First Steps in Building a Knowledge Base of Mathematical Results

Shrey Mishra
DI ENS, ENS,
PSL University, CNRS, Inria
Paris, France
shrey.mishra@ens.psl.eu

Yacine Brihmouche
Criteo
Paris, France
y.brihmouche@criteo.com

Théo Delemazure
LAMSADE, U. Paris-Dauphine,
PSL University, CNRS
Paris, France
theo@delemazure.fr

Antoine Gauquier
DI ENS, ENS,
PSL University, CNRS, Inria
Paris, France
antoine.gauquier@ens.psl.eu

Pierre Senellart
DI ENS, ENS,
PSL University, CNRS, Inria & IUF
Paris, France
pierre@senellart.com

Abstract

This paper explores first steps towards extracting information about theorems and proofs from scholarly documents to build a knowledge base of interlinked results. Specifically, we consider two main tasks: extractions of results and their proofs from the PDF of scientific articles; and establishing which results is used in the proof of which, across the scientific literature. We discuss the problem statement, methodologies, as well as preliminary findings employed in both phases of our approach, highlighting the challenges faced.

1 Introduction

The proliferation of academic publications, particularly in theoretical fields, presents a significant challenge in information retrieval and knowledge extraction. As of *May 2020* (which is the cut-off date we used to build our dataset), the arXiv[1] preprint repository alone hosted approximately *1.7 million academic papers* (see Figure 1), a substantial portion of which include theoretical results such as theorems, lemmas, and propositions (roughly one-third based on cursory search using keyword filtering). And arXiv constitutes a fraction of the entire collection of scientific publications (e.g., the pirate Sci-Hub Web site include 88 million academic papers, and the Crossref[2] publication database contains metadata about 158 million articles as of May 2024).

Researchers typically rely on academic search engines like Google Scholar[3] or AMiner[4] to find relevant literature. While effective for general searches, this method often falls short in mathematical fields such as mathematics or theoretical computer science, where the essence of the work is encapsulated in mathematical results (theorems, lemmas, etc.) rather than at the textual level. We discuss in this paper first steps in a project whose goal is to build a knowledge base of interlinked mathematical results, directly from the scientific literature in PDF format.

The goal of such project would be to enable researchers to perform queries at the level of individual mathematical results, such as:

- ***IR at the Theorem Level:*** For example, asking *Which variants of the vertex cover problem are solvable in polynomial time?* Traditional search engines lack the context-specific understanding to address such specialized inquiries directly.
- ***Visualization of Theoretical Dependencies:*** What does the graph of results used to prove other results look like within a given research article? A knowledge base could help visualize these connections, showing how individual results contribute to broader scientific advancements.
- ***Impact Analysis of Theoretical Foundations:*** How many published results depend on a particular theorem, and what would be the impact if this theorem were proven incorrect? Understanding the dependency network of a theorem could be crucial for assessing the robustness of a field's knowledge base.

Our methodology for constructing the knowledge graph involves a two-step process, as illustrated in Figure 2:

1. *Result Extraction:* The first step focuses on identifying and extracting theoretical results

[1] https://arxiv.org/
[2] https://www.crossref.org/
[3] https://scholar.google.com/
[4] https://www.aminer.org/

Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024), pages 165–174

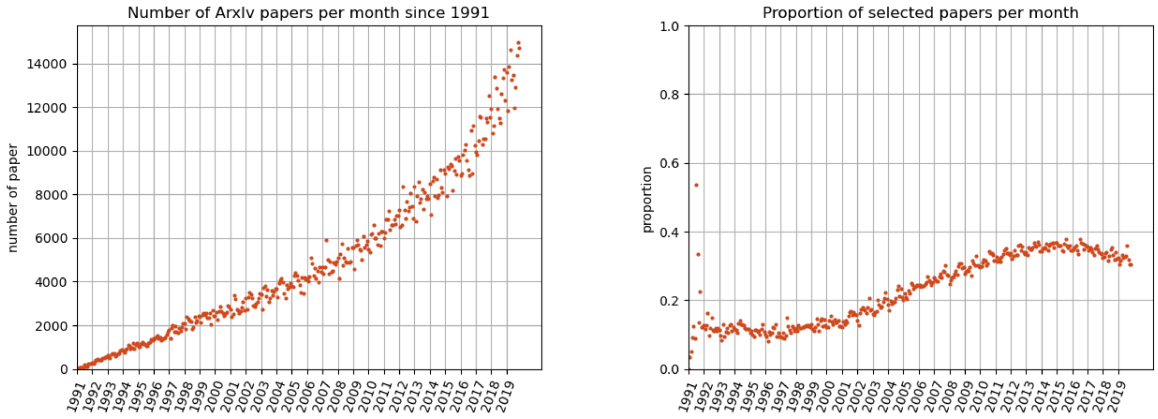

Figure 1: Evolution of the number of papers on arXiv **published** each month from its inception to May 2020 *(left panel)* and the proportion of them **containing mathematical results** *(right panel)*.

from academic papers, along with their proofs. This involves sophisticated text and data mining techniques to discern and isolate relevant mathematical expressions and their accompanying discussions within a dense and complex document layout.

