block design (see figure 5.34). This adaptation improves feature extraction and learning capabilities, particularly in handling diverse and complex image datasets.

5.3.3 Preliminary Experimental results – CNNs

Before exploring complex architectures for our classification task, it is beneficial to establish a simple baseline CNN. This basic model consists of:

- Convolutional Layer: 16 kernels of size 3x3 to extract features.
- Pooling Layer: Max pooling to reduce spatial dimensions.
- Flattening Layer: Converts data to a 1D vector.
- Dense Layer: 84 neurons with ReLU activation.
- Output Layer: Softmax with three classes for classification.

A simple layer-wise visualization to implement VGG architecture in TensorFlow is provided, see figure 5.37.

Table 5.8: Comparing a baseline CNN Architecture with more recognised architectures

Architecture	Unique features	Loss (Epoch 1)	Accuracy	#params
Baseline CNN ¹		0.6604	0.7184	321M
Le net ²		0.6368	0.7258	115M
Alex net ³		0.605	0.7339	894M
VGG ⁴		0.6483	0.7238	149M

¹ Conv → pooling → dense → dense: Relu activations, maxpooling, SGD optimizer.

² Original paper had no padding; 'same' padding applied with relu activation.

³ No dropouts, local response normalization with several fully connected layers.

⁴ The Model uses a simplified configuration without additional features.

Table 5.9: Performance comparison of VGG and AlexNet (with 50% dropout rate)

Architecture	Loss (Epoch 1)	Accuracy	#params
Alex net	0.6278	0.7264	894M
VGG	0.73	0.6976	149M

We will monitor the cross-entropy loss on validation data as our primary metric, training the model for just one epoch to establish a performance baseline. This setup uses ReLU activations consistently across the model and employs the SGD optimizer with an initial learning rate of 0.01, providing a straightforward evaluation of the model's initial learning capability.

As we progressed through different CNN architectures for our model evaluation, we observed an interesting trend: model accuracy improved, moving from Baseline CNN to LeNet and then to AlexNet. However, despite the greater complexity and depth of VGG-19, it performed worse than the simpler LeNet architecture. This underperformance may be attributed to issues common in deeper networks like VGG-19, such as overfitting and gradient vanishing. To address these challenges, we implemented a couple of strategies:

- **Dropout:** We introduced a dropout layer with a 0.5 drop chance after every dense layer in the VGG-19 architecture. Dropout helps in reducing overfitting by randomly setting the output features of hidden units to zero during training. This variation forces the network to learn more robust features that are not reliant on specific weights of neurons.
- **Batch Normalization:** We added batch normalization layers after each dense layer. This technique normalizes each layer's inputs to have zero mean and unit variance. This helps stabilize the learning process and has been shown to significantly improve the training of deep networks by reducing the internal covariate shift.
- **Adam Optimizer:** We switched to using the Adam optimizer with a lower initial learning rate of 0.001. Adam adjusts the learning rate during training and is known for its effectiveness in converging faster than traditional stochastic gradient descent, especially when dealing with problems that are sensitive to the learning rate and require finer adjustments.

These modifications are intended to mitigate the specific issues encountered with the VGG-19 model in our setting, and they are reflective of broader strategies that can be beneficial when training complex neural networks on challenging tasks.

Introducing adjustments like dropout and batch normalization initially seemed promising to address issues like overfitting and gradient vanishing in VGG-19; however, these changes led to unintended consequences. With a 50% dropout rate, a significant number of neurons are turned off at each step during training, which ideally requires the model to be trained over a much larger number of epochs to learn from the data effectively. This extended training demands considerable GPU resources and time, which conflicts with our goal of maintaining efficiency in the exploratory phase, see table 5.9.

The concept of dropout was pioneered in 2014 in the master's thesis of Nitesh Srivastav under the supervision of Geoff Hinton. This technique was not a part of the design in earlier conventional networks

Table 5.10: Performance of a ResNet model with skip connection and more hidden layers

Architecture	Loss (Epoch 1)	Accuracy	#params
Alex Net *	0.605	0.7339	894M
Resnet50 (no pooling)	0.6185	0.7323	23M

Table 5.11: Impact of pooling and ImageNet initialization

Architecture	Loss (Epoch 1)	#params
Resnet50	0.6185	23M
Resnet(avg-pooling)	0.6305	23M
Resnet50 (with max pooling layer)	0.6319	23M
Resnet50 (max pooling with Imagenet weights)	0.747	23M

like AlexNet and VGG. However, architectures like ResNet incorporate skip connections and integrate dropout layers. These skip connections in ResNet are crucial for mitigating issues like gradient vanishing by allowing gradients to flow more freely through the network.

Given this, we opted to test the ResNet architecture, specifically ResNet50, which includes these beneficial skip connections. Using the Adam optimizer with ResNet is expected to address the shortcomings observed with VGG, potentially leading to better performance than traditional plain networks. The ResNet50 model, with its deep layers and skip connections, theoretically should offer enhanced learning capabilities, particularly in handling the complexities of our dataset without the training inefficiencies seen in VGG.

Thus, the shift to ResNet50 marks a strategic pivot towards leveraging modern architectural advancements that integrate robust mechanisms like dropout and skip connections to improve training dynamics and model performance. The transition to ResNet yielded performance comparable to AlexNet but with significantly fewer parameters, offering the potential to expedite inference times by approximately 40 times as evidenced by table 5.10.

Next, we explored the effects of utilizing ImageNet weights for initialization despite the dissimilarity between our training data and ImageNet. As anticipated, employing ImageNet weight initialization resulted in an uptick in the loss function.

It is a common assumption that pooling layers might not significantly impact model performance. However, they are intended to enhance generalization by reducing sensitivity to exact feature locations in the input. To test this, we extended the training duration of our model to three epochs. The results showed that models without any pooling experienced an increase in loss after the initial epoch, failing to continue decreasing the loss. In contrast, models employing average pooling continued to converge, suggesting that average pooling helps the network generalize better by abstracting features more effectively, see table 5.12.

This behaviour underscores the importance of pooling layers in CNN architectures, particularly when managing to overfit and enhancing the ability of the model to generalize from the training data to unseen data. By reducing the spatial size of the representation, pooling layers help decrease the number of parameters and computations in the network, thereby not only improving efficiency but also reducing the risk of overfitting by summarizing the features in the input.

In our comparative analysis of neural network architectures, we tested the ResNet model against the DenseNet model. Despite DenseNet’s innovative architecture, which theoretically allows for more efficient use of parameters through its densely connected layers, it did not perform as well as the ResNet

Table 5.12: Impact of pooling on generalization, after more epochs

Architecture	Loss (Epoch 3)	Accuracy	#params
Resnet50 (no pooling)	0.6534	0.7294	23M
Resnet 50 (avg pooling)	0.6174	0.7424	23M

Table 5.13: Comparative Analysis of CNN Architectures with Average Pooling

Architecture (avg pooling)	Loss (epoch 1)	Accuracy	#params
ResNet50	0.6185	0.7423	23M
DenseNet 169	0.7252	0.6942	12M
NASNet	OOM	OOM	85M
EfficientNet (B0)	0.5741	0.7500	4M
ConvNext (tiny)	0.7521	0.6862	27M
ConvNext (large)	0.7554	0.6862	196M

model in our specific application. However, it is crucial to note that DenseNet operates with nearly 50% fewer parameters than ResNet, which underscores its potential for efficiency in scenarios where parameter economy is critical, see table 5.13.

We also attempted to train with NASNet but faced out-of-memory (OOM) issues due to the model’s demands at the same batch size used for other models, preventing a fair comparison under the same computational constraints.

EfficientNet, known for its compound scaling approach—systematically scaling up the depth, width, and resolution of the model—outperformed all previous models. It also maintains a smaller parameter footprint, about five times smaller than some of the competing models, making it exceptionally suitable as a backbone network for various applications.

Lastly, unfortunately, the ConvNext model, a newer architecture designed to generalize well across different settings, did not perform well in our tests. It failed to generalize effectively, highlighting that newer models, despite their theoretical advantages, might require more specific tuning or may not always translate to better performance across all data types.

This series of tests underscores the importance of selecting the exemplary architecture based on the specific needs and constraints of the project, including computational resources, model efficiency, and the particular characteristics of the dataset.

5.3.4 Evaluation on Large Dataset

Based on the testing outcomes of table 5.13 on small dataset, we have chosen to utilize EfficientNetB0 as the baseline architecture for the sizeable multimodal dataset, primarily due to its parameter efficiency, which facilitates faster inference times. This choice aims to enhance the practicality of deploying this model in real-world applications where quick inference is crucial. Key implementation details include:

- **Architecture Decision:** EfficientNetB0 is selected for its balance between Accuracy and efficiency, making it ideal for handling diverse data types and sizes efficiently.
- **Operational Settings:** To further optimize the model’s performance and computational requirements, we adjust the input size to 400x1400. This modification is particularly beneficial when dealing with larger image patches, allowing for cropping to this specified size. Cropping helps reduce the computational load by ensuring that all inputs are uniform, potentially speeding up the inference process.
- **Class Setting and Optimization:** The model will now operate in a 4-class setting, including a category for overlaps, which adds complexity and realism to the classification tasks. To manage this, we use the LAMB [YLR+20] optimizer with a learning rate of 2e-5. The LAMB optimizer is chosen for its effectiveness in handling large batches and datasets, which is suitable given the new class settings and the challenges of managing more extensive and varied data.

The integration of average pooling in the EfficientNet architecture significantly reduces the parameters by roughly 30% as compared to no pool, while maintaining or enhancing Accuracy. This adjustment makes the model more efficient and preserves its effectiveness, making it a favourable option for complex tasks requiring high Accuracy with a lean model structure.

Further advances are observed with EfficientNetV2[TL21], which, despite being similarly sized to its predecessor, offers accuracy and inference speed improvements. This version is particularly notable for its enhanced performance capabilities, highlighted in the EfficientNetV2 paper, which reports training times

Table 5.14: Performance comparison of several Vision models varying across different sizes

Model	#Batches	Inf. time (ms/step)	Accuracy (%)	Mean F ₁ (%)	#Params
Dummy	-	-	59.41	24.85	-
EfficientNetB0	5	29	65.27	45.85	6.9M
EfficientNetB0_max	5	35	58.22	25.64	4.0M
EfficientNetB0_avg	5	34	62.93	39.46	4.0M
EfficientNetB4_avg	5	61	65.87	47.39	17.6M
EfficientNetB7_avg	5	145	61.22	42.30	64.1M
EfficientNetV2s_avg	5	70	59.42	24.84	20.3M
EfficientNetV2m_avg	5	94	64.02	42.64	53.2M
EfficientNetB4_avg	9	88	68.47	54.24	17.6M
EfficientNetV2s_avg	9	71	59.81	26.76	20.3M
EfficientNetV2m_avg	9	92	69.44	59.96	53.2M

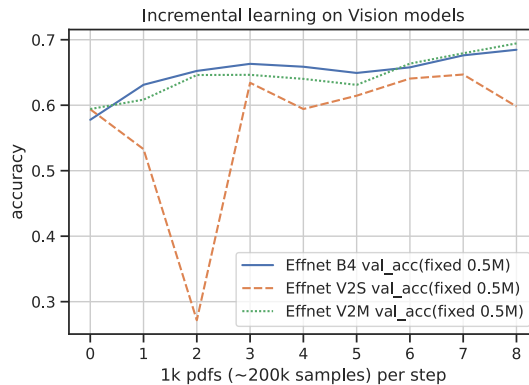


Figure 5.38: Accuracy of vision models on finetuning task concerning number of batches

that are 5x to 11x faster compared to previous iterations. The use of EfficientNetV2 allowed us to save training time which was utilised in further exploration of different sizes of Efficientnetv2 variants.

Overall performance of the CNN-based vision model can be seen in table 5.14. We evaluate our models on the large validation data after incrementally processing each batch of approximately 1,000 PDFs. This allows us to assess the model’s effectiveness in real-time and make necessary adjustments based on its performance to cut off training when convergence is suspected, see figure 5.38.

5.4 Text Modality

Indeed, while font and visual elements contribute some information towards classifying paragraphs in scientific texts, the actual text content remains the most critical modality. The textual content provides direct and contextual clues essential for accurately determining the label of a paragraph, such as ‘Introduction’, ‘Methodology’, ‘Results’, or ‘Discussion’.

In the referenced Doceng paper [MPS21], language models previously trained on general English datasets were finetuned to classify text at a line level for scientific content. This process involved adapting these models to recognize and interpret the specific language and format typical of scientific writing.

Building on this approach, applying these language models at the paragraph level could enhance performance significantly. At this level, models benefit from a much broader context, which is crucial for understanding the flow and purpose of entire paragraphs rather than isolated lines. This allows the models to capture better the logical structure and thematic significance of the text, improving their ability to accurately classify each paragraph according to its role within the document.

Thus, the next step involves extending these techniques from line-level to paragraph-level classification, expecting that the broader context provided at this level will lead to more accurate and nuanced model predictions. This adaptation is crucial for tasks like document summarization, thematic analysis, and information extraction, where understanding the full context and connectivity of the text significantly

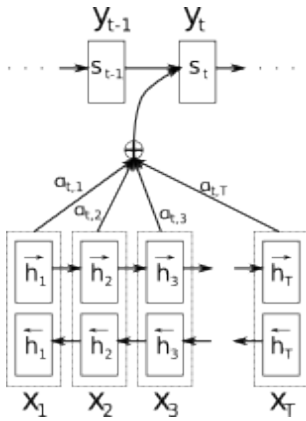


Figure 5.39: (a) BiLSTM models with assigned attention weights

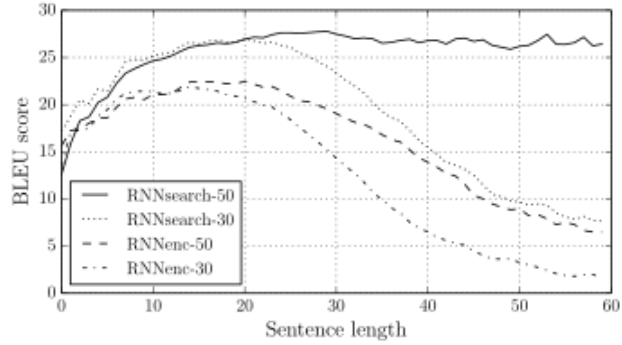


Figure 5.40: (b) Performance degradation for long sequence translation when using LSTM (without attention weights)

Figure 5.41: Yoshua's [BCB15] early work on Attention mechanism

impacts performance.

5.4.1 Related Work

Before the advent of large language models, NLP tasks often relied on relatively simplistic methods for feature generation, such as Bag of Words (BOW) and TF-IDF (Term Frequency-Inverse Document Frequency). These techniques were effective for some applications but fell short in capturing deeper linguistic semantics.

The landscape began to shift with the introduction of Word2Vec[MCCD13] in 2013. Developed by researchers at Google, Word2Vec represented a significant advancement by using an autoencoder model, specifically a skip-gram architecture, to capture the semantic meaning of words. This model was trained by predicting surrounding words in a sentence, which allowed it to learn word associations from large amounts of text data.

However, Word2Vec and similar models still struggled with context-dependency—the meaning of words can change based on the surrounding words, a nuance these models could not fully grasp. To address limitations in capturing context and long-term dependencies, LSTM (Long Short-Term Memory) [HS97] networks became popular. Notably discussed in the ELMo [SWWP⁺21] (Embeddings from Language Models) paper, LSTMs improved upon earlier models by better handling data sequences, such as long sentences in text translation tasks.

Despite the progress with LSTMs, they still faced challenges capturing long-term dependencies and specific relationships between words. An influential paper [BCB15] by Yoshua Bengio and colleagues introduced a novel approach that involved assigning weights to each LSTM cell to leverage bidirectional context, enhancing the model's ability to understand relationships and dependencies between words across more extensive texts.

This Evolution set the stage for developing more sophisticated models that integrate these advancements, leading to the current generation of large language models like BERT [DCLT19] and GPT [RNS⁺18], which use mechanisms like attention and transformers to capture nuanced and context-dependent text meanings more effectively than ever before. These models learn from the immediate context and adjust their understanding based on the broader discourse, resulting in significantly enhanced language understanding and generation capabilities.

Before the release of the influential paper [VSP⁺17] "Attention is All You Need" in 2017, Yoshua Bengio and his team had already explored the development of formal attention mechanisms to capture relationships between different types of inputs better. One notable application of these early attention concepts was in image captioning. A significant milestone in this area came in 2015 with [XBK⁺15] that presented an early proof of concept linking specific words to corresponding segments of an image. This early work demonstrated the potential of attention mechanisms to enhance model capabilities by focusing on relevant parts of the input data, paving the way for the more refined and robust self-attention mechanisms introduced in "Attention is All You Need."

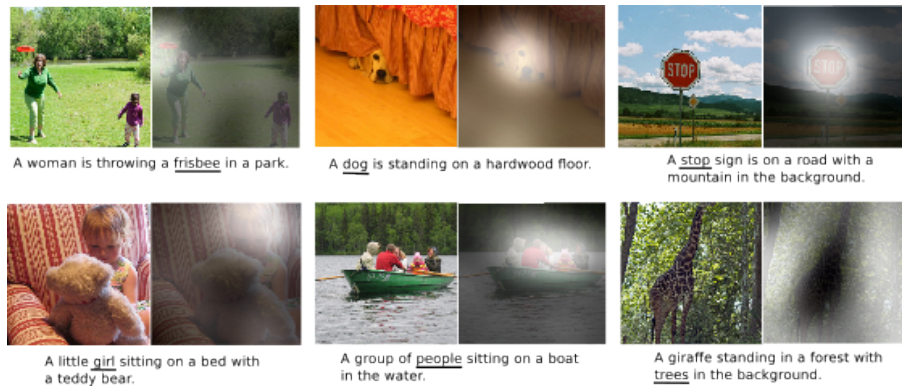


Figure 5.42: Attention mechanism [XBK⁺15] in cross-modal setting

[VSP⁺17] introduced the Transformer model, which relies entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. This marked a substantial shift in how models processed sequential data, offering a new architecture that was particularly effective for a wide range of NLP tasks and became foundational for subsequent advancements in the field. The self-attention mechanism allows the model to weigh the importance of different words within a sentence, regardless of their positional distance from each other, leading to more nuanced and contextually rich interpretations of text.

The attention mechanism, particularly the Multihead attention, fundamentally transformed how models understand the relationships and dependencies between tokens in sequences. Initially proposed with the concepts of Queries (Q), Keys (K), and Values (V), the attention mechanism uses a learnable matrix to capture the interdependencies among tokens. This was a critical advancement over previous models, allowing for a richer, more nuanced understanding of text. Some of the important features that were scaled to build the modern transformer architecture include:

- **Development of Multihead Attention:** This mechanism was further enhanced by parallelizing the operations, leading to what is now known as Multihead attention. This advancement was detailed in the seminal "Attention is All You Need" paper, introducing a simple yet powerful encoder-decoder architecture. The model incorporated a mathematical function based on sine and cosine to embed positional information of tokens, which is crucial for maintaining the sequence order without recurrent networks.
- **Evolution into BERT:** The development of BERT (Bidirectional Encoder Representations from Transformers) marked a significant evolution of the Multihead attention concept. Detailed in its 2018 paper, BERT applied the Transformer's encoder architecture to a wide range of NLP tasks beyond just translation, as evaluated on the GLUE [WSM⁺18] dataset. One of the notable changes introduced with BERT was the shift from hardcoded mathematical functions (such as Sine, Cosine) for positional encoding to learnable positional embeddings, enhancing the model's ability to understand and adapt to different textual contexts.
- **Variants of BERT:** BERT was also designed in several configurations, with varying numbers of encoder blocks to suit different computational resources and performance needs. These variants offered improvements in handling more complex tasks and larger datasets, demonstrating significant performance gains over the base model configuration.

These advancements highlight the dynamic nature of NLP research, where ongoing innovations push the boundaries of what is possible with machine learning models in understanding and processing human language.

The Evolution of NLP models from GPT [RNS⁺18] to BERT has showcased significant differences in their design and applications, particularly in how they employ the attention mechanism. While GPT models are unidirectional, allowing tokens to only attend to previous tokens, BERT uses a bidirectional approach. This bidirectionality enables BERT to excel in encoding tasks such as classification, making it highly effective for multimodal approaches where features from the [CLS] token are often used for feature extraction. However with more modern transformers, changes were made on the design aspects, some of them are mentioned below:

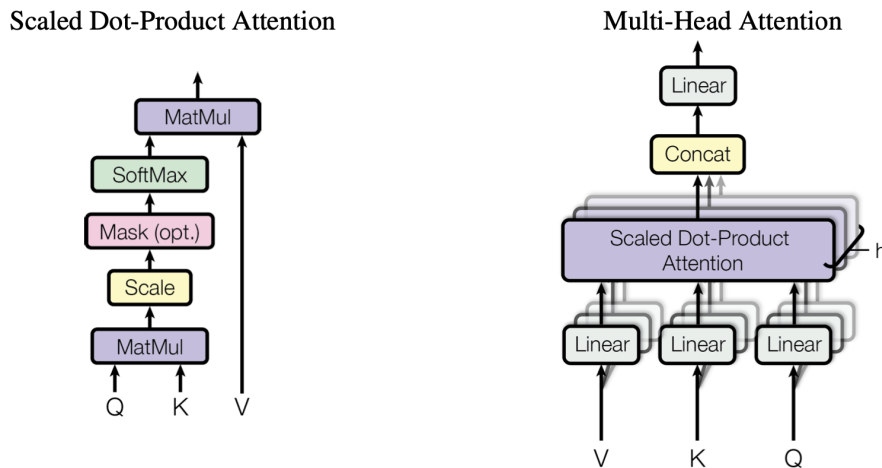


Figure 5.43: Attention mechanism [VSP⁺17] used in transformer models

1. GPT vs. BERT: GPT's generative abilities made it popular for tasks requiring forward prediction of text, but its unidirectional nature limits its effectiveness in tasks that benefit from understanding the full context of input, such as classification. BERT's bidirectional mechanism allows it to perform better on these tasks by understanding the context from both directions.
2. Pretraining Techniques: BERT's training involves steps like Masked Language Modeling (MLM) and Next Sentence Prediction (NSP), which are designed to enhance its understanding of language semantics in a self-supervised manner. This pretraining helps the model capture nuanced relationships between words based on extensive text data and is analogous to how humans build foundational language skills before specializing.
3. Domain-Specific Models: The success of BERT inspired the creation of specialized models like SciBERT, BioBERT, and MathBERT, each pre-trained on domain-specific corpora to perform better on related tasks. For instance:
 - SciBERT [BLC19]: Optimized for scientific texts, making it relevant for applications in computer science and related fields.
 - BioBERT [LYK⁺20]: Tailored for biomedical text, hence its divergence from general scientific applications.
 - MathBERT [PYGT21]: Specifically pre-trained on mathematical texts and formulas, showing effectiveness in mathematical tasks but requiring \LaTeX sources.
4. Architectural and Training Optimizations: Post-BERT developments include models like RoBERTa [LOG⁺19] (Robustly optimized BERT approach), which enhances BERT by training on even more data and optimizing its training process. DistilBERT focuses on reducing the model size and training/inference time through techniques like knowledge distillation, addressing the computational efficiency challenges faced by BERT.