2. *Result Connection:* The second step is about pinpointing the exact papers and specific results that are referenced within the proof of a given result. This step is crucial for mapping out the dependencies and relationships between different results, enhancing our understanding of how concepts in theoretical fields are interconnected. Note that there are actually two subtasks of increasing complexity: identifying which paper of a given corpus is cited in the proof of a given result; and identifying which result within that paper is used in the proof of a given result.

In this paper, we report on preliminary work on each task (and subtask), considered individually (though in reality they could benefit from a unified approach). This strategy was employed to mitigate the risk of one task's methodology negatively impacting the others, particularly given the sequential dependencies inherent in this work.

We briefly discuss related work in Section 2 then explain how we built the (partly labeled) dataset on which we operate in Section 3, along with tools used to build it. We then discuss the methodology employed for the three subtasks in Section 4 before presenting preliminary experimental results in Section 5. Code, additional details, as well as instructions to rebuild the same datasets are provided in two online repositories: `https://github.com/PierreSenellart/theoremkb` for the overall project and `https://github.com/mv96/mm_extraction` specifically for the unimodal and multimodal extraction models.

2 Related Work

Result extraction. For the extraction task, two main related works are noteworthy. Ginev et al. (Ginev and Miller, 2020) addressed the classification of mathematical statements as a 13-class problem (where classes are, e.g., proposition, example, remark). However, their evaluation was limited to the first logical paragraph, which is easier to classify, relied solely on the text modality without capturing sequential dependencies among paragraphs, and, crucially, required the availability of an HTML-ized version of the LaTeX source.

In previous work (Mishra et al., 2021) we explored deep learning-based methods, including object detection and text classification, to extract proofs and theorems. We proposed several unimodal approaches across different modalities; however, there was no clear method for comparing and evaluating these modalities. For instance, our text-based approach used fine-tuned transformers on text lines extracted by the pdfalto tool, while the vision approach utilized bitmap renderings of entire PDF pages. In Section 5, we include the line-based approach as a baseline.

In contrast, this work proposes a modular multimodal classifier that operates at the paragraph level, incorporating text, vision, and font modalities simultaneously, while also capturing sequential information to predict paragraph labels.

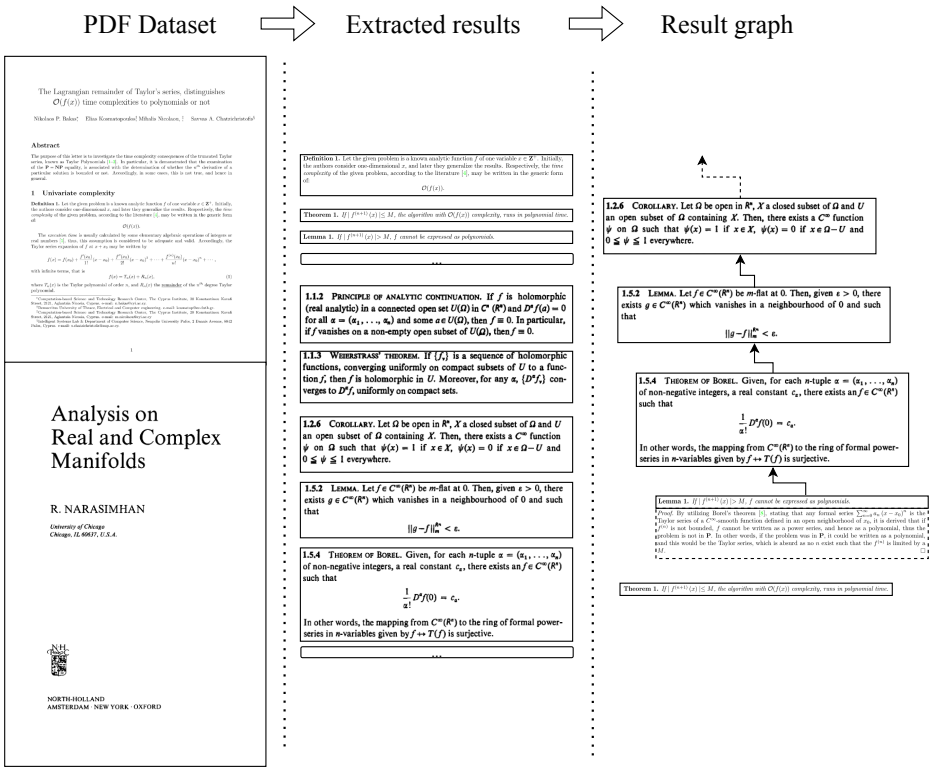

Figure 2: Two-step process for building a knowledge graph of papers: (i) result extraction; (ii) result connection

Result connection. A notable task is citation extraction and disambiguation. Cohan et al. (Cohan et al., 2019) proposed a classifier to detect citation intent based on the sentence context. It identifies three primary intents: Background, Result, and Method. They used a bidirectional LSTM with an attention layer and three independent MLPs for training. The main task was identifying citation intent, while auxiliary tasks included guessing the section title.

There are other projects whose goal is to create a structured knowledge base of mathematical results, though not by exploiting the content of PDF articles, including:

- MBASE (Kohlhase and Franke, 2001): Proposed in 2001 for formal representations of mathematical results, but development stopped in 2003.
- Logipedia[5]: A comprehensive library of formal mathematical statements for theorem provers.
- Ganesalingam and Gowers (Ganesalingam and Gowers, 2013): An automated model that writes proofs in a human style by mimicking researchers' reasoning.

3 Dataset Construction

Our approach relies on training machine learning models, which requires constructing a (sizable) labeled dataset. We explain in this section which tools we use for basic parsing of PDF documents, how we built a labeled dataset using these tools and newly developed ones, and present the characteristics of our dataset.

3.1 Information Extraction Tools

One pivotal tool in our toolkit is Grobid[6] (Lopez, 2009), a robust application also used by platforms such as Semantic Scholar[7]. Grobid is precisely engineered to extract text from scientific documents and convert raw PDFs into XML files. These files contain structured information, including segmented texts at the level of sentences and paragraphs, and further annotate critical sections such as abstracts, introductions, author names, titles, and references, along with their precise page numbers and coordinates – Grobid does not extract mathematical statements and proofs, though, which is our focus here.

Unlike many document AI models (Xu et al., 2020, 2021; Huang et al., 2022) designed for ex-

[5] http://logipedia.inria.fr/

[6] https://github.com/kermitt2/grobid

[7] https://www.semanticscholar.org

tracting information from single-page documents such as receipts or bills, Grobid leverages domain-specific knowledge to handle scientific content. It manages the layout complexities typical of multi-page scientific articles, such as page breaks and possible two-column layouts. It achieves this with remarkable efficiency, processing documents at $\approx$ 10 PDFs per second on a standard computer. We also rely on pdfalto[8], a lower-level tool by the same author as Grobid, that takes a PDF input and produce between other things a sequence of text lines with comprehensive font information for each character, such as typeface and size, along with their exact location on the page.

3.2 Automatic Labeling from LaTeX Sources

In order to construct a labeled dataset for the result extraction script, we have developed a custom tool that extracts the coordinates of mathematical statement and sources for document from which their LaTeX source is available. We insist on the fact that the availability of LaTeX source is only necessary to build a labeled dataset for training and evaluating our result extraction approach – they are not required at extraction time.

Even when LaTeX sources are available, extracting the coordinates of theorems and proofs in the resulting PDF is not trivial: indeed, there are many ways authors of a LaTeX document can use to create this form of environment, including by defining their own macros – and as TeX is a Turing-complete programming language, it is impossible to build a robust extractor just based on analyzing the source code. Instead of doing that, we developed a LaTeX package that performs the extraction, injected it in the prologue of the LaTeX document, and recompiled the entire document. The LaTeX package modifies the code of the `\newtheorem` macro (which is either the standard LaTeX one or the one provided by the `amsthm` or `ntheorem` package), of the `\declaretheorem` command (of the `thmtools` package), of the `\spnewtheorem` command (of Springer's document classes such as `llncs`) and of the `proof` environment, in order to add, whenever a LaTeX theorem or proof environment is used in the paper, a PDF hyperlink around the environment. This PDF hyperlink points to a URI that uniquely identifies the result within the document. After compilation, we extract the coordinates of these hyperlinks from the PDF document (we use pdfalto for this). Note that this does not cover all possible cases (documents written in an obsolete version of LaTeX; documents with a fully custom way of writing theorems and proofs; etc.) but this is fine: the goal is to build a labeled dataset large enough to be representative, not to have a foolproof labeling method.

Once the coordinates of every result and proof is obtained, we use a merging script to merge the output of Grobid and of pdfalto to match the paragraphs extracted by Grobid with the coordinates of text lines and the hyperlink-labeled results given by pdfalto. We also retrieve font information from pdfalto and we construct bitmap renderings of every PDF page using PyMuPDF's `fitz` module[9].