These advancements highlight the dynamic nature of NLP research and the continuous efforts to adapt and optimize models for various specific tasks and more prominent, more complex datasets. The field is rapidly evolving, with each new model building on the successes and learning from the limitations of its predecessors, pushing the boundaries of what is possible in natural language understanding and generation.

For a significant period, there was a prevailing belief in the machine learning community that a model with more parameters typically performs better. This hypothesis found substantial support with the release of OpenAI's GPT-3 in 2020, which featured 175 billion parameters. GPT-3 [BMR⁺20] demonstrated remarkable capabilities, especially in generating coherent and contextually relevant text, which seemed to validate that larger models could perform better.

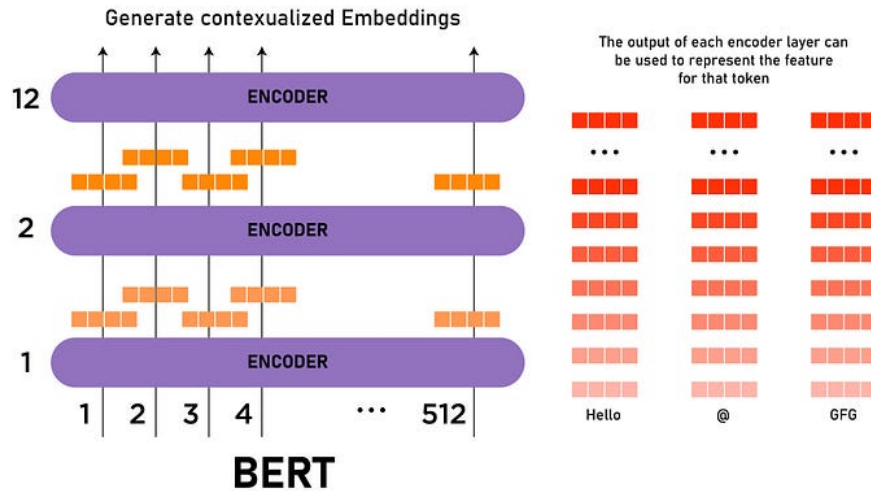


Figure 5.44: Typical encoder block in BERT [DCLT19]

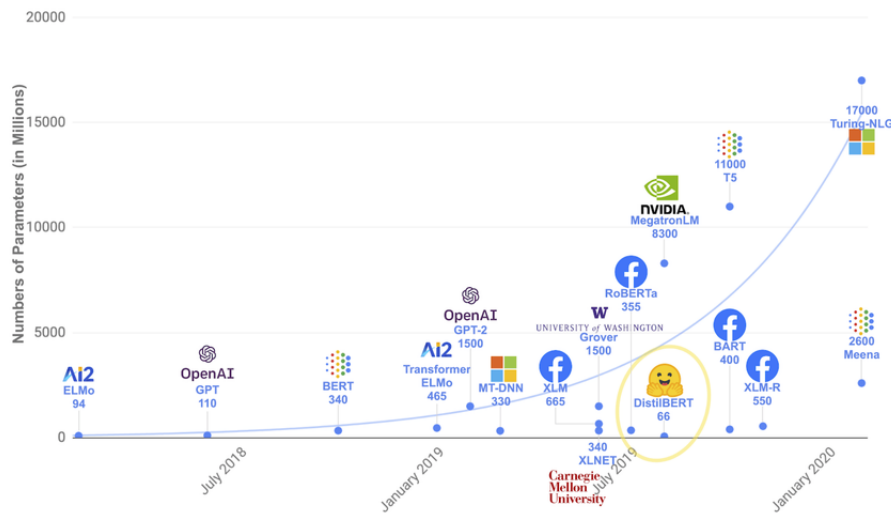


Figure 5.45: Evolution of the number of params in typical LLMs, reproduced from <https://huggingface.co/learn/nlp-course/chapter1/4>

This idea was pushed even further when Nvidia released the Megatron [SPP⁺19] model in 2022 (for comparison purposes refer 5.45) which boasted up to 530 billion parameters. This development underscored a trend towards building ever larger models in the pursuit of enhanced AI capabilities. The scale of these models brought about advancements in natural language processing and other AI domains, suggesting that, at least for specific applications, increasing model size could lead to significant performance improvements. However, this approach also raised questions about efficiency, cost, and the diminishing returns of simply scaling up model parameters.

Pretraining huge language models (LLMs) with extensive numbers of parameters are notably resource-intensive, not just in terms of computational power but also in cost. These large models, while powerful, often struggle with inefficiency during inference, particularly when used as a feature extraction backbone for other tasks. This inefficiency is exacerbated by the quadratic time complexity of the attention mechanism used in transformers, where n represents the number of tokens, leading to significant computational demands as model size increases.

Interestingly, the relationship between model size and effectiveness is not strictly linear. While larger models have shown remarkable capabilities, the quality of training data and the breadth of the pretraining corpus frequently play more critical roles in the model's performance than sheer size alone. The Chinchilla [HBM⁺22b] model developed by Google DeepMind highlighted this by establishing a linear relationship

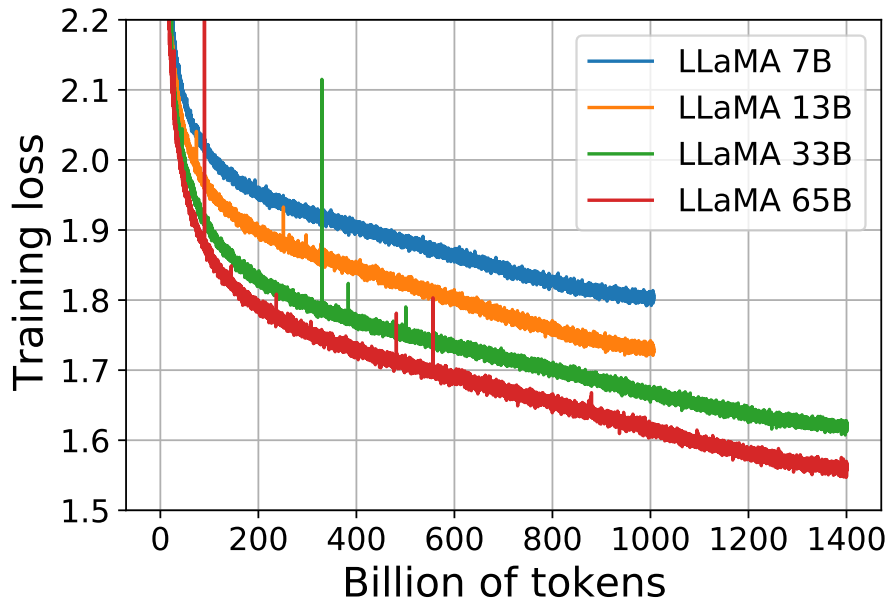


Figure 5.46: Training smaller models (LLAMA) [TLI⁺23] on more data based on chinchilla scaling laws

between the amount of training data and the optimal number of parameters. This finding suggested that when trained on denser or more extensive datasets, smaller models could outperform larger models that were relatively undertrained.

This concept, known as the Chinchilla scaling laws, indicates that data usage efficiency in training can be more significant than the number of parameters a model has. Many large models, including GPT-3, were identified as undertrained given their capacity, suggesting that there is potential to achieve more performance with smaller, more data-efficient models. This shift in understanding challenges the previous notion that *"bigger is always better"* in the context of neural network design and opens up new avenues for optimizing the balance between model size, training data quality, and computational efficiency.

LLaMA (Language Model from Meta AI) [TLI⁺23] in 2023 utilized the insights from the Chinchilla scaling laws to optimize the balance between model size and training data efficiency. By adopting these principles, LLaMA has been able to achieve performance on par with significantly larger models such as GPT-3 [BMR⁺20], which boasts 175 billion parameters, with just 7 billion parameters in the LLaMA model. This remarkable efficiency has set a new trend in the industry, where smaller foundation models are trained on larger datasets to provide versatile application capabilities. The loss function of LLaMA model showcases the trend that with more training data the loss progressively goes down, see chart 5.46.

By 2023, models were already competing on challenging datasets, showcasing impressive capabilities and hence finetuning was only subject to very limited use cases, some of the mentioned below:

1. Need for smaller foundation model: The smaller, more efficient models can be finetuned on particular and challenging tasks with fewer examples, to support few shot learning scenarios. They are also adaptable for zero-shot learning in simpler tasks, making them highly versatile.
2. Specialized Applications: These LLM's can be further be finetuned for tasks that deviate from general applications, such as coding or specialized mathematical tasks as in our context. As an example, CodeLLaMA [RGG⁺23], built on the LLaMA 2 model [TMS⁺23], has been tailored for coding-related tasks in an instruction-based (question-answer) setting. Continuing this trend, the authors of the LLEMA [ASP⁺23] (a distinct model not to be confused with LLaMA) further trained CodeLLaMA on the Proofpile-2 dataset, enhancing its formal theorem-proving capabilities without additional finetuning, see figure 5.47.

This change represents a significant shift towards using high-efficiency models that leverage vast data to achieve state-of-the-art results without training the model on scratch, instead using an already pretrained model as checkpoint and start gaining impressive abilities such as math, coding skills by further training on these model checkpoints.

Input:
 Let $f(r) = \sum_{j=2}^{2008} \frac{1}{j^r} = \frac{1}{2^r} + \frac{1}{3^r} + \dots + \frac{1}{2008^r}$. Find $\sum_{k=2}^{\infty} f(k)$.

LEMMA 34B solution:
 We have

$$\begin{aligned}
 \sum_{k=2}^{\infty} f(k) &= \sum_{k=2}^{\infty} \sum_{j=2}^{2008} \frac{1}{j^k} = \sum_{j=2}^{2008} \sum_{k=2}^{\infty} \frac{1}{j^k} = \sum_{j=2}^{2008} \frac{1}{j^2} \sum_{k=0}^{\infty} \frac{1}{j^k} = \sum_{j=2}^{2008} \frac{1}{j^2} \frac{1}{1 - \frac{1}{j}} \\
 &= \sum_{j=2}^{2008} \frac{1}{j(j-1)} = \sum_{j=2}^{2008} \left(\frac{1}{j-1} - \frac{1}{j} \right) \\
 &= \left(\frac{1}{1} - \frac{1}{2} \right) + \left(\frac{1}{2} - \frac{1}{3} \right) + \dots + \left(\frac{1}{2007} - \frac{1}{2008} \right) \\
 &= 1 - \frac{1}{2008} \\
 &= \boxed{\frac{2007}{2008}}.
 \end{aligned}$$

Final Answer: The final answer is $\frac{2007}{2008}$.

Figure 5.47: Induced math capabilities by LLEMA [ASP⁺23] model by training CodeLLaMA on the Proofpile-2 dataset.

The landscape of language models is seeing remarkable developments, with smaller, open-sourced models like **Orca Math** (8B parameters) demonstrating impressive capabilities. As of March 2024, Orca Math [MKRA24] has achieved state of the art performance of 86.81% on the GSM8-k [CKB⁺21] dataset, a mathematical understanding benchmark, which, while not quite reaching the 97% performance of the much larger GPT-4 [Ope23] with its rumoured 1.76 trillion parameters shows a significant competitive edge.

However, despite these advances, there are inherent challenges associated with deploying such powerful models:

1. **Feature Backbone:** Most large language models (LLMs) are specifically designed for natural language processing (NLP) tasks, where text is the sole modality. However, in our approach, we utilize LLMs as feature backbones to construct a multimodal model. Despite their impressive performance, these models have a very large number of parameters and substantial memory consumption, which presents significant challenges for gradient propagation during training and deployment at inference time.
2. **Data Source and Training Bias:** Since many models are trained on extensive internet-sourced datasets, there is an inherent risk that the model may simply be regurgitating information included in its training data. This raises questions about the true learning capability of the model, as highlighted by Yan LeCun, regarding whether the model is genuinely 'understanding' or merely 'restating' known information.
3. **Applicability to Non-Generative Tasks:** Large models are often designed to handle various generative tasks, from composing texts to answering queries. This generality can make them less suitable as specialized tools for non-generative tasks such as Named Entity Recognition (NER), where models need to label specific words rather than generate text. Adapting these models for such specific tasks often requires significant modifications to prevent them from producing outputs of arbitrary length.
4. **Dependency on Prompting:** The effectiveness of these models can heavily depend on the quality of the prompts they are given. Prompt engineering has become critical, as skilful prompting can significantly enhance model performance. However, this reliance on precise prompts can complicate the replication of experimental results, particularly in settings where input tokens must be rigorously defined.
5. **Ethical Concerns:** The capacity of models to generate harmful content, including toxic, hallucinatory, or racially insensitive remarks, remains a major concern. Instances like Galactica [TKC⁺22], a model developed by META for scientific QA that inadvertently created fictitious scientific names, underscore the potential dangers. This raises ethical questions about accountability, as posed by Yan

Table 5.15: DistilBERT Model Performance

Model Casing	Batch Size (bsz)	Loss (1 epoch)	Accuracy (%)
Uncased	16	0.5682	74.61
Cased	16	0.565	74.75
Uncased	96	0.5894	74.67
Cased	96	0.5712	74.54
Uncased	112	0.5646	74.91
Cased	112	0.5681	74.68

LeCun: *"Who bears responsibility when a model generates inappropriate content—the developers, the platform hosting the model, or the end-users ? "*

These challenges underscore the need for ongoing research, thoughtful model training, and careful deployment, especially as these tools become more integrated into various aspects of daily life and professional fields.

5.4.2 Preliminary Work (Finetuning Models, Pretraining Language Model)

We are analyzing data from 4,644 PDFs, selecting text that matches the paragraph coordinates from Grobid. After cleaning the data by removing duplicates, NaNs, and overlapping classes, we have 623,997 samples for training and 71,387 for validation.

We chose DistilBERT [SDCW19] for initial experiments due to its balance between performance and resource use, as documented in the "Doceng" paper. Our next steps involve finetuning DistilBERT (see table 5.15) to find the best batch size and decide whether to use cased or uncased text. These choices affect the model's learning process and its ability to generalize:

1. Batch Size: We must find a batch size that balances gradient stability and generalization ability. Larger batches might stabilize the gradient but could reduce the model's ability to generalize well.
2. Casing: Deciding on using cased or uncased text will impact how the model perceives different words, which could be important for recognizing proper nouns and other context-specific information.

These adjustments are critical for optimizing the model's performance on our specific dataset.

In our experiments with the DistilBERT model, adjusting the batch size and whether to use cased or uncased text did not result in any significant performance improvements.

5.4.3 Evaluation of Language Model on Large Dataset

Pretraining Language Model from Scratch

The effectiveness of domain-specific pretraining for language models has been highlighted in various studies, including research on CamemBERT [MMS⁺20], which showed that models trained on a comparatively smaller dataset of 4 GB of web-crawled data could outperform those trained on over 130 GB, particularly in domain-specific tasks. This finding underscores the importance of tailored pretraining and motivates our approach to pre-train our language model specifically for mathematical and scientific content.

The unique challenges of scientific terminology necessitate specialized training to enhance model understanding of this complex language. Drawing from successful strategies in previous studies, we choose to pre-train our model from scratch using a corpus of mathematical articles. This approach is intended to improve the model's performance in recognizing and processing scientific language and make it more efficient during finetuning phases, requiring fewer examples to adapt to specialized tasks.

Further, we explore the base vocabulary differences (see figure 5.48) across various language models to assess the specificity and relevance of our pre-trained model. Initial analyses show that our model referred to as `trained_tokenizer` in our documentation, shares a maximum of 33% vocabulary overlap with existing models, including SciBERT, which is trained on a scientific corpus. This relatively low overlap indicates that our model is developing a unique vocabulary set more closely aligned with the specific linguistic features of mathematical texts.

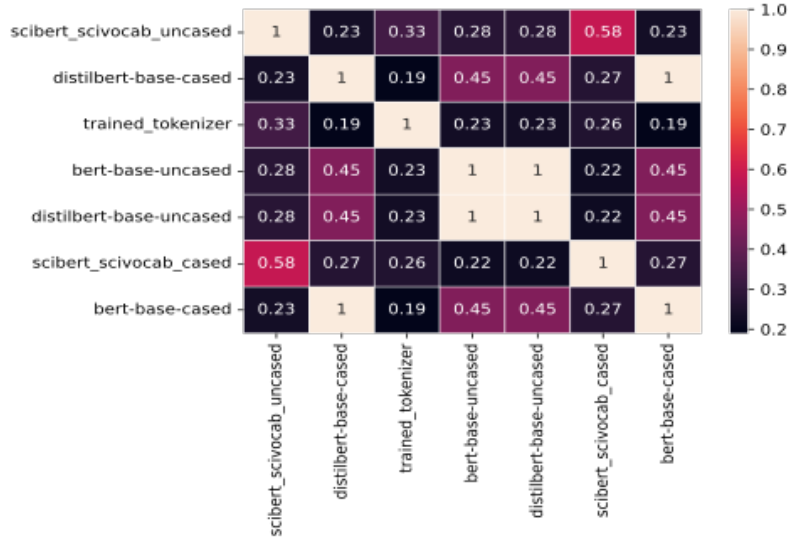


Figure 5.48: Vocabulary overlaps with many other existing models

By focusing on these aspects, we aim not just to enhance the performance of our model on scientific tasks but also to foster a more efficient learning process that can effectively leverage domain-specific knowledge without extensive retraining.

Our study undertook an extensive pretraining phase using 11 GB of textual data, encompassing 196,846 scientific articles. This pretraining was conducted over 11 epochs, utilizing the LAMB optimizer (You et al., 2020). We deployed this training across 4 NVIDIA A100 GPUs with a total batch size of 256, applying a distributed mirrored strategy. The initial learning rate was set at 2×10^{-5} , and the total duration for pretraining was 176 hours.

The pretrained-RoBERTa-like model on the Mask Language modelling (MLM) task is available here at https://huggingface.co/InriaValda/cc_math_roberta_ep10?text=Dijkstra%27s+algorithm+is+an+algorithm+for+finding+the+%3Cmask%3E+paths+between+nodes+in+a+weighted+graph

The scale and nature of the pretraining data are crucial for a language model’s performance and generalization speed. This is underscored by findings from the Chinchilla [HBM⁺22a] study (Hoffmann et al., 2022), which emphasizes the importance of data size through the "Chinchilla scaling laws". For instance, Chinchilla (70B)[HBM⁺22a] managed to outperform Gopher [RBC⁺21] (280B) by enhancing its training data volume by 4x, measured in terms of token size, showing that more data can trump larger model sizes, a principle further validated by the performance of LLAMA (7B) [TLI⁺23] over GPT-3 (175B).

In this research, we opted against using models with several billion parameters due to several factors:

1. Integration vs. Depth: Our primary focus is on theorem-proof identification, which requires integrating various Modalities rather than delving deep into advanced singular modalities. We start with base models to ensure scalability and relevance to our scientific domain.
2. Budgetary Considerations: The substantial computational costs of training and evaluating large models, particularly when multiple modalities are involved, necessitate a more pragmatic approach. We chose base models for their feasibility and flexibility within our modular framework.

We assessed the effectiveness of our pretraining through two configurations—a BERT-like [DCLT19] model and a RoBERTa-like [LOG⁺19] model—as detailed in table 5.16. To quantitatively evaluate the quality of the pretraining, we reported the perplexity of the Masked Language Modelling (MLM) task, see table 5.16 drawing a parallel to the method used in the RoBERTa study (Liu et al., 2019). This approach aligns with established practices and provides a clear metric for comparing the effectiveness of our pretraining regimen.

We show the Evolution of the MLM loss in Figure 5.49. For qualitative analysis, we intentionally picked up samples that required specific vocabulary understanding for the MLM task, see table 5.17.

Language model	Batch size	Steps	Learning rate	Perplexity (epoch #10)	Time per epoch (h)
BERT-like (110M)	256	47 773	2×10^{-5}	3.034	11
RoBERTa-like (124M)	256	47 773	2×10^{-5}	2.857	16

Table 5.16: Pretraining configurations (on large multimodal dataset)

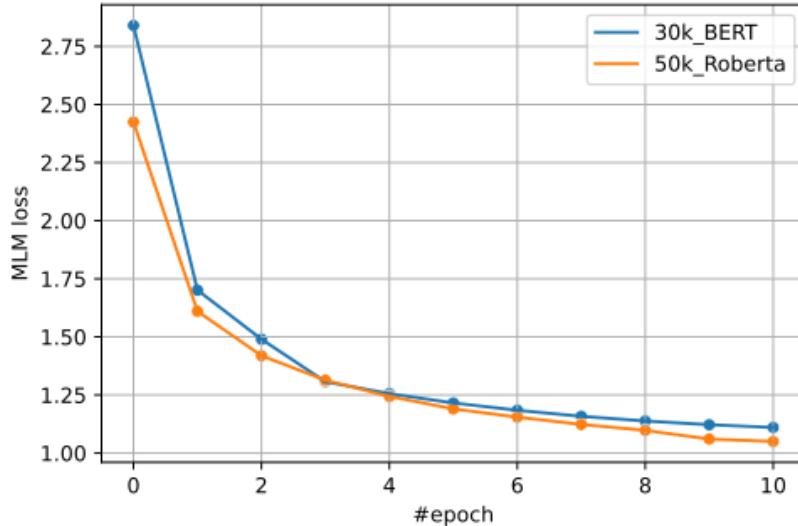


Figure 5.49: MLM loss for two pre-trained models, as a function of the pretraining epoch

Finetuning the Language Model

In our study, we proceed with the finetuning phase on a sizeable multimodal dataset using SciBERT, BERT, and our own pre-trained BERT model. All models are uncased, and the task is a 4-class classification. We employ the LAMB optimizer, setting the learning rate at 2×10^{-5} with a batch size of 16 per GPU across four NVIDIA A100 GPUs with 1k pdfs per batch, see figure 5.50.

The finetuning begins with models that have been pretrained for just one epoch. This initial low epoch pretraining approach allows us to explore the effect of minimal initial learning, focussing on subsequent finetuning performance. The method tests the hypothesis that even a slight pretraining can provide a beneficial starting point for more specialized training on targeted tasks like the classification challenge we face. By initializing weights from minimal pretraining, we aim to capture the early gains of language understanding while avoiding overfitting that may come from more extensive pretraining.

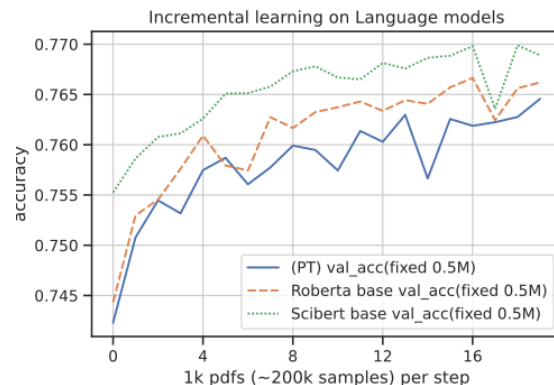


Figure 5.50: Accuracy of language models on finetuning task concerning number of batches

Masked sentence	Pretrained model	BERT-like	BERT model
This concludes the [MASK].	proof lemma case thesis	claim theorem	game story film play episode novel
We show this by [MASK].	induction lemma contradiction	. definition a	ourselves accident name themselves ear hand
By [MASK] 's inequality.	jensen minkowski	holder cauchy	young fourier brown russell fisher newton
The [MASK] is definite positive.	inequality function	case slimit sum	result sign value answer form
In particular any field is a [MASK].	. 1 group f field		field theory domain variety category
To determine the shortest distance in a graph, one can use [MASK] 's algorithm.	dijkstra hamilton	grover tarjan newton	shannon newton taylor wilson moore
An illustration of the superiority of quantum computer is provided by [MASK] 's algorithm.	grover kitaev	shor dijkstra yao	turing newton shannon maxwell einstein
One of the ways of avoiding [MASK] is using cross validation, that helps in estimating the error over test set, and in deciding what parameters work best for your model.	overfitting misspecification	error noise	errors error this bias uncertainty

Table 5.17: Inference of BERT-like pre-trained model on selected MLM tasks

Table 5.18: Performance comparison of various text based models

Model	#Batches	Inf. time (ms/step)	Accuracy (%)	Mean F ₁ (%)	#Params
Dummy	—	—	59.41	24.85	—
RoBERTa [LOG ⁺ 19] base	20	23	76.45	71.84	124M
Pretrained	20	23	76.46	72.39	124M
SciBERT [BLC19] base	20	23	76.97	72.87	110M

Table 5.19: Samples to target Accuracy for text models

Model	Data size	Samples to 65%	Samples to 70%
RoBERTa base	160 GB	41 472	186 496
Pretrained	11 GB, 197k papers	39 552	141 632
SciBERT base	1.14M papers	36 928	91 200

Table 5.18 illustrates that the SciBERT [BLC19] model outperforms and generalizes better than the pre-trained RoBERTa-like model in our classification task. Recognizing that SciBERT was trained explicitly on a substantial corpus of scientific papers is crucial, which likely contributes to its effectiveness in our context.

This comparison highlights a critical point about the relevance of domain-specific training. Even though the SciBERT model had access to a larger volume of training data, a model like ours, trained on a more focused scientific corpus, can achieve comparable performance with considerably less data, see table 5.19. This underscores the significance of tailored training data in developing efficient and highly effective models for specific applications.

The ability of domain-specific models to perform on par with more generally trained but larger models reflects the importance of quality and relevance in training data over sheer quantity. This insight is essential for optimizing model training strategies, especially when resources are limited or when specificity is required.

We can also deploy these trained models for classification tasks with minimal lines of code, see code in figure 5.51. The trained model supports transfer learning for similar or related tasks, as demonstrated in Shufan’s Named Entity Recognition (NER) task, see subsection 3.4.10. Models trained in this section are smaller and more lightweight compared to billion parameter LLMs, enabling their integration into the larger multimodal model. This approach makes the larger model trainable without extreme memory constraints during backpropagation and reduces inference time when deploying the theoremkb project on the web.

```
#for NLP based model
import numpy as np
import tensorflow as tf

from transformers import AutoTokenizer, AutoModel, utils
from transformers import TFAutoModelForSequenceClassification

#choose a path
load_path="InriaValda/roberta_from_scratch_ft"

#tokenizers
tokenizer = AutoTokenizer.from_pretrained(load_path)
loaded_model = TFAutoModelForSequenceClassification.from_pretrained(load_path, output_atten

sample1=""Proof. A feasible solution to this linear
program will define (setting  $p_i = e^{x_i}$ )
a sequence  $p = (p_1, \dots, p_n) \in (0, 1]^n$  such that""

input_text_tokenized = tokenizer.encode(sample,
                                       truncation=True,
                                       padding=True,
                                       return_tensors="tf")

print(input_text_tokenized)

prediction = loaded_model(input_text_tokenized)

prediction_logits = prediction[0]
prediction_probs = tf.nn.softmax(prediction_logits,axis=1).numpy()

np.set_printoptions(suppress=True)
print(f'The prediction probs are: {prediction_probs}')
print("rounded label(argmax) :{}".format(np.argmax(prediction_probs)))
```

Figure 5.51: A code example showcasing how to use the trained model (with a few lines of code)

Multimodal Approaches

Contenu du chapitre

6.1	Introduction	101
6.2	Preliminary Results – Multimodal	101
6.2.1	Manual labelling to filter outliers	102
6.2.2	Stacking Classifier – Multimodal	103
6.3	Related Work	104
6.3.1	Multimodal Deep Learning (using Cross-Modality Losses)	105
6.3.2	Multimodal Deep Learning (using Feature-Level Fusion)	106
6.4	Evaluation of Multimodal Model on Large Dataet	110

6.1 Introduction

In the previous chapter of this thesis, see chapter 5, we developed specialized backbone models for individual modalities—font, vision, and language. Each model is designed to be influential within its scope, focusing intensively on the specific features pertinent to its modality. However, to comprehensively understand the data, it is crucial to integrate these unimodal approaches into a cohesive multimodal framework. This integration aims to explore how features from different modalities relate to each other, the possibility of constructing a joint multimodal representation, aiming to surpass the performance of any single modality.

We will benchmark the multimodal model against the best-performing unimodal baselines to address these questions. This comparison will help us determine the added value of combining modalities over using any single one independently. Furthermore, we will explore whether the models developed for individual modalities can be effectively reused and integrated into the multimodal framework. By leveraging the strengths of each unimodal model, we aim to create a more robust and accurate system that capitalizes on the diverse aspects of the data.

This approach not only seeks to validate the efficacy of multimodal integration but also to establish a methodological framework for future research where complex datasets require nuanced interpretations that no single modality can provide on its own.

6.2 Preliminary Results – Multimodal

Building on the preliminary search described in chapter 5, we now proceed to apply the selected unimodal models to a dataset consisting of 622,618 training samples and 77,237 validation samples. This dataset, similar to the one processed in the vision section (see section 5.3.1) is fully equipped with all necessary features: rendered bitmap images, available font information, and text content from paragraphs, with all duplicates removed.

For our analysis, we choose one base model from each modality explored in the preliminary search:

1. Font Modality (taken from table 5.6): A 128-cell LSTM[HS97] model specifically tailored for font recognition, handling hundreds of unique font tokens with a relatively modest 98,000 parameters.
2. Text Modality (taken from table 5.15): The DistilBERT [SDCW19] uncased model features 66 million parameters and is well-suited for processing textual data.

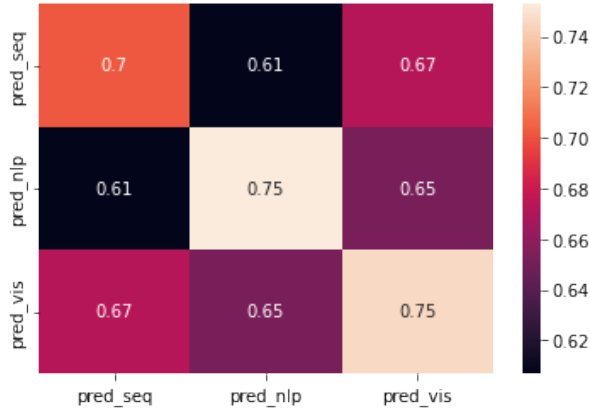


Figure 6.1: Agreement among class labels within the classifiers trained on different modalities

3. Vision Modality (taken from table 5.13): EfficientNetB0[TL19], with 4 million parameters, was selected for its efficiency and effectiveness in handling image data.

Each of these models was initially configured for a 3-class setting. To adapt them to the comprehensive dataset, we re-run these models to generate confidence intervals for the entire training dataset across all three modalities. This new set of confidence scores allows us to construct an enriched dataset comprising nine features—three confidence intervals from each of the three modalities.

This enriched dataset is the foundation for the training and validation phases. Integrating confidence intervals from diverse modalities aims to leverage each model’s strengths, potentially enhancing our multimodal analysis’s accuracy and robustness. This approach tests the models’ interoperability and explores how multimodal data can be synergistically used to improve prediction outcomes.

6.2.1 Manual labelling to filter outliers

Our analysis shows that the different modalities—font, vision, and language—tend to agree in most cases (see figure 6.1). However, there are instances where the modalities exhibit strong disagreements or discrepancies. Such disagreements could stem from outliers in the data extraction or labelling processes, suggesting anomalies that might not be adequately captured by automated systems alone.

These inconsistencies highlight the potential need for manual validation to ensure the accuracy and reliability of our multimodal analysis. Manual checks could help identify and correct errors in the automated extraction or classification processes, thereby improving the overall robustness of our model. This step is crucial for maintaining high data quality, especially when automated systems fail to interpret complex or ambiguous information accurately.

We employ an entropy-based approach to address and measure disagreements between different modalities in our multimodal dataset. This method calculates the entropy across all nine confidence interval features derived from the modalities. The concept is similar to the technique in decision trees, where entropy or gini impurity is utilized to determine the most informative features that best split the data.

By calculating the entropy for each feature, we can rank them in descending order from the highest to the lowest level of disagreement among the modalities. This ranking helps us to identify the most inconsistencies or intermodality disagreements, indicating potential issues in the data extraction or labelling processes that may not be apparent at first glance.

To further investigate these discrepancies, we select the top 1,000 PDFs that exhibit the highest modality disagreements. For each document, we identify the first occurrence of such disagreement, assuming that similar errors might recur within the same document. This approach allows for a focused review of the most problematic cases.

We utilize a data labelling tool such as Label Studio [Hum24] to facilitate the manual validation. This tool supports collaborative annotations, significantly speeding up the process of visually inspecting bitmap images and assigning the correct labels manually, see a sample template of Label Studio tool in figure 6.2. This manual intervention is crucial for verifying and correcting the labels where automated systems might have failed, ensuring the accuracy and integrity of our dataset. This step not only helps rectify immediate



Figure 6.2: An example of fast labelling using label studio on a dummy classification task

issues but also contributes to refining the model’s performance by providing cleaner, more reliable data for training.

We developed a systematic approach to identifying and correcting mislabeled data points to refine our multimodal model’s accuracy and improve our dataset’s quality. The process involves a comparison between the labels predicted by our algorithms and those manually assigned by annotators.

Upon identifying discrepancies, we examine the corresponding data XML and labelling files for each mislabeled instance to pinpoint potential causes of errors. This analysis allows us to enhance our \LaTeX labelling script, ensuring it assigns more accurate labels based on our findings.

For example, through this review process, we discovered several keywords that were not being correctly captured by the \LaTeX extraction script. One significant error involved misclassifying the label *"theorem"* as *"proof"* when theorems were discussed within proof contexts. These insights led to targeted corrections in our \LaTeX extraction script.

Applying these refined scripts to a larger corpus of 196,000 PDFs, which includes our multimodal validation dataset of 0.5 million samples resulted in more accurate data labelling. This enhanced dataset supports more reliable training and validation of our models, ultimately improving their performance and the validity of our research findings. This iterative process of testing, error analysis, and script adjustment is crucial for maintaining the integrity and accuracy of our machine-learning projects.

6.2.2 Stacking Classifier – Multimodal

Building meta-classifiers on top of the nine confidence interval features from the unimodal classifiers is a strategic approach to enhance prediction accuracy. This method, known as stacking, leverages the strengths of multiple base classifiers by using their predictions as input for a secondary classifier. This meta classifier can effectively integrate diverse signals from the underlying models to produce a more accurate final prediction.

The meta classifier could be a relatively straightforward machine learning model. Models such as logistic regression, decision trees, or even another neural network are common choices for this layer, depending on the complexity of the task and the level of integration needed. The essential advantage of this approach is that it can capture interactions between the predictions of the different unimodal classifiers that may not be apparent from their outputs alone.

This technique is beneficial in multimodal contexts where each model may capture different aspects of the data. By combining their predictions, the meta classifier can potentially correct errors made by individual unimodal classifiers and capitalize on their combined predictive power where they perform best. The process involves training the meta-classifier on a labelled dataset with the features as confidence scores of each unimodal classifier.

Implementing a stacking classifier enhances the system’s overall predictive performance and robustness, making it a valuable addition to multimodal machine learning pipelines.

Our research found that most simple machine learning models failed to generalize effectively (see figure 6.1) when used to build on the predictions from unimodal classifiers. Their performance generally fell short compared to the best-performing unimodal baseline. However, models utilizing gradient-boosted decision

Table 6.1: Performance comparison of meta classifiers trained on top of confidence intervals

Model	Better than Baseline?	Accuracy (%)	F ₁ Score	Time taken (s)
AdaBoostClassifier	✓	76.19	0.7546	36.81
BaggingClassifier	×	74.45	0.7337	77.54
BernoulliNB	×	72.94	0.7366	0.21
CalibratedClassifierCV	✓	76.30	0.7532	1146.14
DecisionTreeClassifier	×	68.97	0.6887	12.39
DummyClassifier	×	65.42	0.5174	0.12
ExtraTreeClassifier	×	68.74	0.6859	0.60
ExtraTreesClassifier	✓	75.79	0.7502	51.74
GaussianNB	×	74.48	0.7504	0.21
KNeighborsClassifier	×	74.16	0.7346	3.69
LSTM	-	64.40	-	-
EfficientNetB0	-	75.00	-	-
DistilBERT-uncased	-	74.62	-	-

trees—particularly those using a stacked approach—demonstrated better generalization capabilities, improving performance over any single unimodal baseline.

This observation has prompted us to shift our focus towards exploring deep learning-based approaches. Instead of relying solely on class interval predictions from previous models, we aim to work directly with the raw features extracted by deep learning models. This strategy is based on the hypothesis that deep learning architectures, particularly those capable of handling complex feature interactions and hierarchies, may offer more nuanced and powerful ways to integrate information across different modalities.

The intent is to utilize the inherent capacity of deep learning models to learn rich, hierarchical representations of data, which could potentially uncover more profound insights and patterns that simpler models might miss. This approach aligns with current trends in machine learning research, where deep learning continues to push the boundaries of what is possible regarding model sophistication and performance across various types of data and tasks.

6.3 Related Work

In this section, we organize the discussion into two main parts to effectively contextualize our research within the broader field of document AI, specifically focusing on multimodal learning:

- **Pretraining Transformer Models for Joint Feature Embedding:** This section reviews models designed to pre-train transformers to learn joint feature embeddings from various modalities, such as language and vision, specifically focusing on document AI tasks. We discuss how these models are typically applied to documents like bills and receipts, which are often assumed to contain all necessary information on the first page without spanning multiple pages or dealing with page breaks. However, scientific papers, particularly those containing lengthy proofs, frequently extend over several pages and include page breaks, presenting unique challenges that these models may not address. We will explore the limitations of current approaches and how they handle complex documents in a detailed sub-section, examining issues such as cross-modality losses and the need for more robust models to navigate the structured formats of scientific literature.
- **Multimodal Models Focusing on Feature-Level Fusion:** The second part of the related work section delves into multimodal models that emphasize feature-level fusion. These models integrate features extracted from standard deep learning backbones and apply additional layers on top of these backbones to enhance model performance. By synthesizing information from diverse sources like text, images, and possibly metadata, these models aim to create a more comprehensive understanding of the content. We discuss various strategies for feature integration, such as concatenation, attention mechanisms, or more complex fusion techniques, which are crucial for effectively combining modalities in tasks that require nuanced understanding, such as document analysis and comprehension.

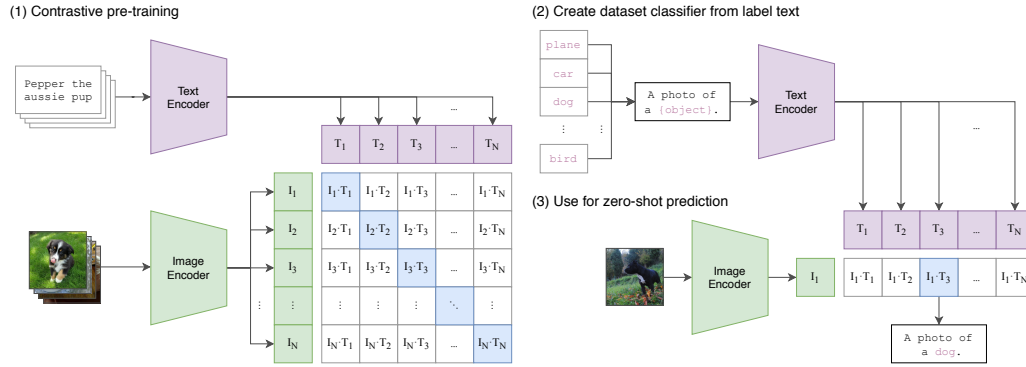


Figure 6.3: CLIP-based [RKH⁺21] architecture (can be applied on any image/text encoder)

Both sections aim to provide a comprehensive overview of the state of the art in multimodal learning within document AI, highlighting the advancements and the areas needing further research. This structured approach helps delineate the scope of current technologies and sets the stage for introducing our contributions to addressing the identified gaps in the literature.

6.3.1 Multimodal Deep Learning (using Cross-Modality Losses)

Recent advancements in artificial intelligence have introduced several transformative multimodal transformer architectures designed to integrate and process different data modalities within a single framework. Among these innovations, the LayoutLM ([XLC⁺20], [XXL⁺21], [HLC⁺22]) series (including versions 2 and 3) stands out for its unique approach of incorporating 2D positional embeddings and utilizing a masked visual-language model loss alongside multi-label document classification loss. This methodology enables the effective handling of complex interactions between visual and textual data early in the model’s input stage.

However, an alternative and increasingly popular approach is the late fusion technique, exemplified by CLIP [RKH⁺21] (Contrastive Language–Image Pretraining). In CLIP, text and visual features are independently extracted and then projected into a shared latent space where they are aligned using contrastive loss. This method facilitates zero-shot learning capabilities powered by supervision derived from language models. A notable advantage of CLIP [RKH⁺21] is its flexibility in dynamically updating or replacing the backbone architecture, accommodating rapid advancements in model design and training processes.

In our research, we have chosen to implement a late fusion strategy akin to CLIP rather than an early fusion model like LayoutLM for several key reasons:

- 1. Modular Backbone Integration:** Our approach prioritizes flexibility, allowing for the seamless interchange of various backbone architectures. This enhances the model’s performance and scalability and avoids the limitations of the fixed dimensional constraints typical of early fusion models.
- 2. Reevaluating Cross-Modality Capture:** While early fusion models like LayoutLM are praised for enhancing the understanding of cross-modality relationships through pretraining on specialized losses, this assumption requires more rigorous examination. We challenge this notion by directly comparing different fusion methods, focusing on using raw features combined with a standard cross-entropy classification loss. This approach ensures that all models are evaluated under consistent conditions, maintaining the same backbone architectures used in unimodal setups.

By adopting these strategies, our research aims to provide a clearer, more comprehensive understanding of how different fusion techniques affect the processing and performance of multimodal data integration, thereby contributing valuable insights into developing more effective and adaptable AI systems.

Details on LayoutLM comparisons to baselines:

In the LayoutLM paper [XLC⁺20], Table 5 showcases comparisons with various baselines for document classification tasks, many unimodal, alongside a single multimodal approach adapted from [DPR19]. Successive iterations, LayoutLMv2 and LayoutLMv3, primarily benchmark against the initial version or this same multimodal baseline, as evidenced in Table 3 of [XXL⁺21] and Table 1 of [HLC⁺22]. This comparison pattern is echoed in other document classification frameworks like LiLT [WJD22], where the multimodal

baseline incorporates significantly less powerful backbones. The baseline for multimodal comparison utilizes an XGBoost classifier [CG16], processing class scores (not features) from essential backbones (VGG-16 [SZ15] for visual and BOW for text), a setup that superficially engages with multimodality compared to LayoutLM’s use of ResNet-101 and BERT. Despite employing less advanced backbones, the performance of this multimodal network (93.03, as seen in Table 5 of the LayoutLM paper or table 6.2) closely approaches that of LayoutLM (94.42), as evident in table 6.2. This observation raises critical questions about the source of LayoutLM’s performance gains: Are they due to its unique loss functions, or merely the result of employing more robust baseline architectures for comparison?

Model	Accuracy
BERT _{BASE}	89.81%
UniLMv2 _{BASE}	90.06%
BERT _{LARGE}	89.92%
UniLMv2 _{LARGE}	90.20%
LayoutLM _{BASE} (w/ image)	94.42%
LayoutLM _{LARGE} (w/ image)	94.43%
LayoutLMv2 _{BASE}	95.25%
LayoutLMv2 _{LARGE}	95.64%
VGG-16 (Afzal et al., 2017)	90.97%
Single model (Das et al., 2018)	91.11%
Ensemble (Das et al., 2018)	92.21%
InceptionResNetV2 (Szegedy et al., 2017)	92.63%
LadderNet (Sarkhel and Nandi, 2019)	92.77%
Single model (Dauphinee et al., 2019)	93.03%
Ensemble (Dauphinee et al., 2019)	93.07%

Table 6.2: Classification table directly taken from LayoutLMv2 paper

Related Work on DiT: DiT [LXL⁺22], a notable architecture for document classification leveraging a vanilla Vision Transformer (ViT) backbone, is showcased for its adaptability to classification tasks in Table 1 of its publication. This table, however, limits its comparison to DiT’s performance (92.11) against the ResNext model (90.65), which we consider a suboptimal baseline due to advancements in CNN models like EfficientNet and EfficientNetv2. These newer models enhance performance and optimize training and inference times. For context, Table 2 in the EfficientNet paper [TL19] and Table 7 in the EfficientNetv2 paper [TL21] offer direct comparisons with the ResNext and vanilla ViT backbones, respectively, demonstrating the superiority of EfficientNet variants. Notably, Table 1 of the DiT paper [LXL⁺22] omits metrics such as FLOPs or training time per epoch, details that are explicitly addressed in Table 7 of [TL21], underscoring EfficientNetv2’s advancements over ViT-based architectures.

6.3.2 Multimodal Deep Learning (using Feature-Level Fusion)

Multimodal Fusion Mechanism (Shared Latent Representation)

One of the straightforward fusion strategies is to generate raw features from several deep learning-based models that process different modalities and then project them into the same latent space. Projecting them in the same latent space allows them to learn features and often time intermodality relationships. One of the early works [KSZ14] in this direction was done by Ryan Kiros, which could do vector arithmetic on multimodal feature spaces by projecting them on the exact shared representation (using LSTM and VGG-19 as base model, see figure 6.4).

This was similar to how initially word2vec [MCCD13] aimed at representing word vectors. Their paper also proposed a decoder network built on joint multimodal embedding that could generate new novel descriptions from scratch. Presenting the words in the same shared representation allows us to look for the nearest neighbours in the same or different modality. Thus making vector arithmetic possible among different modalities (see figure 6.5). These results showcase that some cross-modality information can be captured using a simple fusion layer.

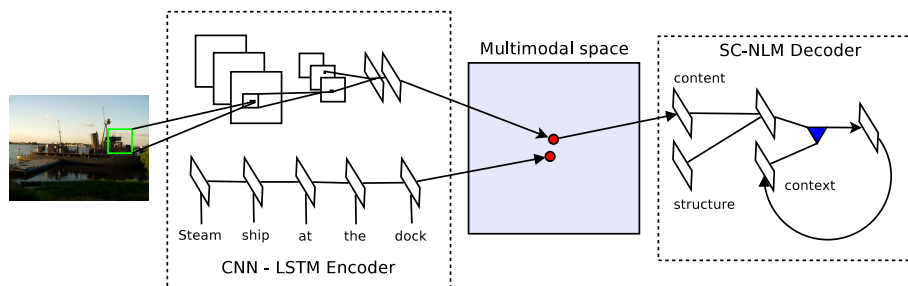


Figure 6.4: Multimodal [KSZ14] based fusion approach

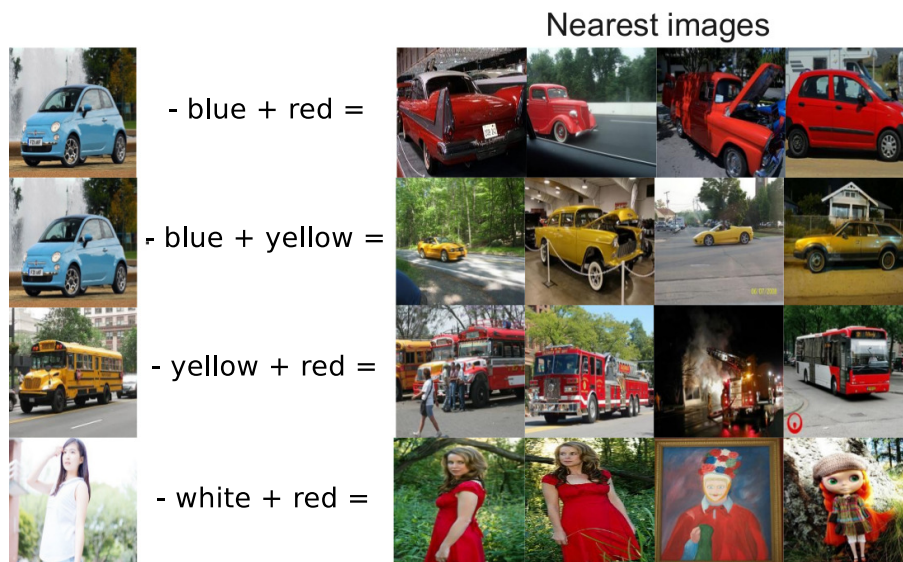


Figure 6.5: Vector arithmetic possible on multimodal feature spaces [KSZ14]

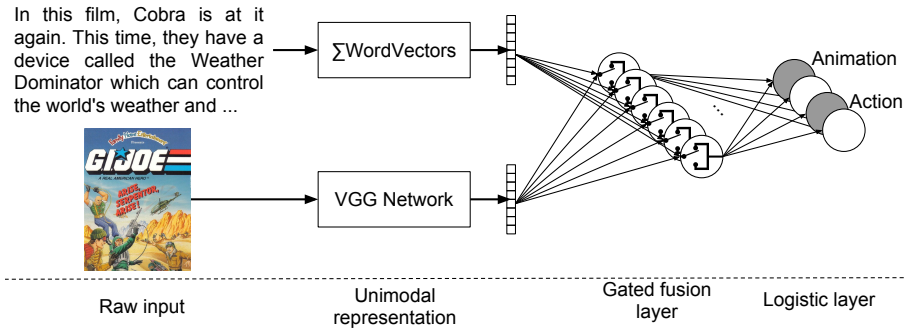


Figure 6.6: GMU [ASMyGG20] as a differentiable layer in the multimodal network

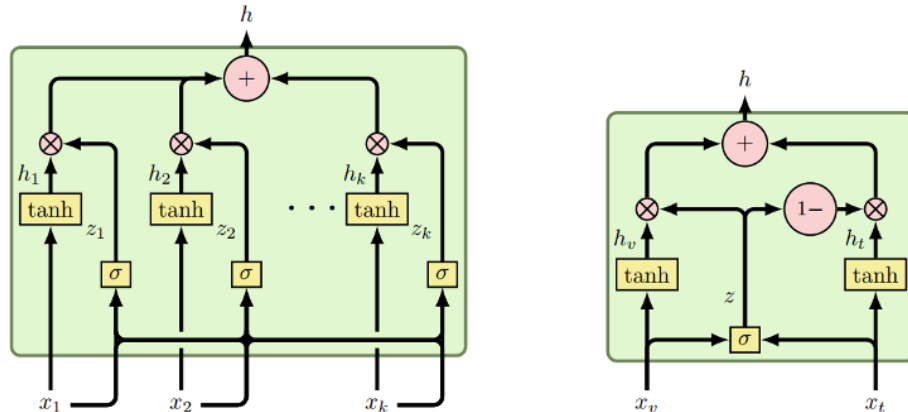


Figure 6.7: An illustration of GMU mechanism [ASMyGG20] proposed in the paper

Gated Multimodal Units (GMU) Mechanism

"Gated Multimodal Units for Information Fusion" [ASMyGG20] by Arevalo et al. Introduces a novel model architecture for multimodal learning using gated neural networks, known as the Gated Multimodal Unit (GMU). The GMU aims to create an intermediate representation of data from different modalities—such as text and images—by employing multiplicative gates to control the influence of each modality on the unit’s activation, see figure 6.7.

The model is designed to be an internal unit within a neural network architecture optimized for the specific objective function of the task. The GMU was evaluated in a multi-label movie genre classification task (see figure 6.6), utilizing movie plots (textual data) and movie posters (visual data). It showcased superior performance over single-modality methods and other feature-level fusion strategies, validating the effectiveness of the GMU’s selective gating mechanism in capturing complex interactions between different data types.

Furthermore, alongside the paper, the authors release the MM-IMDb dataset, which is purported to be the largest publicly available multimodal dataset for genre prediction at the time of its release, comprising movie plots, posters, and metadata (see figure 6.8). This dataset allows for the exploration of several tasks like document categorization and recommendation systems and could potentially foster further research in multimodal learning and data analysis.

In essence, this research demonstrates the viability of the GMU as a reusable, differentiable model component for multimodal learning, easily integrated and optimized within broader neural network architectures. It opens avenues for incorporating non-linearities to handle multimodal inputs’ intricacies better. It paves the way for more intricate architectures, such as deep GMU layers and integrating attention mechanisms.

EmbraceNet Mechanism

The paper "EmbraceNet: A robust deep learning architecture for multimodal classification" [CL19] by Jun-Ho Choi and Jong-Seok Lee from Yonsei University proposes a novel neural network architecture designed for tasks involving multimodal data classification. The key challenge addressed by EmbraceNet



Figure 6.8: Multimodal IMDb [ASM_yGG20] dataset proposed as the GMU task in the paper

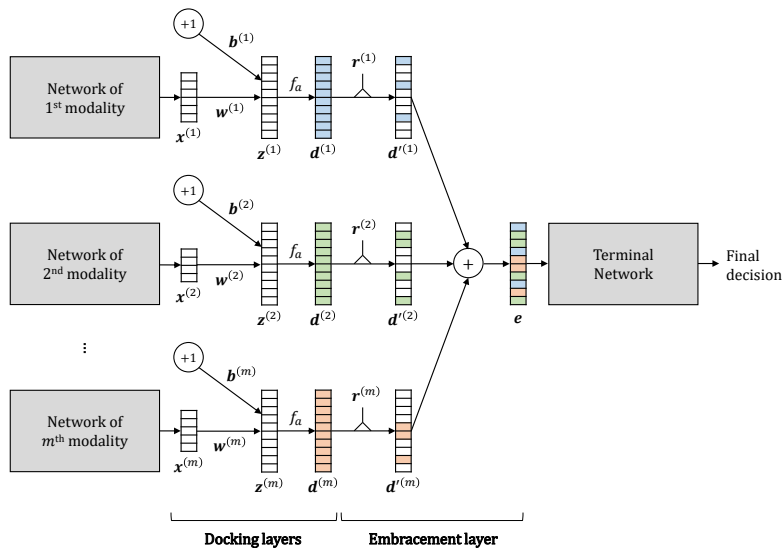


Figure 6.9: EmbraceNet architecture [CL19] (with docker layers)

is ensuring robustness against the partial or complete loss of data from one or more modalities, which is a common real-world issue.

EmbraceNet [CL19] introduces a unique structure comprising docking and embracement layers that facilitate multimodal data integration. The docking layers convert the data from each modality into a uniform representation suitable for fusion, while the embracement layer probabilistically combines these representations. This approach enables the model to maintain high performance even when some modalities are missing, a significant advantage over traditional deep-learning models that may struggle under such conditions (see figure 6.9).

The authors test EmbraceNet using two datasets: one for gas sensor arrays, which includes measurements from various chemical sources, and the OPPORTUNITY dataset [CSC+13], which consists of multi-labelled human activity data collected from numerous sensors. The performance of EmbraceNet is compared with state-of-the-art models across different scenarios, including when parts of data are missing either entirely or in blocks. The results indicate that EmbraceNet consistently outperforms other architectures, particularly in its robustness to data loss.

EmbraceNet's adaptability is showcased through its high compatibility with different types of learning models and their effectiveness in learning cross-modal correlations, key for tasks like activity recognition where such associations are vital. Furthermore, the embracement process within EmbraceNet, which selects information from each modality for combination contributes to its ability to handle missing data adeptly.

The paper concludes that EmbraceNet presents a viable solution for multimodal data fusion in real-world applications, demonstrating resilience to data loss and compatibility with various network structures. The proposed architecture provides a method for multimodal classification tasks.

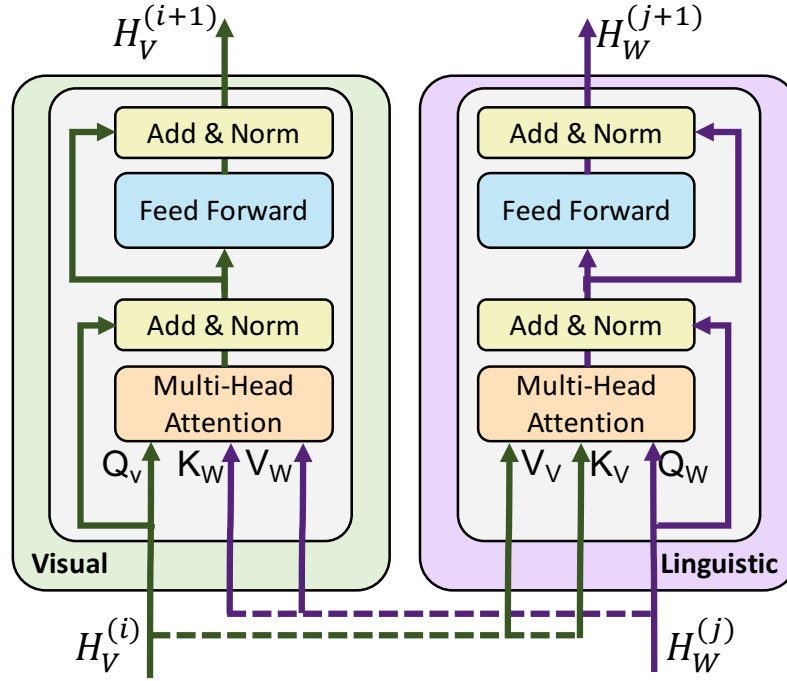


Figure 6.10: ViLBERT’s [LBPL19] encoder architecture

Cross-Modal Attention Mechanism

ViLBERT [LBPL19], which stands for Vision-and-Language BERT, is an extension of the famous BERT architecture for language understanding and is adapted to handle multimodal inputs of images and text.

The model consists of two parallel BERT-like networks that separately process visual and textual inputs (see figure 6.10). These networks are interconnected through co-attentional transformer layers, enabling the model to capture the associations between the two modalities. The co-attentional layers allow for variable depth processing for each modality and enable sparse interaction through co-attention, an innovation that significantly contributes to the model’s efficiency and efficacy.

ViLBERT is pre-trained on a large Conceptual Captions dataset containing around 3.3 million image-caption pairs. The pretraining involves two proxy tasks: masked multimodal learning and multimodal alignment prediction. In the first task, the model learns to predict masked words and image regions, while in the second, it predicts whether an image-caption pair matches.

After pretraining, ViLBERT is fine-tuned for various downstream vision and language tasks, including visual question answering, commonsense reasoning, referring expressions, and caption-based image retrieval. It sets new state-of-the-art benchmarks across these tasks, demonstrating the effectiveness of pretraining a visual grounding model and transferring it to various applications.

The paper concludes that ViLBERT represents a shift towards considering visual grounding as a foundational method that can be pretrained and transferred across multiple tasks rather than something learned separately within each task.

6.4 Evaluation of Multimodal Model on Large Dataet

We compare different modalities for late fusion: bilinear gated units [KGJM18], EmbraceNet [CL19], gated multimodal units (GMU) [ASMyGG20], and attention mechanisms such as those of ViLBERT [LBPL19], typically applied to dual modalities but extendable to multiple. Focusing on feature-level fusion, these methods ensure modularity and adaptability across architectures.

Our comparison to simple fusion methods is narrowed to concatenation, which is identified as the most effective among essential fusion strategies, allowing direct comparison with our unimodal baselines. These comparisons rely on cross-entropy loss for classification tasks, omitting additional losses like contrastive loss. Notably, the feature backbones are frozen during fusion, preventing weight updates and situating multimodal fusion as an augmentation to our unimodal framework.

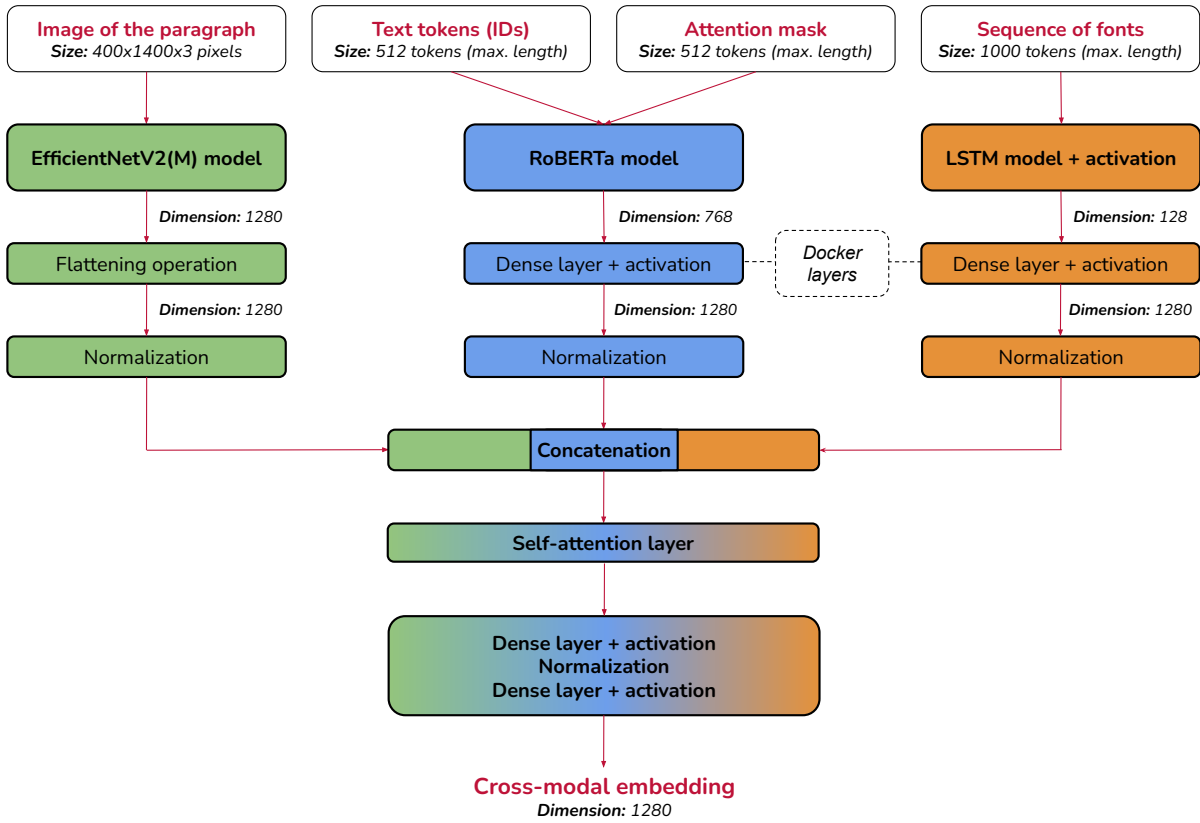


Figure 6.11: Cross-modal attention to capture relationships between the three modalities

For our baseline models used in multimodal analysis, we modify each to generate feature-rich outputs by removing the final softmax layer. Specifically, utilizing an LSTM architecture, the font-based model produces a 128-dimensional output. The vision-based model, an EfficientNetv2M, outputs 1280 features, while the text-based RoBERTa-like model generates 768 features taken from the [CLS] token of the last layer.

Building on the foundations set in previous sections (refer to unimodal chapter 5), we explore multimodal fusion techniques such as EmbraceNet [CL19], GMU (Gated Multimodal Unit) [ASMyGG20], and bilinear mechanisms [KGJM18] in a three-modality setting. The ViLBERT model inspires our approach to implement cross-modal attention (see figure 6.11). However, rather than using separate query vectors to attend to different sets of keys and values, we concatenate all the features from each modality into a single vector. This concatenated vector serves as the query and the value input, allowing each feature to attend to every other feature across modalities, including itself. This method enhances the model’s ability to capture and integrate cross-modal relationships, potentially leading to richer and more informative representations that better reflect the interdependencies among the different data types. This approach could significantly contribute to understanding complex interactions across text, visual, and font-related features, leading to more robust performance in tasks requiring deep multimodal insights.

In developing our multimodal models, we begin with the foundational approach of concatenation, which serves as a benchmark for all subsequent models. This primary method combines the feature outputs from each modality—font, vision, and text—into a single, unified representation.

To refine this approach and ensure that no single modality dominates due to a more significant feature dimension, we introduce “docker layers.” These layers project the outputs of each modality into a common dimensional space (1280 dimensions in this case), standardizing the feature size across modalities before concatenation. This normalization step helps balance the influence each modality has on the overall model’s performance.

We experiment with various fusion mechanisms after establishing a baseline with concatenated features. We replace simple concatenation with fusion techniques that vary in dimensionality sizes (768, 1280, 2304 dimensions), aiming to optimize the integration of modal information.

We then explore the bilinear fusion mechanism [KGJM18], which allows for an interactive combination

of features from different modalities, potentially capturing more complex interdependencies than linear methods like concatenation.

Following bilinear fusion, we investigate the EmbraceNet mechanism. This approach is designed to dynamically select and integrate features from multiple modalities, improving the model’s ability to adapt to the specific characteristics of the input data.

Finally, we experiment with integrating docker layers with techniques such as the Gated Multimodal Unit (GMU) or cross-modal attention mechanisms. The GMU method allows for the selective integration of modalities based on their relevance to the task. At the same time, cross-modal attention enables each feature to interact with every other feature across modalities, including itself.

Each step in this progression—from simple concatenation to more complex fusion and attention mechanisms—aims to incrementally improve the model’s ability to understand and process multimodal inputs, thereby enhancing its predictive accuracy and applicability to real-world tasks.

Our exploration of multimodal deep learning architectures has shown promising results, see table 6.3 with almost all multimodal configurations outperforming the best unimodal architecture (natural language processing model). The performance metrics for the various modalities were closely matched, with scores ranging from 78 to 78.5, see figure 6.3, indicating a high level of efficacy across the board.

Adding just a few million parameters to the base models resulted in reasonable gains, underscoring the efficiency of the multimodal approach without requiring a substantial increase in complexity. Among the techniques we implemented, docker layers proved generally effective across most approaches, helping to balance the contributions of different modalities by projecting them into a unified dimensional space.

The most impactful technique, however, was the cross-modal attention mechanism. This approach, allowed each feature to interact with features from other modalities (including itself) assigning them weights, similar to what attention mechanism does with word tokens in Transformer models. This success highlights the potential of attention mechanisms to significantly improve the integration and processing of multimodal data, leading to richer and more accurate model predictions.

These findings suggest that adding late fusion based enhancements to build multimodal deep-learning architectures can capture crossmodality and lead to substantial performance improvements, making them highly effective for complex tasks across diverse data types.

Table 6.3: Performance comparison of multimodal fusion techniques (with @dimensions and model architecture) on large multimodal dataset

Model Architecture	# Params (Total/Trainable)	Acc. (%)	Mean F ₁ (%)
Concatenated raw features (@2176)	179M/8K	77.90	74.34
Docker layers (@1280) + concat (@3840)	180M/1M	78.12	75.05
Docker layers (@1280) + fusion (@768)	183M/4M	78.50	75.38
docker layers(@1280) +fusion (@1280)	185M/6M	78.44	75.24
docker layers(@1280) +fusion (@2304)	189M/10M	78.42	75.13
bilinear mechanism (@1280)	182M/3M	77.99	74.78
docker layers (@1280) +bilinear gated mechanism(@1280)	185M/6M	78.30	75.20
docker layers(@1280) +GMU mechanism (@1280)	185M/6M	78.12	75.52
docker layers (@1280) +attention mechanism (@1280)	185M/6M	78.50	75.37
docker layers(@1280) +multihead attention (@1280, 8 heads)	244M/65M	78.34	75.25
EmbraceNet mechanism (@1280) (balanced prob +docker layers (@1280) incl)	182M/3M	77.73	74.92
EmbraceNet mechanism (@1280)(weighted prob +docker layers (@1280) incl)	182M/3M	77.74	74.53
docker layers(@1280) +fusion (@2304) +fusion (@768)	191M/12M	78.50	75.33
docker layer (@1280) +Cross-modal attention (@1280) +fusion (@768)	186M/7M	78.45	75.25
docker layer (@1280) +GMU mechanism (@1280)+fusion (@768)	186M/7M	78.49	75.27
<i>Unimodal Baselines:</i>			
RoBERTalike Pretrained	124M	76.46	72.39
EfficientNetV2M	53M	69.44	59.96
LSTM	2M	64.93	45.48

Sequential Approaches

Contenu du chapitre

7.1	Introduction	115
7.2	Related Work	116
7.2.1	Conditional Random Fields	116
7.2.2	BiLSTMs	117
7.2.3	Transformers	117
7.2.4	Transforming Attention mechanism to handle long Sequences	118
7.2.5	Hierarchical Attention Transformers	119
7.3	Evaluation of Sequential Model on Large Dataset	121

7.1 Introduction

Recognizing the importance of contextual information across different paragraphs is crucial to enhancing our document analysis model. This includes understanding the relationships and dependencies between paragraphs based on their content and spatial positioning, such as page number, relative position within the document, and coordinates (top-left and bottom-right). Additionally, determining whether sequential paragraphs are on the same page can provide valuable insights into the structure and flow of the document.

Human readers naturally grasp this context, but encoding it for machine processing requires sequential modelling techniques. We employ methods like conditional random fields (CRF) and sliding window transformers, adept at capturing sequential and relational information across text blocks to address this.

Building on the capabilities of our existing multimodal classifier (refer to chapter 6), we generate a multimodal embedding for each paragraph or text block in the document. These embeddings encapsulate information from various modalities and are input into sequential learning algorithms that aims to map surrounding paragraph context. This process allows the model to understand the continuity and context of the document more deeply, akin to human reading patterns. For a full architectural overview, refer figure 7.1.

By integrating these sophisticated models, we aim to create a more intuitive and accurate document analysis system that mimics human cognitive abilities in understanding complex textual structures. This

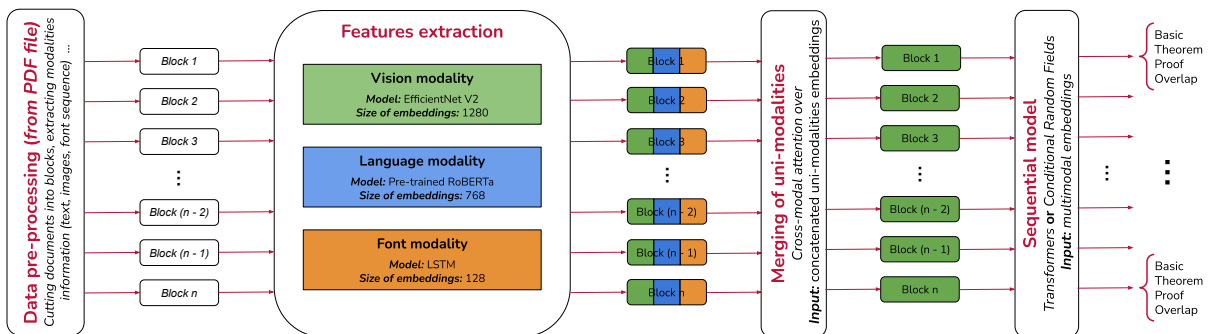


Figure 7.1: full data pipeline including the sequential learning model (at the end)

approach improves the model's accuracy and its ability to handle diverse and extensive datasets, providing a robust solution for automated document analysis.

7.2 Related Work

7.2.1 Conditional Random Fields

Conditional Random Fields (CRFs) [LMP01] is a statistical modelling method frequently utilized in pattern recognition and machine learning for modelling sequential dependencies in data. Unlike models that predict sequence labels independently, CRFs consider the context of label occurrences, enhancing their effectiveness in scenarios where relationships between sequential data points are pivotal.

CRFs are particularly valued in natural language processing (NLP) and computer vision. In NLP, they are employed for tasks like part-of-speech tagging and named entity recognition, where understanding the context of a word is crucial for assigning the correct tag. For example, the word "bark" could be tagged differently based on whether it appears in the context of a tree or a dog.

In computer vision, CRFs are instrumental in image segmentation tasks. Here, predicting a pixel's label depends on the labels of adjacent pixels, leveraging spatial dependencies to improve segmentation accuracy.

The core advantage of CRFs over models like Hidden Markov Models (HMMs) lies in their ability to directly model the conditional probability of a label sequence given input features. This approach allows CRFs to focus on the dependencies between observed features and labels without needing to model the overall distribution of the observed data. This direct modelling of conditional distributions avoids the inefficiencies associated with HMMs, which require modelling the joint distribution of labels and observed data.

By integrating context directly into the prediction process, CRFs provide a framework for tasks requiring a nuanced understanding of data relationships, making them a powerful tool in fields that rely heavily on accurate sequence prediction and classification.

Conditional Random Fields (CRFs) are extensively utilized in various natural language processing tasks, including keyphrase extraction framed as a Named Entity Recognition (NER) task. This modelling approach allows for each token within a text to be associated with a set of potential labels, offering a method for identifying relevant terms or phrases within documents.

One of the critical strengths of CRFs is their ability to handle data sequences of arbitrary lengths, which makes them highly adaptable for texts of varying sizes—from sentences to full-length papers. This flexibility is critical in research environments where document lengths can vary significantly.

The integration of CRFs with neural network architectures, particularly Bi-directional Long Short-Term Memory networks (Bi-LSTMs), has been a popular approach in recent research, see figure 7.2. This combination leverages the deep-learning capabilities of Bi-LSTMs to capture complex data patterns and the sequential data modelling strength of CRFs. For example:

- The study [PC19] titled "Exploring Word Embeddings in CRF-based Keyphrase Extraction from Research Papers" explores the efficacy of CRFs in keyphrase extraction, demonstrating how word embeddings can enhance the CRF's performance by providing rich, contextual representations of words.
- In [ACG19] "Bi-LSTM-CRF Sequence Labeling for Keyphrase Extraction from Scholarly Documents", the authors implement a Bi-LSTM layer topped with a CRF layer to refine the predictions made by the neural network, ensuring that the final keyphrase predictions are contextually coherent and accurate.
- "Theory Entity Extraction for Social and Behavioral Sciences Papers using Distant Supervision" [WSW22] also employs a Bi-LSTM-CRF architecture, highlighting its utility in extracting complex theoretical constructs from scholarly texts in specific academic domains.

These studies collectively underscore the versatility and effectiveness of CRF models, especially when combined with advanced Neural architectures like Bi-LSTMs process and extract valuable information from extensive and varied textual data. The ability to scale to different text lengths and the enhanced accuracy from deep learning models make CRF an indispensable tool in advanced text analysis tasks.

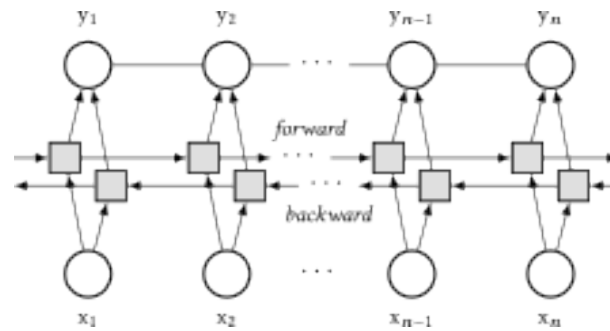


Figure 7.2: CRF layer on top of BiLSTM [ACG19]

7.2.2 BiLSTMs

In natural language processing tasks, especially those involving sequential data like text from scientific papers, the context surrounding each word is crucial for accurately predicting its label. Conditional Random Fields (CRFs) are traditionally used in such scenarios because they consider the dependencies of nearby labels, which greatly influence the tagging of each word. However, CRFs may not capture broader contextual dependencies that span multiple paragraphs, which is essential for a deeper understanding of the text.

Bi-directional Long Short-Term Memory (BiLSTM) networks offer a more dynamic solution by providing bidirectional context, which captures information from past and future states in the data sequence. This is particularly useful for understanding the context of a paragraph not only by its preceding text but also by what follows. The gated mechanism of BiLSTMs helps in controlling the flow of information, allowing the model to retain or forget information, which is crucial for handling long documents.

Despite the advantages, BiLSTMs come with their own set of challenges:

1. **Lack of Attention Mechanism:** Unlike models like Transformers that use attention mechanisms to focus on important parts of the input data, BiLSTMs lack this capability. They treat all parts of the input sequence as equally important unless explicitly modelled otherwise.
2. **Gradient Vanishing Problem:** BiLSTMs are susceptible to gradient vanishing, especially when dealing with long input sequences. In long documents, the influence of words or paragraphs appearing early in the sequence can diminish or disappear when the gradients are propagated back through the network during training. This can lead to poor performance on initial sections of a document, as the model fails to update weights effectively for early inputs.

To address these limitations, integrating BiLSTMs with mechanisms like attention or using newer architectures like Transformers, which inherently include attention mechanisms and might provide a more robust framework for handling complex and lengthy documents. These models capture the nuances of language more effectively by focusing on relevant parts of the data and mitigating issues related to long-distance dependencies in text.

7.2.3 Transformers

Transformer models were primarily developed to address the limitations of LSTM [HS97] models concerning long-term dependencies, enhancing performance through advanced attention mechanisms. Despite their success, transformers are not without flaws, particularly their high memory demands due to the quadratic scaling of the attention mechanism with respect to the number of tokens, n^2 . This scaling issue restricts the practical application of transformers in processing very long documents, as their maximum input length is typically capped (e.g., BERT's [DCLT19] maximum length of 512 tokens).

Innovative methods have been proposed to segment data into manageable chunks within the transformer's maximum context window to manage lengthy texts. The transformer model then processes these segments individually, with each segment generating its own set of embeddings. These embeddings can be further processed using a recurrent LSTM layer. This effectively connects the segmented data into a cohesive sequence to maintain coherence across segments and capture inter-segment relationships.

The approach is detailed in [PZV⁺19] which suggests two hierarchical structures:

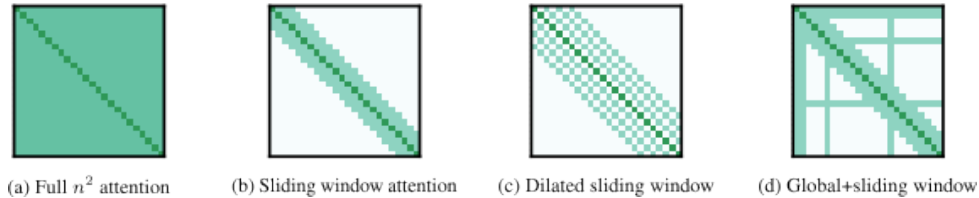


Figure 7.3: Longformer attention mechanism [BPC20]

1. Recurrence over BERT (RoBERT): This method applies a recurrent layer over the segmented outputs of a BERT model, enhancing the model’s ability to integrate information across multiple data chunks.
2. Transformer over BERT (ToBERT): This configuration uses an additional transformer layer to process the outputs from BERT across segments, aiming to capture more complex inter-segment relationships.

Such hierarchical structures are beneficial for processing long documents and crucial for computationally intensive tasks, where running a full transformer model on all data is impractical. For example, in image segmentation, hierarchical transformers like the Swin Transformer [LLC⁺21] have been used to manage computational load effectively by processing divided chunks of the input, thereby reducing the overall computation time while maintaining or enhancing performance.

This segmenting and hierarchical approach allows transformers to be applied more broadly and efficiently, even in traditionally challenging scenarios due to the data’s length or complexity.

7.2.4 Transforming Attention mechanism to handle long Sequences

Effectively handling long sequences is crucial for enhancing the scalability and efficiency of Transformer models, particularly when dealing with extensive texts that span multiple pages. While Transformer models better capture long-term dependencies than an LSTM, they remain restrictive in terms of broad-term context and resource consumption, with memory usage scaling as $O(n^2)$. Several papers have suggested a shift from Transformer models, such as the recently released state space model (SSM) Mamba [GD23] in May 2024. Mamba aims to address the time complexity issues inherent in Transformer models across various modalities, including language, audio, and genomics datasets. Mamba offers linear time complexity and outperforms Transformers twice its size, providing five times higher throughput. Due to time constraints in this thesis, we will not be experimenting with Mamba in our results. However, this motivates the upcoming section on why the standard Transformer architecture is not directly useful in our setting.

Having identified the primary task of long document classification and the limitations of the standard Transformer architecture, we will now explore how several popular Transformer architectures that address the challenges associated with long documents. Specifically, we will discuss how these models modify the attention mechanism to adapt to long sequences.

1. Longformer Architecture [BPC20]: The Longformer is designed to manage the computational challenges of long sequences. Standard transformers exhibit quadratic growth in computational cost with increasing sequence length due to their self-attention mechanism, which involves computing interactions between every pair of positions in the input sequence. To address this, the Longformer introduces a sliding window attention mechanism. This approach restricts each token’s attention to a fixed-size window surrounding it, significantly reducing computational requirements while capturing essential local context (see figure 7.3). Additionally, the Longformer employs global attention on selected tokens crucial for understanding the entire document, such as specific entity tokens or special tokens like [CLS] used in classification tasks. This hybrid attention strategy enables the Longformer to handle much longer texts efficiently. It is suitable for applications like document classification, question answering, and coreference resolution where extended context is beneficial.
2. Big Bird Architecture [ZGD⁺20]: Like the Longformer, the Big Bird model addresses the limitations of traditional transformers by incorporating a sparse attention mechanism, effectively reducing the computational complexity from quadratic to linear. Big Bird’s attention mechanism is a novel combination of:

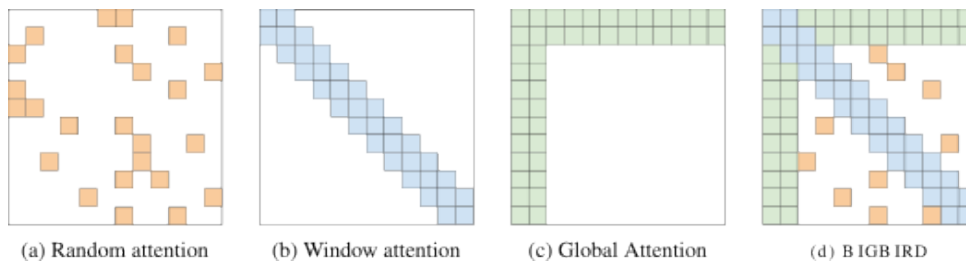


Figure 7.4: Bigbird Attention mechanism [ZGD⁺20]

- (a) Local Windowed Attention: Maintains awareness of local context by allowing each token to attend only to a fixed number of nearby tokens.
- (b) Global Attention: Enables certain key tokens to attend across the entire sequence, integrating broad contextual information crucial for understanding the document.
- (c) Random Attention: Introduces randomness in attention patterns across the sequence, enhancing the model’s robustness and generalization ability.

This configuration allows Big Bird to process much longer sequences — up to 4096 tokens — using the same amount of computational resources as traditional models handle only 512 tokens (see figure 7.4). This capacity is precious in tasks requiring extensive contextual understanding, such as document summarization and genomic sequence analysis, which were previously impractical without segmenting the text or heavily truncating it.

Both Longformer and Big Bird demonstrate that it is possible to adapt the transformer architecture to extend their applicability to longer texts without incurring prohibitive computational costs. By modifying the attention mechanisms, these models not only maintain the strengths of standard transformers but also expand their utility to a wider range of longer-sequence NLP tasks, providing more accurate and contextually aware outputs. These innovations represent significant advancements in the field of NLP, offering more flexible and efficient solutions for processing extensive sequences, broadening the scope of possible applications for transformer-based models.

7.2.5 Hierarchical Attention Transformers

Hierarchical Attention Transformers (HATs) [CDF⁺22] represent a significant evolution in natural language processing, especially suited for handling complex, long document classification tasks. These models integrate a hierarchical structure that effectively manages the challenges associated with extensive texts, a domain where traditional transformers often fall short due to the high computational demand of their self-attention mechanisms.

Similar to BigBird and Longformer, HATs deploy a novel architecture that transforms the attention mechanism itself (see figure 7.6). Instead of applying the attention mechanism to the full corpus, HATs focus local attention on small segments and build a global understanding of the document. This is achieved by having cross-segment interactions across different local segments (using their [CLS] tokens), denoted as the cross-segment encoder (CSE), as illustrated in Figure 7.5.

1. Efficiency and Performance: HATs are particularly noted for their computational efficiency. They are designed to consume less GPU memory and offer faster processing speeds than conventional models like the Longformer and Big Bird. For example, HATs have been shown to process texts up to 40-45% faster and use about 10-20% less GPU memory. This increase in efficiency does not compromise their performance; on the contrary, HATs often outperform other models in tasks involving long document classification such as Bigbird and Longformer. This makes them highly suitable for processing detailed and extensive documents typically found in legal and biomedical fields.
2. Hierarchical Attention Mechanism: The core feature of HATs is their hierarchical attention mechanism, which applies attention sequentially at different levels of the document structure. This method allows the model to effectively parse and interpret large texts by focusing on local contexts within smaller segments and then integrating these insights across the entire document (see figure 7.5). This tiered approach boosts processing efficiency and enhances the model’s ability

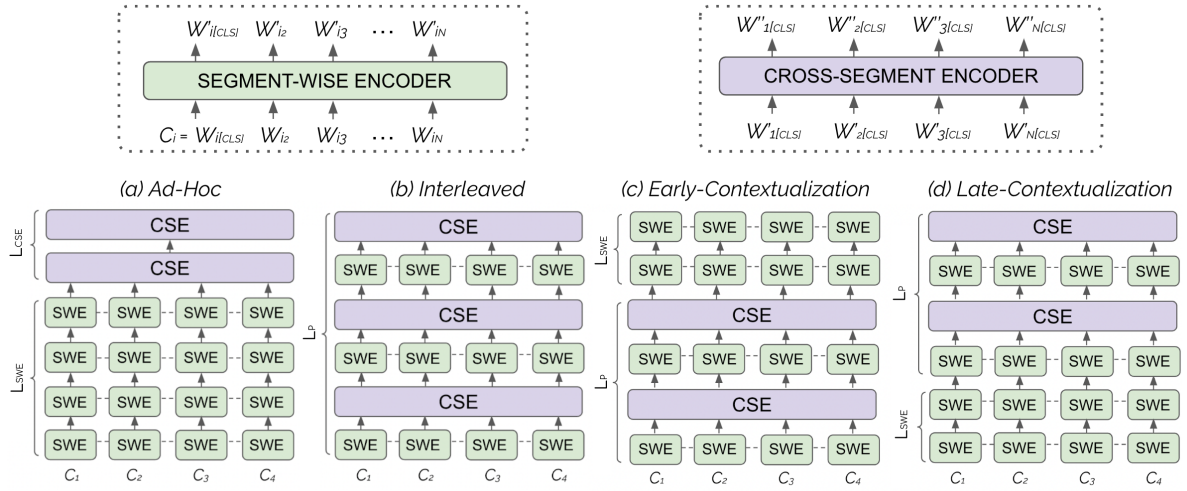


Figure 7.5: Types of Hierarchical Attention Transformer [CDF⁺ 22] models

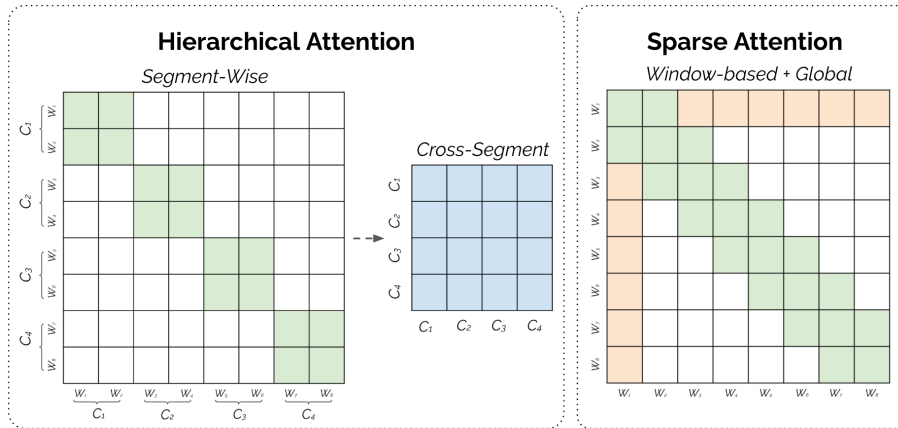


Figure 7.6: HATs Attention mechanism [CDF⁺ 22]

Table 7.1: Impact of large Max Length on Transformer Model (attention heads =16)

Max Len	Train loss	Accuracy (%)	Mean F ₁ (%)
1024	0.4984	80.37	77.06
512	0.4953	80.46	77.39
256	0.4565	80.65	78.64
128	0.4843	78.79	76.16

Table 7.2: Impact of Attention Heads (with maxlen = 256) based on results from Table 7.1

Heads	Train Loss	Accuracy (%)	Mean F ₁ (%)
8	0.5014	80.80	78.11
12	0.5007	80.83	78.06
16	0.4986	80.86	78.13
20	0.4992	80.84	78.15

to discern complex contextual relationships vital for understanding detailed documents. There are many proposed HAT architectures but we our study employs only the best performing HAT i.e. the interleaving style.

3. Architectural Advantages: The hierarchical design of HATs significantly reduces the computational burden associated with processing large volumes of text, which is a common challenge in traditional transformer models. By managing attention in a layered and segmented manner, HATs maintain high accuracy and nuanced text interpretation, crucial for domains requiring a deep understanding of extensive textual information.

Overall, Hierarchical Attention Transformers offer a promising solution for long document NLP tasks that involve analyzing and classifying long documents, providing a blend of speed, efficiency, and accuracy that is crucial for practical applications in specialized fields.

7.3 Evaluation of Sequential Model on Large Dataset

Our experimental framework initiates with a comparison against a baseline Conditional Random Field (CRF) model, known for its proficiency in modelling sequence data. Following this, we delve into the Sliding Window Transformer architecture (see figure 7.7), which utilizes multimodal embeddings derived from a crossmodal attention model. This architecture also incorporates six layout-level features, including page number, top-left and bottom-right coordinates of text blocks, and page continuity indicators to enhance its contextual understanding. The goal is to build add multimodal capabilities to a long document classification model while ensuring the approach is fully modular add offers incremental gains at every level across every modality.

To understand the development of the SW transformer architecture introduced in figure 7.7 we must need to understand what were the main problems of sclaiing a vanilla transformer architecture on top of the multimodal model. Here we show incremental changes that we go through to reach our SW transformer architecture.

1. Base Encoder Design: The journey begins with crafting a base encoder similar to BERT, focusing initially on tuning the ‘maxlen’ parameter (having 20 attention heads instead of 16), which defines the number of multimodal embeddings the encoder attends to. Recognizing that document classification often depends on local paragraph context, we found optimal performance with a ‘maxlen’ of 16, indicating that most paragraphs depend only on a few surrounding blocks for context. see tables 7.1, 7.2, 7.3.
2. Parameter Efficient: Our experimentation revealed better performance with shorter ‘maxlen’. This led to an adjustment in the encoder’s feedforward network, which was traditionally a significant contributor to parameter count in BERT models. We achieved comparable or improved performance

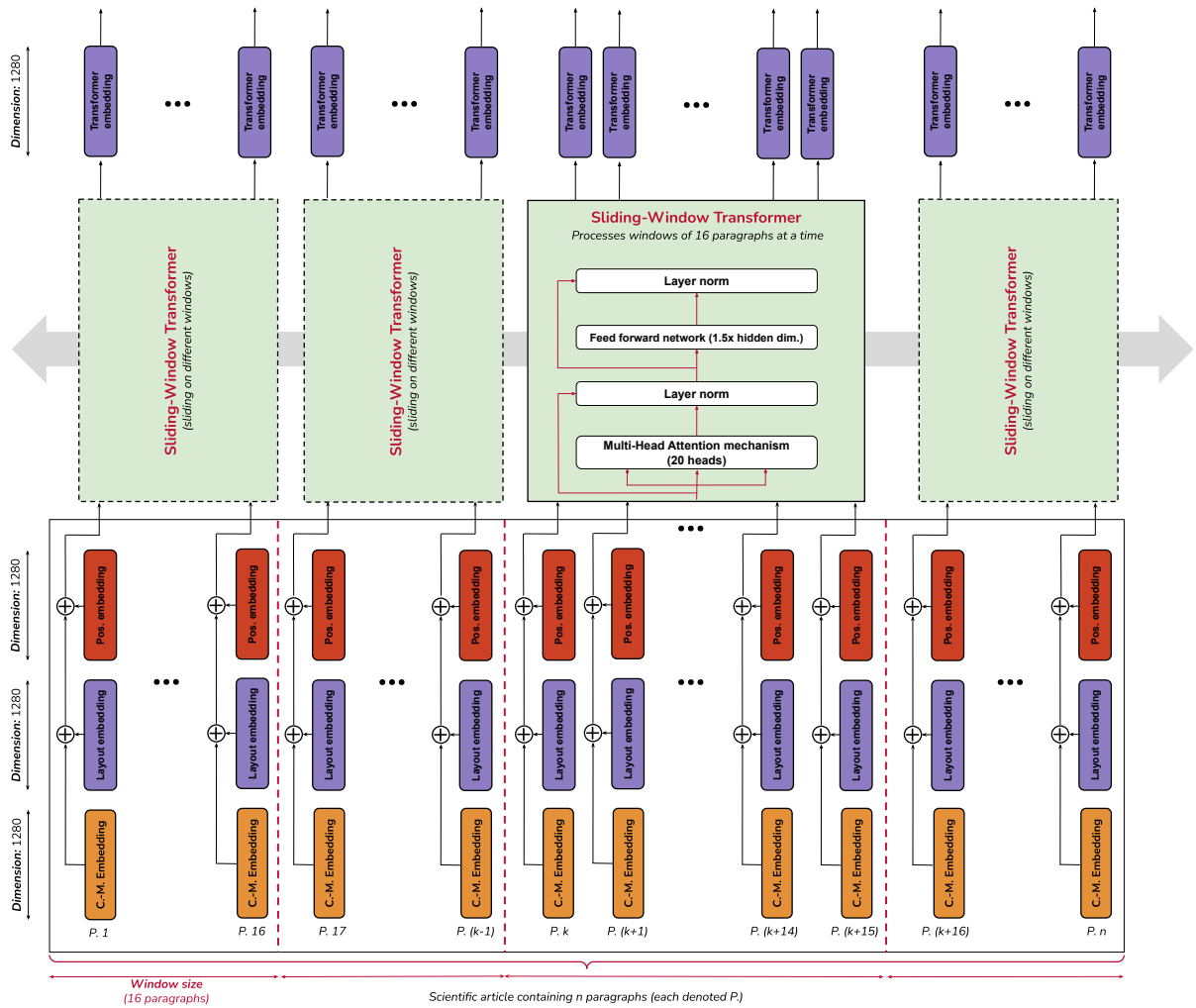


Figure 7.7: Sliding window Transformer

Table 7.3: Impact of small Max Length on Transformer Model (with heads=20) based on results from Table 7.2

Max Len	Train Loss	Accuracy (%)	Mean F ₁ (%)
128	0.4846	81.20	78.61
64	0.4609	82.03	79.48
32	0.4307	85.46	83.90
16	0.3890	85.17	85.01
8	0.3970	83.16	81.15

Table 7.4: Impact of different ff -dim multipliers on Transformer Model (with $maxlen=16$, $heads=20$) based on results from Table 7.3, 7.2

ff -dim	Train Loss	#Params	Accuracy (%)	Mean F_1 (%)
6 times	0.3905	26.90M	85.03	84.88
4 times	0.3890	20.34M	85.17	85.01
2 times	0.3888	13.79M	85.24	84.90
1 times	0.3906	10.51M	85.46	84.93
0.5 times	0.3967	8.87M	84.81	84.51
0.25 times	0.3960	8.05M	85.24	84.53

Table 7.5: Comparison of Bert-like and Efficient Transformer Models (with $maxlen=16$, $heads=20$, $encoders=1$)

Model	Train Loss	#Params	Accuracy (%)	Mean F_1 (%)
BERT like ff -dim= $4\times$	0.3890	20.34M	85.17	85.01
Efficient ff -dim= $1.5\times$	0.3864	12.15M	85.63	85.25

with substantially fewer parameters by reducing the scaling factor in the feedforward dimension (ff -dim) from the suggested $4x$ to a more modest $1.5x$. see tables 7.4, 7.6, 7.5.

3. Exploring Window Sizes: Adjusting the sliding window size revealed that a window of 16 with varying ‘ $maxlen$ ’ values efficiently handles longer documents by segmenting them into manageable parts without loss of context or performance, see table 7.7.
4. Sliding Window Mechanism: Incorporating a sliding window directly into the model allows it to handle longer sequences without subdividing them into smaller chunks, a common approach for managing ‘ $maxlen$ ’ constraints. This adaptation is particularly beneficial for documents where segment length varies, as it maintains continuity and context more effectively, see table 7.8.
5. Sampling and Epoch Adjustments: Reducing ‘ $maxlen$ ’ typically increases the number of data chunks available for training, doubling the sample size and necessitating an adjustment in training epochs to maintain balance. This adjustment proved effective, with models showing robust performance even when ‘ $maxlen$ ’ was increased to as much as 1024, see table 7.9.
6. Hierarchical Attention Transformer (HAT) Integration: Satisfied with the sliding window encoder’s performance, we aimed to incorporate HAT architecture to capture long-range dependencies across document segments better. The idea was to use the sliding window as segment encoder block and then design a hierarchical structure on top of this by adding Cross segment encoder layer to mimic the interleaving architecture. Modifications included random CLS token initialization for segment-wise embedding, allowing the model to form a comprehensive representation of cross-segment relationships, see figure 7.8. The experiments were performed on stacks of 2 and 3 interleaving layers, see table 7.10 and 7.11.

We propose the Sliding Window Transformer as our final architecture to capture the sequential information required for our analysis. While Hierarchical Attention Transformers (HATs) [CDF⁺22] theoretically capture long-term dependencies that might be beneficial for identifying labels in

Table 7.6: Impact of Encoder Blocks (with $maxlen=16$, $heads=20$, ff -dim= $1\times$) on Transformer model based on results from Table 7.4, 7.2, 7.3

Encoders	Train Loss	#Params	Accuracy (%)	Mean F_1 (%)
1	0.3906	10.51M	85.46	84.93
2	0.3890	20.35M	84.91	84.67

Table 7.7: Impact of SW mechanism of (window size=16) applied to encoder architecture found in Table 7.5

Max Len	Accuracy (%)	Mean F ₁ (%)
1024	82.23	80.82
512	82.84	80.68
256	83.04	80.62
128	84.45	83.22
64	86.26	85.51
32	86.59	85.97
16	86.33	85.65

Table 7.8: Impact of Window Size (Maxlen =32) on Transformer model based on the results from Table 7.7

Sliding Window	Train Loss	Accuracy (%)	Mean F ₁ (%)
32	0.3728	85.39	84.537
16	0.3496	86.78	86.079
8	0.3639	86.15	85.438
4	0.3903	84.55	83.361

Table 7.9: Increasing the number of Epochs to counter smaller maxlen (SW=16) based on the results from Table 7.8

Maxlen	Validation Samples	Epochs	Train Loss	Accuracy (%)	Mean F ₁ (%)
32	18K	1	0.3465	86.80	86.21
64	10K	2	0.3415	86.70	86.05
128	6K	4	0.3352	87.30	86.76
256	5K	8	0.3157	87.33	86.99
512	4K	16	0.2771	87.52	86.58
1024	4K	32	0.2726	87.81	87.18
2048	4K	32	0.2692	86.58	85.41

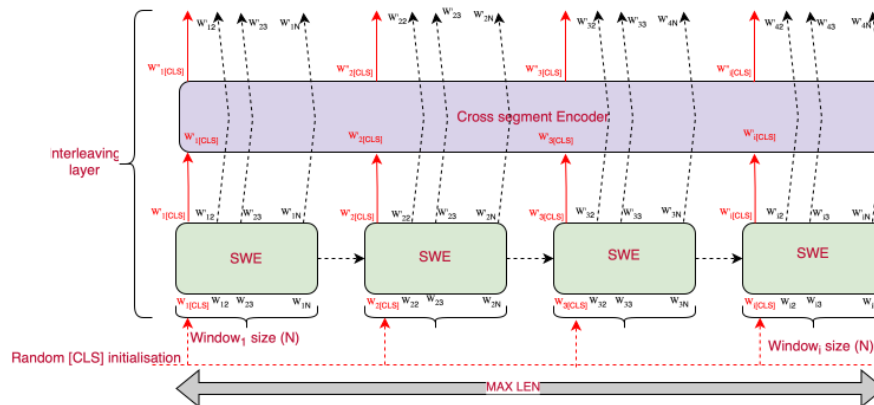


Figure 7.8: HAT interleaving layer in our setting

Table 7.10: HAT Performance with **2 interleaving layers** ($SWE=1$, $CWE=1$, and for Max Len 1024 and 32 epochs, similar to the training configurations reported in Table 7.9)

Window Size	Params (M)	Train Loss	Accuracy (%)	Mean F ₁ (%)
32	47	0.3415	86.58	85.93
64	47	0.3006	86.44	85.47
128	47	0.2990	85.18	84.10
256	47	0.3279	87.52	86.58
512	47	0.5040	79.81	78.03

Table 7.11: HAT Performance with **3 interleaving layers** ($SWE=1$, $CWE=1$, and for Max Len 1024 and 32 epochs, similar to the training configurations reported in Table 7.9)

Window Size	Params(M)	Train Loss	Accuracy (%)	Mean F ₁ (%)
32	70	0.2917	86.43	85.78
64	70	0.2758	86.70	86.05
128	70	0.2871	86.62	85.90
256	70	0.4460	80.45	78.96
512	70	0.5304	76.95	75.46

tasks involving extensive cross-segment relationships, our specific task does not generally demand such extensive long-term dependencies. This could be partially explained by our high results on a smaller and much more local context span. The focus primarily remains on local contextual relationships within the document, similar to how a human would examine a few paragraphs to build the context. Hence, while HATs might not be strictly necessary for our immediate needs, we believe exploring their potential benefits in our experimental setup is worthwhile to ensure comprehensive coverage of all possible interaction dynamics within the document. Therefore, it is necessary to validate the efficiency of simpler models (Sliding Window Transformer) against more complex architectures (HATs) to ensure that our solution is robust and parameter-efficient across various scenarios.

Discussions and Conclusions

Contenu du chapitre

8.1 Introduction	127
8.2 Conclusions	128
8.3 Post-Hoc Explanations	129
8.4 Limitations	129
8.5 Ethical considerations	131

8.1 Introduction

This section summarises the findings from earlier analyses and presents a comprehensive review of the overall results. This synthesis highlights the performance metrics and effectiveness of the evaluated models and addresses their classification capabilities. Additionally, we discuss the broader implications of our research, including potential limitations and ethical considerations. This holistic approach ensures that we not only present empirical data but also critically assess the impact and responsible application of our findings in real-world scenarios.

This thesis introduces the Sliding Window (SW) Transformer as the optimal approach for capturing sequential paragraph information and integrating it with multimodal embeddings via a cross-modal attention model. This method significantly enhances both the overall and individual performances of unimodal models, illustrating the effectiveness of our sequential strategy. The proposed model is entirely deep learning-based (see table 8.1), eliminating manual feature engineering and enhancing accuracy through data-driven learning.

The architecture of each model varies, with each configuration tailored to specific dimensions and outputs, emphasizing our focus on modularity. This allows the models to be adaptable and efficient across different scenarios and data modality.

To establish a robust evaluation framework, we incorporate three straightforward baselines inspired by relevant literature, ensuring a comprehensive assessment of our model’s capabilities:

1. Top-k first words: This method is inspired by strategies used in existing literature where the initial words of paragraphs play a crucial role in classification. Here, a vocabulary of top- k unique words for each class is created. A paragraph’s initial word is checked against this vocabulary to assign a label, simulating approaches focusing on key initial terms to determine content classification.
2. Text classifier from previous work [MPS21]: We adopt a text classifier previously fine-tuned for similar tasks, which processes text lines extracted directly from pdfalto (see Chapter 4). This classifier is adept at handling standard classifications but does not account for the “overlap” class, highlighting its limitation in dealing with complex labelling scenarios.
3. Dummy classifier: Predicts the majority class (Basic) to highlight class imbalance in the dataset.

These baselines serve as comparative benchmarks to validate the effectiveness of our proposed SW Transformer, underlining its superior capability in handling complex document structures and diverse data integration challenges.

Table 8.1: Overall performance comparison (accuracy and mean F_1 over the three classes basic, theorem, and proof) of individual modality models and multimodal model, with and without the sequential approach, for each model, the number of batches (1 000 PDF documents, roughly 200k samples) it was trained on is indicated (here + indicates additional batches on which further training of sequential paragraph model)

Modality	Model chosen	Seq. approach	#Batches	#Params (M)	Accuracy (%)	Mean F_1 (%)
Dummy	always predicts <i>basic</i>	—	—	—	59.41	24.85
Top-k first word	use only first word	—	—	—	52.84	44.20
Line-based [MPS21]	Bert (fine-tuned)	—	—	110	57.31	55.71
Font	LSTM 128 cells	-	11	2	64.93	45.48
		CRF	11+8	2	71.15	64.51
		SW Transformer	11+8	2	76.22	71.78
Vision	EfficientNetV2m_avg	-	9	53	69.44	59.96
		CRF	9+8	53	74.63	70.82
		SW Transformer	9+8	65	79.59	77.66
Text	Pretrained RoBERTa-like	-	20	124	76.46	72.39
		CRF	20+8	124	83.13	81.00
		SW Transformer	20+8	129	87.50	86.67
Multimodal	Cross-modal attention	-	2	185	78.50	75.37
		CRF	2+8	185	84.39	82.91
		SW Transformer	2+8	198	87.81	87.23
		HAT	2+8	232	87.52	86.58

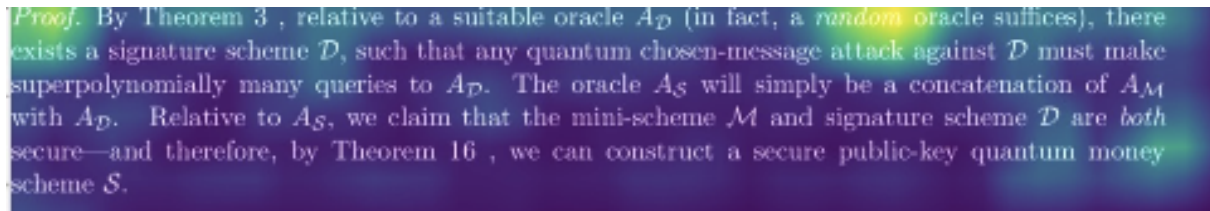
8.2 Conclusions

The results from our investigation into various models for document analysis highlight several key findings:

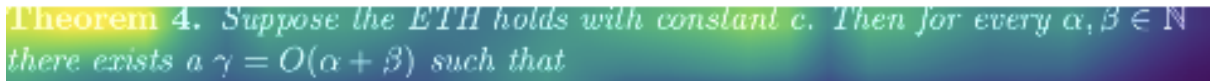
1. **Difficulty of Task:** The highest accuracy achieved was 88%, with a mean F_1 score of 87%. This underscores the inherent challenge of the task, similar to difficulties faced by human annotators in distinguishing between proofs and theorems, especially within document segments. The complexity of the task sets a natural limit on performance, preventing near-perfect outcomes.
2. **Performance of Unimodal Models:**
 - (a) **Font-Based Model:** This model showed the weakest performance among the tested models, albeit still surpassing a baseline dummy model. This indicates that while it does learn, it captures less decisive features for this task.
 - (b) **Text-Based Model:** This emerged as the most effective unimodal approach, reinforcing that textual data provides the most significant clues for distinguishing elements within scientific documents.
3. **Advantages of Multimodal Models:** The multimodal approach, which integrates signals from multiple data types, outperformed all unimodal strategies, albeit by a narrow margin over the text-based model. This suggests that a combined model leverages complementary information from different sources effectively, though the primary gains are driven by textual analysis.
4. **Impact of Sequential Models:** Incorporating models that handle sequences, such as CRFs and Sliding Window Transformers, markedly improved performance across all models, unimodal and multimodal, by 5 to 10 points in terms of accuracy and mean F_1 score. This highlights the importance of capturing sequential dependencies in document structure analysis.

These insights are instrumental in guiding future research and application development, particularly in enhancing model architectures to handle better the complexities of document analysis in legal, scientific, and technical domains.

To effectively gauge the performance of various language models – our custom-pretrained model, RoBERTa, and SciBERT – we conducted assessments based on accuracy and mean F_1 scores, as detailed in Table 5.18. These evaluations took place on a validation dataset, revealing that all models, despite having similar parameter counts and inference times, exhibited close accuracy (76.465% to 76.97%) and mean F_1 scores (71.84% to 72.87%), with convergence achieved after 20 batches. Notably, SciBERT showed a slight edge in performance.



Proof. By Theorem 3 , relative to a suitable oracle $A_{\mathcal{D}}$ (in fact, a *random* oracle suffices), there exists a signature scheme \mathcal{D} , such that any quantum chosen-message attack against \mathcal{D} must make superpolynomially many queries to $A_{\mathcal{D}}$. The oracle $A_{\mathcal{S}}$ will simply be a concatenation of $A_{\mathcal{M}}$ with $A_{\mathcal{D}}$. Relative to $A_{\mathcal{S}}$, we claim that the mini-scheme \mathcal{M} and signature scheme \mathcal{D} are *both* secure—and therefore, by Theorem 16 , we can construct a secure public-key quantum money scheme \mathcal{S} .



Theorem 4. *Suppose the ETH holds with constant c . Then for every $\alpha, \beta \in \mathbb{N}$ there exists a $\gamma = O(\alpha + \beta)$ such that*

Figure 8.1: Gradcam visualizations [SCD⁺17] with image classifiers

Additionally, our pretrained model required significantly fewer data points for fine-tuning to reach predetermined accuracy thresholds (e.g., 65% or 70%) compared to RoBERTa, which was trained on a corpus 15 times larger. While SciBERT also outperformed our model, it was trained on a dataset 5.5 times larger, consisting exclusively of scientific papers, which likely contributed to its enhanced performance.

Turning to vision-based models, as summarized in Table 5.14, various pooling methods (none, max, average) were tested on the simplest model to evaluate their impact. Interestingly, models without pooling performed comparably to more complex models, with average pooling yielding the best results across the scenarios tested cutting a major chunk of parameters.

To explore font sequence information, we experimented with different configurations of LSTM cells, including a switch to GRU cells and implementing a Bidirectional LSTM to enhance the capture of sequential information. According to our findings in Table 5.7, incorporating a bidirectional component into the font models showed only modest gains in label accuracy for text blocks, suggesting a limited impact on overall performance.

8.3 Post-Hoc Explanations

While our models inherently lack direct interpretability due to the black-box nature of deep learning approaches, we endeavour to offer some explanations through visualization techniques. For instance, we utilize Grad-CAM visualizations [SCD⁺17] (see Figure 8.1) to illustrate which parts of an image our model focuses on before determining its label. The visualizations reveal that the model concentrates on italic and bold text within the images.

For our NLP-based transformer model, we visualize the attention weights of the last layer to understand the model’s focus within the text. Typically, the CLS token primarily attends to the initial tokens of a sentence, providing insights into what the model deems vital for prediction (see Figure 8.2).

Although our font-based model does not lend itself to straightforward post hoc explanations, it holds a unique advantage in scenarios involving texts in different languages. The font model may retain effectiveness since it identifies fonts independent of language content, potentially outperforming vision and language models when dealing with non-English texts (see Figure 8.3). This adaptability underscores the model’s utility in diverse linguistic environments.

8.4 Limitations

Our proposed approach achieves an accuracy greater than 85%, though it is important to note that the task remains challenging, even for human annotators. This model is a foundational step towards constructing a comprehensive knowledge base of mathematical theorems from PDF documents, though substantial further research is necessary to realize this application fully.

The model’s training and testing were conducted exclusively on English-language mathematical articles. While these constitute a significant portion of the academic literature, extending the model’s applicability to articles in other languages, such as Russian or Arabic, would require additional verification.

Furthermore, the ground-truth dataset was derived from arXiv papers written in L^AT_EX and compatible with modern L^AT_EX compilers. This specificity means the model’s performance might not translate directly

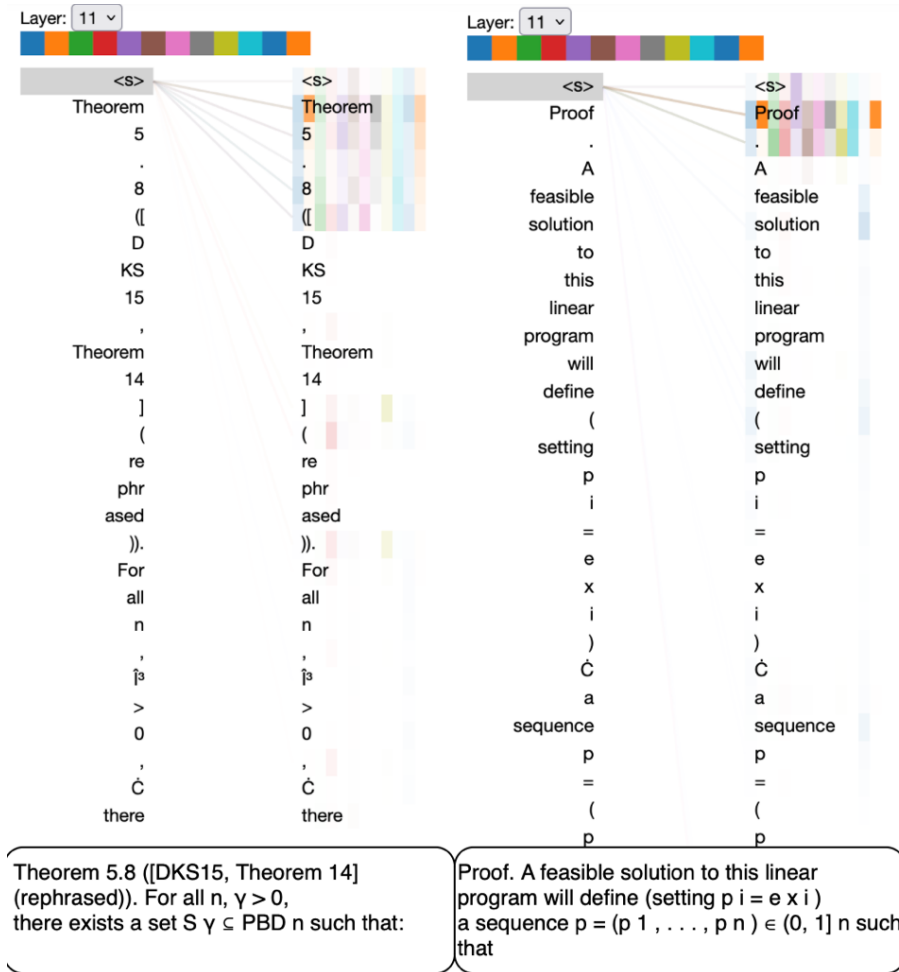


Figure 8.2: Visualisation of attention weights of the last layer

Теорема 3 При $\log_n k = o(n)$ в классе функций P_k^n существует функция, минимальная ДНФ реализация которой содержит не менее N_k конъюнкций, где для N_k выполнено

$$N_k \geq \Omega\left(\frac{nk \log n}{\log k}\right).$$

Доказательство. Указанную оценку можно получить оценив сложность покрытия околонулевых точек функции принимающей нулевые значения на одном из слоев булева куба. ■

Figure 8.3: An example where font model retains effectiveness

to other formats like scanned documents, articles composed in Microsoft Word, or articles using uncommon document classes, where different preprocessing techniques such as OCR might be necessary.

8.5 Ethical considerations

We foresee no significant ethical concerns in developing machine learning models to extract mathematical statements and proofs from academic articles. However, users should exercise caution when employing these models in scenarios that demand absolute accuracy.

Training machine learning and deep learning models necessitates substantial computational resources, contributing to energy consumption and potentially to greenhouse gas emissions. We have endeavoured to mitigate this impact by opting for models with efficient parameter counts. Moreover, the main supercomputer utilized in our computations (specifically, the Jean Zay supercomputer hosted by IDRIS and managed by GENCI) operates on low-carbon nuclear-powered electricity, and the residual heat they generate is repurposed into an urban heating network.

While our training dataset comprises publicly accessible arXiv articles adhering to arXiv’s terms and conditions, redistribution of this dataset is restricted by the licensing terms of arXiv, barring some exceptions for documents under Creative Commons licenses. The legal implications of distributing models trained on such datasets remain ambiguous.

Classification Report

Utilizing classification reports will be invaluable to effectively showcase the outcomes of the experiments conducted throughout the thesis. These reports can precisely illuminate key findings across the various models tested, pinpointing specific classes where models may underperform. This detailed presentation not only aids debugging but also provides insights into potential biases within the models. In the subsequent sections, we will present several classification reports, offering a structured analysis of each model’s performance across different classifications. This methodological approach ensures a comprehensive understanding of each model’s strengths and weaknesses, contributing significantly to the overall research assessment.

For a more global view of the classification report (including the best models of each modality only), see Table A.1. This table is useful as it highlights the importance of each modality for specific classes. For example, the font modality excels in identifying ‘*Theorem*’ classes, while the vision modality improves detection for both ‘*Theorem*’ and ‘*Proof*’ classes.

Table A.1: Class-wise precision and recall scores for best unimodal, multimodal, and sequential models

	<i>Font</i>		<i>Vision</i>		<i>Text</i>		<i>Multimodal</i>		<i>Sequence CRF</i>		<i>Sequence Transformers</i>	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Basic	0.6534	0.9750	0.6902	0.9119	0.7963	0.8539	0.7953	0.8863	0.8538	0.8993	0.9047	0.8970
Theorem	0.8657	0.3770	0.7778	0.6158	0.7498	0.6038	0.8561	0.6845	0.8569	0.8019	0.8768	0.9030
Proof	0.5039	0.0375	0.6086	0.2223	0.6860	0.6717	0.7129	0.6184	0.8022	0.7570	0.8157	0.8376

A.1 Font Sequence Models

Table A.2: Classification Report of the LSTM-128 model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6534	0.9750	0.7694	314501
Proof	0.5039	0.0375	0.0698	125524
Theorem	0.8657	0.3770	0.5252	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6493	529296
macro avg	0.5012	0.3474	0.3411	529296
weighted avg	0.6374	0.6493	0.5589	529296

Table A.3: Classification Report of the GRU-128 model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6354	0.8611	0.7312	314501
Proof	0.3146	0.1565	0.2090	125524
Theorem	0.7444	0.3527	0.4787	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6059	529296
macro avg	0.4236	0.3426	0.3547	529296
weighted avg	0.5728	0.6059	0.5616	529296

Table A.4: Classification Report of the BiLSTM-128 Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6344	0.9729	0.7680	314501
Proof	0.4806	0.0490	0.0889	125524
Theorem	0.8891	0.3544	0.5067	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6471	529296
macro avg	0.5010	0.3441	0.3409	529296
weighted avg	0.6351	0.6471	0.5596	529296

Table A.5: Classification Report of the BiGRU-128 Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6415	0.9264	0.7581	314501
Proof	0.4387	0.1128	0.1795	125524
Theorem	0.8133	0.4009	0.5370	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6422	529296
macro avg	0.4734	0.3600	0.3687	529296
weighted avg	0.6170	0.6422	0.5801	529296

A.2 Vision-Based Models

Table A.6: Classification Report of the *efficientnetB05* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6414	0.9667	0.7711	314501
Proof	0.3142	0.0092	0.0178	125524
Theorem	0.7809	0.4698	0.5866	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6527	529296
macro avg	0.4341	0.3614	0.3439	529296
weighted avg	0.5822	0.6527	0.5575	529296

Table A.7: Classification Report of the *efficientnetB0_max5* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.5902	0.9766	0.7357	314501
Proof	0.1344	0.0073	0.0139	125524
Theorem	0.1628	0.0008	0.0016	85801
Overlap	0.0094	0.0043	0.0059	3470
accuracy			0.5822	529296
macro avg	0.2242	0.2473	0.1893	529296
weighted avg	0.4090	0.5822	0.4408	529296

Table A.8: Classification Report of the *efficientnetB0_avg5* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6206	0.9810	0.7602	314501
Proof	0.6232	0.0142	0.0277	125524
Theorem	0.7780	0.2656	0.3961	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6293	529296
macro avg	0.5054	0.3152	0.2960	529296
weighted avg	0.6427	0.6293	0.5225	529296

Table A.9: Classification Report of the *efficientnetB4_avg5* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6449	0.9663	0.7735	314501
Proof	0.5394	0.0223	0.0428	125524
Theorem	0.7944	0.4891	0.6054	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6587	529296
macro avg	0.4947	0.3694	0.3554	529296
weighted avg	0.6399	0.6587	0.5679	529296

Table A.10: Classification Report of the *efficientnetB7_avg5* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6424	0.8746	0.7408	314501
Proof	0.0526	0.0000	0.0001	125524
Theorem	0.4914	0.5708	0.5281	85801
Overlap	0.0097	0.0037	0.0054	3470
accuracy			0.6122	529296
macro avg	0.2990	0.3622	0.3186	529296
weighted avg	0.4739	0.6122	0.5258	529296

Table A.11: Classification Report of the *efficientnetv2s_avg5* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.5942	1.0000	0.7454	314501
Proof	0.0000	0.0000	0.0000	125524
Theorem	0.0000	0.0000	0.0000	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.5942	529296
macro avg	0.1485	0.2500	0.1864	529296
weighted avg	0.3531	0.5942	0.4429	529296

Table A.12: Classification Report of the *efficientnetv2m_avg5* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6408	0.9774	0.7741	314501
Proof	0.0000	0.0000	0.0000	125524
Theorem	0.8162	0.3659	0.5053	85801
Overlap	0.0080	0.0256	0.0122	3470
accuracy			0.6402	529296
macro avg	0.3662	0.3422	0.3229	529296
weighted avg	0.5131	0.6402	0.5419	529296

Table A.13: Classification Report of the *efficientnetB4_avg9* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6639	0.9648	0.7866	314501
Proof	0.7128	0.1057	0.1841	125524
Theorem	0.8556	0.5326	0.6565	85801
Overlap	0.0351	0.0023	0.0043	3470
accuracy			0.6847	529296
macro avg	0.5668	0.4014	0.4079	529296
weighted avg	0.7025	0.6847	0.6175	529296

Table A.14: Classification Report of the *efficientnetv2s_avg9* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.5967	0.9986	0.7471	314501
Proof	0.0000	0.0000	0.0000	125524
Theorem	0.8337	0.0289	0.0558	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.5981	529296
macro avg	0.3576	0.2569	0.2007	529296
weighted avg	0.4897	0.5981	0.4529	529296

Table A.15: Classification Report of the *efficientnetv2m_avg9* Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.6902	0.9119	0.7857	314501
Proof	0.6086	0.2223	0.3257	125524
Theorem	0.7778	0.6158	0.6874	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.6944	529296
macro avg	0.5192	0.4375	0.4497	529296
weighted avg	0.6805	0.6944	0.6555	529296

A.3 NLP Models

Table A.16: Classification Report of the roberta_base_ft Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7791	0.8817	0.8272	314501
Proof	0.7223	0.5968	0.6536	125524
Theorem	0.7530	0.6108	0.6745	85801
Overlap	0.5714	0.0081	0.0159	3470
accuracy			0.7645	529296
macro avg	0.7064	0.5243	0.5428	529296
weighted avg	0.7600	0.7645	0.7560	529296

Table A.17: Classification Report of the roberta_from_scratch Performance

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7963	0.8539	0.8241	314501
Proof	0.6860	0.6717	0.6788	125524
Theorem	0.7498	0.6038	0.6689	85801
Overlap	0.5278	0.0055	0.0108	3470
accuracy			0.7646	529296
macro avg	0.6900	0.5337	0.5457	529296
weighted avg	0.7608	0.7646	0.7591	529296

Table A.18: Classification Report of the scibert_base_ft Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7962	0.8657	0.8295	314501
Proof	0.7108	0.6447	0.6761	125524
Theorem	0.7375	0.6318	0.6806	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.7697	529296
macro avg	0.5611	0.5356	0.5465	529296
weighted avg	0.7612	0.7697	0.7635	529296

A.4 Multimodal Models

Table A.19: Classification Report of the Concatenated Model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7868	0.8871	0.8350	314501
Proof	0.7064	0.6187	0.6592	125524
Theorem	0.8487	0.6502	0.7360	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.7790	529296
macro avg	0.5857	0.5388	0.5576	529296
weighted avg	0.7736	0.7790	0.7718	529296

Table A.20: Classification Report of the Docker + concat

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8034	0.8697	0.8355	314501
Proof	0.7054	0.6460	0.6742	125524
Theorem	0.8011	0.6909	0.7419	85801
Overlap	0.3434	0.0882	0.1403	3470
accuracy			0.7812	529296
macro avg	0.6633	0.5732	0.5979	529296
weighted avg	0.7768	0.7812	0.7767	529296

Table A.21: Classification Report of the docker+fusion(768) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7983	0.8826	0.8383	314501
Proof	0.7107	0.6254	0.6653	125524
Theorem	0.8417	0.6895	0.7580	85801
Overlap	0.3354	0.0789	0.1278	3470
accuracy			0.7850	529296
macro avg	0.6715	0.5691	0.5974	529296
weighted avg	0.7815	0.7850	0.7796	529296

Table A.22: Classification Report of the *docker+fusion(1280)* model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7930	0.8876	0.8376	314501
Proof	0.7187	0.6123	0.6613	125524
Theorem	0.8489	0.6854	0.7584	85801
Overlap	0.3333	0.1037	0.1582	3470
accuracy			0.7844	529296
macro avg	0.6735	0.5723	0.6039	529296
weighted avg	0.7814	0.7844	0.7785	529296

Table A.23: Classification Report of the *docker+fusion(2304)* model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7920	0.8903	0.8383	314501
Proof	0.7217	0.6023	0.6566	125524
Theorem	0.8429	0.6902	0.7590	85801
Overlap	0.3537	0.0749	0.1237	3470
accuracy			0.7842	529296
macro avg	0.6776	0.5645	0.5944	529296
weighted avg	0.7807	0.7842	0.7777	529296

Table A.24: Classification Report of the *bilinear mechanism*

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7956	0.8785	0.8350	314501
Proof	0.7004	0.6142	0.6545	125524
Theorem	0.8304	0.6905	0.7540	85801
Overlap	0.2806	0.0496	0.0843	3470
accuracy			0.7799	529296
macro avg	0.6518	0.5582	0.5820	529296
weighted avg	0.7753	0.7799	0.7741	529296

Table A.25: Classification Report of the *docker+ bilinear gated (1280)* model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7933	0.8846	0.8365	314501
Proof	0.7413	0.6186	0.6682	125524
Theorem	0.8471	0.6795	0.7514	85801
Overlap	0.2930	0.0899	0.1376	3470
accuracy			0.7830	529296
macro avg	0.6689	0.5681	0.5980	529296
weighted avg	0.7800	0.7830	0.7774	529296

Table A.26: Classification Report of the docker+GMU(1280) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8202	0.8459	0.8328	314501
Proof	0.6737	0.6914	0.6824	125524
Theorem	0.8038	0.7031	0.7504	85801
Overlap	0.2793	0.0818	0.1266	3470
accuracy			0.7812	529296
macro avg	0.6442	0.5807	0.5981	529296
weighted avg	0.7792	0.7812	0.7792	529296

Table A.27: Classification Report of the docker+cross modal att(1280) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7953	0.8863	0.8383	314501
Proof	0.7129	0.6184	0.6623	125524
Theorem	0.8561	0.6845	0.7607	85801
Overlap	0.3258	0.1248	0.1805	3470
accuracy			0.7850	529296
macro avg	0.6725	0.5785	0.6105	529296
weighted avg	0.7825	0.7850	0.7797	529296

Table A.28: Classification Report of the docker+multihead(1280) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7998	0.8759	0.8361	314501
Proof	0.7135	0.6284	0.6682	125524
Theorem	0.8190	0.6977	0.7534	85801
Overlap	0.3465	0.1184	0.1769	3470
accuracy			0.7834	529296
macro avg	0.6697	0.5802	0.6087	529296
weighted avg	0.7794	0.7833	0.7786	529296

Table A.29: Classification Report of the EmbraceNet(1280) balanced model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8069	0.8590	0.8322	314501
Proof	0.6748	0.6590	0.6672	125524
Theorem	0.8149	0.6909	0.7482	85801
Overlap	0.3204	0.0096	0.0185	3470
accuracy			0.7773	529296
macro avg	0.6543	0.5523	0.5665	529296
weighted avg	0.7737	0.7773	0.7739	529296

Table A.30: Classification Report of the EmbraceNet(1280) unbalanced model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8008	0.8675	0.8328	314501
Proof	0.6865	0.6382	0.6614	125524
Theorem	0.8156	0.6811	0.7423	85801
Overlap	0.3053	0.0251	0.0463	3470
accuracy			0.7774	529296
macro avg	0.6521	0.5530	0.5707	529296
weighted avg	0.7729	0.7774	0.7724	529296

Table A.31: Classification Report of the docker+fusion(2304)+fusion(768) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7952	0.8866	0.8384	314501
Proof	0.7187	0.6130	0.6616	125524
Theorem	0.8410	0.6930	0.7600	85801
Overlap	0.3238	0.0813	0.1299	3470
accuracy			0.7850	529296
macro avg	0.6696	0.5685	0.5974	529296
weighted avg	0.7814	0.7850	0.7790	529296

Table A.32: Classification Report of the docker+ cross-modal att (1280)+fusion(768) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7955	0.8861	0.8384	314501
Proof	0.7193	0.6104	0.6604	125524
Theorem	0.8369	0.6940	0.7587	85801
Overlap	0.3052	0.1144	0.1664	3470
accuracy			0.7845	529296
macro avg	0.6642	0.5762	0.6060	529296
weighted avg	0.7809	0.7845	0.7788	529296

Table A.33: Classification Report of the docker+GMU(1280)+fusion(768) model

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7924	0.8901	0.8384	314501
Proof	0.7151	0.6109	0.6589	125524
Theorem	0.8598	0.6827	0.7610	85801
Overlap	0.3513	0.0639	0.1082	3470
accuracy			0.7849	529296
macro avg	0.6796	0.5619	0.5917	529296
weighted avg	0.7821	0.7849	0.7785	529296

A.5 Sequential Paragraph Models

A.5.1 CRF-Based

Table A.34: Classification Report of font+CRF

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7353	0.8652	0.7949	314501
Proof	0.5923	0.5062	0.5459	125524
Theorem	0.7883	0.4772	0.5945	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.7115	529296
macro avg	0.5289	0.4622	0.4838	529296
weighted avg	0.7051	0.7115	0.6919	529296

Table A.35: Classification Report of vis+CRF

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7629	0.8628	0.8098	314501
Proof	0.6413	0.5234	0.5764	125524
Theorem	0.8145	0.6757	0.7386	85801
Overlap	0.0000	0.0000	0.0000	3470
accuracy			0.7463	529296
macro avg	0.5547	0.5155	0.5312	529296
weighted avg	0.7374	0.7463	0.7376	529296

Table A.36: Classification Report of nlp+CRF

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8443	0.8907	0.8669	314501
Proof	0.8029	0.7595	0.7806	125524
Theorem	0.8185	0.7492	0.7825	85801
Overlap	0.4729	0.0277	0.0523	3470
accuracy			0.8313	529296
macro avg	0.7346	0.6068	0.6206	529296
weighted avg	0.8279	0.8313	0.8274	529296

Table A.37: Classification Report of multimodal+CRF

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8538	0.8993	0.8756	314501
Proof	0.8022	0.7570	0.7789	125524
Theorem	0.8569	0.8019	0.8332	85801
Overlap	0.5639	0.0216	0.0416	3470
accuracy			0.8439	529296
macro avg	0.7715	0.6199	0.6323	529296
weighted avg	0.8416	0.8439	0.8404	529296

A.5.2 Transformer-Based

Table A.38: Classification Report of font+SW transformer

Class	Precision	Recall	F ₁ -score	Support
Basic	0.7797	0.8799	0.8265	314501
Proof	0.6737	0.6049	0.6375	125524
Theorem	0.8272	0.5900	0.6894	85801
Overlap	0.7027	0.0075	0.0148	3470
accuracy			0.7622	529296
macro avg	0.7457	0.5209	0.5420	529296
weighted avg	0.7614	0.7622	0.7541	529296

Table A.39: Classification Report of vis+SW transformer

Class	Precision	Recall	F ₁ -score	Support
Basic	0.8186	0.8665	0.8417	314501
Proof	0.6992	0.6193	0.6572	125524
Theorem	0.8344	0.8280	0.8310	85801
Overlap	0.4332	0.0233	0.0443	3470
accuracy			0.7959	529296
macro avg	0.6964	0.5843	0.5936	529296
weighted avg	0.7904	0.7959	0.7900	529296

Table A.40: Classification Report of nlp+SW transformer

Class	Precision	Recall	F ₁ -score	Support
Basic	0.9101	0.8862	0.8980	314501
Proof	0.8018	0.8746	0.8367	125524
Theorem	0.8108	0.8605	0.8656	85801
Overlap	0.6145	0.2412	0.3446	3470
accuracy			0.8750	529296
macro avg	0.7794	0.7156	0.7367	529296
weighted avg	0.8761	0.8750	0.8746	529296

Table A.41: Classification Report of multimodal+SW transformer

Class	Precision	Recall	F ₁ -score	Support
Basic	0.9047	0.8970	0.9008	314501
Proof	0.8157	0.8376	0.8265	125524
Theorem	0.8768	0.9030	0.8897	85801
Overlap	0.6403	0.0888	0.1559	3470
accuracy			0.8781	529296
macro avg	0.8094	0.6818	0.6930	529296
weighted avg	0.8773	0.8781	0.8763	529296

Bibliography

- [ABMS19] Antoine Amarilli, Pierre Bourhis, Mikaël Monet, and Pierre Senellart. Evaluating datalog via tree automata and cycluits. *Theory of Computing Systems*, 63:1620–1678, 2019. 15
- [ACG19] Rabah A. Al-Zaidy, Cornelia Caragea, and C. Lee Giles. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2551–2557. ACM, 2019. 116, 117
- [ASMyGG20] John Arevalo, Thamar Solorio, Manuel Montes-y Gómez, and Fabio A González. Gated multimodal networks. *Neural Computing and Applications*, 32, 2020. 25, 26, 28, 108, 109, 110, 111
- [ASP+23] Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. Llemma: An open language model for mathematics. *CoRR*, abs/2310.10631, 2023. 93, 94
- [BCB15] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 89
- [BKH16] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 82
- [BLC19] Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: A pretrained language model for scientific text. In *EMNLP/IJCNLP*, 2019. 48, 91, 99
- [BMR+20] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 91, 93
- [BPC20] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. 3, 10, 118
- [Bri22] Yacine Brihmouche. TheoremKB : une base de connaissance des résultats mathématiques. Master’s thesis, Paris IX Dauphine, September 2022. 23, 24
- [BWL20] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal speed and accuracy of object detection. *arXiv:2004.10934*, 2020. 21, 48
- [CDF+22] Ilias Chalkidis, Xiang Dai, Manos Fergadiotis, Prodromos Malakasiotis, and Desmond Elliott. An exploration of hierarchical attention transformers for efficient long document classification. *arXiv preprint arXiv:2210.05529*, 2022. 3, 10, 29, 119, 120, 123

- [CG16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016. 106
- [CHL05] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pages 539–546. IEEE Computer Society, 2005. 23
- [CKB⁺21] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. 94
- [CL19] Jun-Ho Choi and Jong-Seok Lee. Embracenet: A robust deep learning architecture for multimodal classification. *Information Fusion*, 51:259–270, 2019. 28, 108, 109, 110, 111
- [Cro24] Crossref. Crossref: Linking scholarly literature. <https://www.crossref.org>, 2024. Accessed on 2024-03-15. 13
- [CSC⁺13] Ricardo Chavarriaga, Hesam Sagha, Alberto Calatroni, Sundara Tejaswi Digumarti, Gerhard Tröster, José del R. Millán, and Daniel Roggen. The opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognit. Lett.*, 34(15):2033–2042, 2013. 109
- [DBK⁺21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 82, 83
- [DCLT19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HCT*, 2019. 48, 89, 92, 96, 117
- [DDS⁺09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 75
- [Del20] Theo Delemazure. A Knowledge Base of Mathematical Results. Master’s thesis, Ecole Normale Supérieure (ENS), September 2020. 14, 18, 19
- [Den12] Li Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 75
- [DPR19] Tyler Dauphinee, Nikunj Patel, and Mohammad Rashidi. Modular multimodal architecture for document classification. *arXiv preprint arXiv:1912.04376*, 2019. 105
- [DS07] Nilesh Dalvi and Dan Suciu. The dichotomy of conjunctive queries on probabilistic structures. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 293–302, 2007. 14
- [Fre18] FreeDesktop. Poppler. <https://poppler.freedesktop.org/>, 2018. 32
- [GD23] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023. 118
- [GDDM14] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 580–587. IEEE Computer Society, 2014. 48
- [Gir15] Ross B. Girshick. Fast R-CNN. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1440–1448. IEEE Computer Society, 2015. 48
- [GM20] Deyan Ginev and Bruce R. Miller. Scientific statement classification over arXiv.org. In *LREC*, 2020. 61, 62

- [GS23] Antoine Gauquier and Pierre Senellart. Automatically inferring the document class of a scientific article. In *Proceedings of the ACM Symposium on Document Engineering 2023, DocEng 2023, Limerick, Ireland, August 22-25, 2023*, pages 11:1–11:10. ACM, 2023. 24, 25
- [HBM⁺22a] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. 96
- [HBM⁺22b] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *CoRR*, abs/2203.15556, 2022. 92
- [Her12] André Hernich. Computing universal models under guarded tgds. In *Proceedings of the 15th International Conference on Database Theory*, pages 222–235, 2012. 14
- [HG16] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv:1606.08415*, 2016. 82
- [HLC⁺22] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. LayoutLMv3: Pre-training for document ai with unified text and image masking. In *ACM MM*, 2022. 1, 7, 105
- [HLVDMW17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 80
- [HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8), 1997. 25, 28, 48, 69, 74, 89, 101, 117
- [Hum24] HumanSignal. Label studio. <https://labelstud.io/>, 2024. 42, 102
- [HZC⁺17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv:1704.04861*, 2017. 76
- [HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 76, 78, 80
- [IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 76
- [Joc20] Glenn Jocher. ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements, October 2020. 40
- [JS23] Shufan Jiang and Pierre Senellart. Extracting definienda in mathematical scholarly articles with transformers. *CoRR*, abs/2311.12448, 2023. 27
- [Kar14] Simonyan Karen. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv: 1409.1556*, 2014. 78
- [KGJM18] Douwe Kiela, Edouard Grave, Armand Joulin, and Tomas Mikolov. Efficient large-scale multi-modal classification. In *AAAI*, 2018. 28, 110, 111
- [KPF01] LIII KPFERS. On lines and planes of closest fit to systems of points in space. In *Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems (SIGMOD)*, page 19, 1901. 65
- [Kri09] Alex Krizhevsky. Learning multiple layers of features from tiny images. Master’s thesis, University of Toronto, 2009. 75
- [KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012. 77, 78

- [KSZ14] Ryan Kiros, Ruslan Salakhutdinov, and Richard S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *CoRR*, abs/1411.2539, 2014. 106, 107
- [L⁺24a] Patrice Lopez et al. Grobid. <https://github.com/kermitt2/grobid>, 2008–2024. 2, 9, 37, 39, 60
- [L⁺24b] Patrice Lopez et al. pdfalto. <https://github.com/kermitt2/pdfalto>, 2024. 2, 9, 37
- [LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998. 77
- [LBPL19] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019. 28, 110
- [LLC⁺21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 83, 118
- [LMB⁺14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 47, 82
- [LMP01] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001. 28, 116
- [LMW⁺22] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *CVPR*, 2022. 82, 83
- [LOG⁺19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692*, 2019. 27, 28, 91, 96, 99
- [LXC⁺20] Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. Docbank: A benchmark dataset for document layout analysis. *arXiv preprint arXiv:2006.01038*, 2020. 36, 39
- [LXL⁺22] Junlong Li, Yiheng Xu, Tengchao Lv, Lei Cui, Cha Zhang, and Furu Wei. Dit: Self-supervised pre-training for document image transformer. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 3530–3539, 2022. 106
- [LXT⁺18] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6391–6401, 2018. 78, 79
- [LYK⁺20] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4), 2020. 91
- [LYL18] Xiaohui Li, Fei Yin, and Cheng-Lin Liu. Page object detection from PDF document images by deep structured prediction and supervised clustering. In *24th International Conference on Pattern Recognition, ICPR 2018, Beijing, China, August 20-24, 2018*, pages 3627–3632. IEEE Computer Society, 2018. 48, 52
- [MBD⁺24] Shrey Mishra, Yacine Brihmoche, Theo Delemazure, Antoine Gauquier, and Pierre Senellart. First steps in building a knowledge base of mathematical results. In *SDP Fourth Workshop on Scholarly Document Processing at ACL 2024*, 2024. 2, 8
- [MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 89, 106

- [Mea92] Al Mead. Review of the development of multidimensional scaling methods. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 41(1):27–39, 1992. 65
- [MGS23] Shrey Mishra, Antoine Gauquier, and Pierre Senellart. Multimodal machine learning for extraction of theorems and proofs in the scientific literature. *CoRR*, abs/2307.09047, 2023. 3, 9, 25, 26, 28
- [MHM18] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. 65
- [MKRA24] Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. Orca-math: Unlocking the potential of slms in grade school math. *CoRR*, abs/2402.14830, 2024. 94
- [MMS⁺20] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamel Seddah, and Benoît Sagot. CamemBERT: a tasty French language model. In *ACL*, 2020. 95
- [MPS21] Shrey Mishra, Lucas Pluvinae, and Pierre Senellart. Towards extraction of theorems and proofs in scholarly articles. In *DocEng*, 2021. 20, 23, 25, 45, 59, 88, 127, 128
- [Ope23] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. 94
- [PAD⁺22] Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S Nassar, and Peter Staar. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3743–3751, 2022. 39
- [PC19] Krutarth Patel and Cornelia Caragea. Exploring word embeddings in crf-based keyphrase extraction from research papers. In Mayank Kejriwal, Pedro A. Szekely, and Raphaël Troncy, editors, *Proceedings of the 10th International Conference on Knowledge Capture, K-CAP 2019, Marina Del Rey, CA, USA, November 19-21, 2019*, pages 37–44. ACM, 2019. 116
- [Pch19] Daria Pchelina. Information extraction from scientific papers. https://github.com/tooticki/IE_project, 2019. 17, 18
- [Plu20] Lucas Pluvinae. Extracting scientific results from research articles. Master’s thesis, Ecole Normale Supérieure (ENS), September 2020. 20, 21
- [PYGT21] Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. MathBERT: A pre-trained model for mathematical formula understanding. *arXiv:2105.00377*, 2021. 91
- [PZV⁺19] Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. Hierarchical transformers for long document classification. In *2019 IEEE automatic speech recognition and understanding workshop (ASRU)*, pages 838–844. IEEE, 2019. 117
- [RBC⁺21] Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d’Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew J. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorraine Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446, 2021. 96

- [RDGF16] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 779–788. IEEE Computer Society, 2016. 48
- [RGG⁺23] Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950, 2023. 93
- [RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 40
- [RHGS17] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017. 48
- [RKH⁺21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 105
- [RNS⁺18] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. Technical report, OpenAI, 2018. 89, 90
- [SCD⁺17] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 129
- [SDCW19] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv:1910.01108*, 2019. 23, 48, 95, 101
- [Sen19] Pierre Senellart. Theoremkb: Towards a knowledge base of mathematical results. <https://pierre.senellart.com/talks/sinfra-20191213.pdf>, 2019. 1, 7, 14
- [SHK⁺14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1), 2014. 71
- [SJ03] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16 [Neural Information Processing Systems, NIPS 2003, December 8-13, 2003, Vancouver and Whistler, British Columbia, Canada]*, pages 41–48. MIT Press, 2003. 27
- [SLJ⁺15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 76
- [SPP⁺19] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *CoRR*, abs/1909.08053, 2019. 92
- [SWWP⁺21] Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymanowska, Izabela Stefaniak, Michal Jarkiewicz, and Lukasz Okruszek. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research*, 304:114135, 2021. 89
- [SZ15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 106

- [TKC⁺22] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *CoRR*, abs/2211.09085, 2022. 94
- [TL19] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 81, 82, 102, 106
- [TL21] Mingxing Tan and Quoc Le. EfficientNetv2: Smaller models and faster training. In *ICML*, 2021. 25, 28, 81, 82, 87, 106
- [TLI⁺23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 93, 96
- [TMS⁺23] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. 93
- [VdMH08] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 65
- [VM23] Kira Vinogradova and Gene Myers. Local interpretable model-agnostic explanations for multitarget image regression. In Luca Longo, editor, *Joint Proceedings of the xAI-2023 Late-breaking Work, Demos and Doctoral Consortium co-located with the 1st World Conference on eXplainable Artificial Intelligence (xAI-2023), Lisbon, Portugal, July 26-28, 2023*, volume 3554 of *CEUR Workshop Proceedings*, pages 47–52. CEUR-WS.org, 2023. 56
- [VSP⁺17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 77, 89, 90, 91
- [WDH⁺23] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. ConvNeXt v2: Co-designing and scaling ConvNets with masked autoencoders. *arXiv:2301.00808*, 2023. 82, 83
- [WJD22] Jiapeng Wang, Lianwen Jin, and Kai Ding. Lilt: A simple yet effective language-independent layout transformer for structured document understanding. *arXiv preprint arXiv:2202.13669*, 2022. 1, 7, 105
- [WSM⁺18] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461, 2018. 90
- [WSW22] Xin Wei, Lamia Salsabil, and Jian Wu. Theory entity extraction for social and behavioral sciences papers using distant supervision. In Curtis Wigington, Matthew Hardy, Steven R. Bagley, and Steven J. Simske, editors, *Proceedings of the 22nd ACM Symposium on Document Engineering, DocEng 2022, San Jose, California, USA, September 20-23, 2022*, pages 7:1–7:4. ACM, 2022. 116
- [XBK⁺15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015. 89, 90

- [XLC⁺20] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. LayoutLM: Pre-training of text and layout for document image understanding. In *SIGKDD*, 2020. 1, 7, 105
- [XXL⁺21] Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *ACL/IJCNLP*, 2021. 1, 7, 105
- [YK15] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 77
- [YLR⁺20] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *ICLR*, 2020. 87
- [ZGD⁺20] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020. 3, 10, 118, 119
- [ZL17] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *ICLR*, 2017. 77, 81
- [ZTJ19] Xu Zhong, Jianbin Tang, and Antonio Jimeno-Yepes. PubLayNet: largest dataset ever for document layout analysis. *CoRR*, abs/1908.07836, 2019. 39
- [ZTY19] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. PubLayNet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE, 2019. 39
- [ZVSL18] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018. 81
- [ZZP⁺17] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017. 82

RÉSUMÉ

Cette thèse étudie l'extraction d'énoncés et de preuves mathématiques à partir d'articles scientifiques PDF en l'abordant comme un problème de classification multimodale. Elle fait partie du projet plus large TheoremKB, qui cherche à convertir la littérature scientifique en une base de connaissances complète et en libre accès d'énoncés mathématiques et de leurs preuves. La recherche exploite une gamme de techniques allant de l'apprentissage automatique traditionnel aux architectures avancées d'apprentissage profond, notamment les LSTM, les CNN, les détecteurs d'objets, les CRF, les transformeurs.

L'étude exploite une combinaison originale du texte, de caractéristiques de fontes et d'images bitmap provenant des pages PDF comme modalités de saisie distinctes. Elle propose une stratégie d'apprentissage automatique multimodale modulaire et séquentielle qui intègre un mécanisme d'attention intermodale pour produire des plongements multimodaux de paragraphes. Ces plongements sont ensuite traités via une nouvelle architecture de transformeur à fenêtre glissante multimodale qui capture les données séquentielles dans les paragraphes. Cette approche innovante ne repose pas sur le prétraitement de reconnaissance optique de caractères (OCR), sur les sources \LaTeX lors de l'inférence ou sur le pré-entraînement ad hoc avec des fonctions de perte spécialisées, ce qui la rend apte à gérer des documents multipages et des sauts de page typiques des textes mathématiques scientifiques complexes.

Les résultats indiquent une nette amélioration des performances lors du passage du traitement unimodal au traitement multimodal et de l'intégration de la modélisation de paragraphes séquentiels, soulignant l'efficacité de la méthode proposée dans le traitement de documents scientifiques complexes.

MOTS CLÉS

Extraction d'information ★ Apprentissage multimodal ★ Science des données ★ Vision par ordinateur ★ Traitement automatique de la langue

ABSTRACT

This thesis examines the extraction of mathematical statements and proofs from scholarly PDF articles by approaching it as a multimodal classification challenge. It is part of the broader TheoremKB project, which seeks to convert scientific literature into a comprehensive, open-access knowledge base of mathematical statements and their proofs. The research leverages a range of techniques from traditional machine learning to advanced deep learning architectures, including LSTMs, CNNs, Object detectors, CRFs, transformers, etc.

The study utilizes a novel combination of text, font characteristics, and bitmap images from PDF pages as separate input modalities. It proposes a modular, sequential, multimodal machine learning strategy incorporating a cross-modal attention mechanism to produce multimodal paragraph embeddings. These embeddings are processed through a novel multimodal sliding window transformer architecture that captures sequential data across paragraphs. This innovative approach does not rely on Optical Character Recognition (OCR) preprocessing, \LaTeX sources during inference or custom pre-training on specialized losses, making it adept at handling multi-page documents and page breaks, typically in complex scientific, mathematical texts.

The findings indicate a marked performance improvement when moving from unimodal to multimodal processing and integrating sequential paragraph modelling, underscoring the effectiveness of the proposed method for handling intricate scholarly documents.

KEYWORDS

Information extraction ★ Multimodal machine learning ★ Data science ★ Computer vision ★ Natural language processing