3.3 Dataset

We now report on the dataset used in experiments.

Extraction. We retrieved all papers (1.7 million) published on arXiv till May 2020 using arXiv's Bulk Data Access through Amazon S3[10]. This includes both PDF papers and, for documents that were written in LaTeX, their source as well. After filtering this massive corpus to isolate papers containing mathematical statements such as theorems or proofs, we reduced the count to *460 thousand papers*. Further refinement was necessary to exclude non-English papers, and papers that failed to compile using a contemporary PDF compiler (pdflatex from TeXlive 2019), or papers that experienced failures with any of the tools in our pipeline – Grobid, pdfalto, or bitmap rendering. This resulted in a final dataset of 197 thousand papers. Running our merging script on this dataset, we categorized each paragraph into distinct classes: ***Proof, Theorem*** (which includes other theorem-like environments such as definitions, lemmas, corollaries, remarks), ***Basic*** (neither proof nor theorem), and a marginal ***Overlap*** reject class for instances where multiple categories were erroneously grouped due to paragraph splitting bugs in Grobid. To construct our validation dataset, we randomly selected *3,682 papers* comprising approximately *0.5 million paragraphs* with their label, each enriched with font information, bitmap images, and the textual content present.

Connecting Papers and Results. For this task, no automatic labeling is possible so we need to create a smaller scale dataset. We decided to select

[8] https://github.com/kermitt2/grobid

[9] https://pymupdf.readthedocs.io/

[10] https://info.arxiv.org/help/bulk_data_s3.html

a topically consistent subset of papers, focusing on the domain of computational complexity, from 2010 to 2020, obtaining roughly *6 000 scientific articles* under the "Computer Science - Computational Complexity" (cs.CC) category of arXiv. This category was chosen due to the high likelihood of inter-citations among papers within this specific domain. After applying our extraction script we obtained *4 705 papers* that were compilable and did not trigger errors with the deployed tools. Of these, *3 711 papers* contained results that referenced another paper within the same dataset, which we manually labeled.

4 Methodology

In this section, we delve into the specific methodologies adopted for each of the three subtasks outlined in our study: the extraction task and the two connecting tasks – results to paper and results to results.

4.1 Extraction of Results

Intuitively, the following features can be used to isolate results and proofs in PDF papers. *Visual Features:* We identify vital visual indicators such as bold or italic text and the presence of a QED symbol, which typically denotes the end of a proof. *Font-Level Features:* Recognizing specific fonts used at the beginning of paragraphs, as seen in papers formatted in certain styles which consistently uses distinctive fonts for theorem declarations. *Language Information:* Analyzing the textual content within the spatial bounds of a paragraph to discern its nature based on linguistic patterns. *Structural Information:* Utilizing page numbers, relative positions, and the context of neighbouring paragraphs to improve prediction accuracy.

To effectively process these diverse data modalities, we deploy three different classifiers:

Font Model: This model uses an **LSTM** (Hochreiter and Schmidhuber, 1997) to process sequences of font data extracted by pdfalto, capturing dependencies in font usage throughout the document.

Vision-Based Model: We implement a CNN, specifically an **EfficientNetV2** (Tan and Le, 2021), to analyze visual clues. This model processes bitmap images of paragraphs, focusing on initial word formatting and the presence of end-of-proof symbols.

Language Model: **A pretrained RoBERTa** (Liu et al., 2019) language model is used, where the CLS token from the last layer helps identify significant text sections. The model operates on text extracted and structured by Grobid at the paragraph level.

To synthesize insights from these distinctive models trained on different modalities, we employ a **GMU-based mechanism**. This approach, inspired by Gated Multimodal Units (Arevalo et al., 2020), facilitates the integration of features from different modalities. Each model's output is frozen to prevent updates during this phase, ensuring stability in the embeddings they produce.

The features extracted from the Multimodal GMU are then used as input for a **CRF model** that captures sequential dependencies between blocks. Numerous studies (Patel and Caragea, 2019), (Al-Zaidy et al., 2019), (Wei et al., 2022) have demonstrated the efficiency of employing a CRF-based model on top of features extracted by a deep learning approach to model and extract important keyphrases, particularly within scholarly documents. CRF models have also been used to model sequential dependencies and clustering information over features extracted from CNN models on scientific document pages (Li et al., 2018). This model uses sequential information from the previous paragraphs to determine the paragraph's label across the entire document. It incorporates contextual features such as page numbers and coordinate information along with information about page breaks. This approach allows the model to understand and classify each paragraph's role within the broader structure of the document.

This methodology leverages the strengths of each modality and ensures that the extracted information aligns with how a human would typically look at the problem. By combining different techniques across different modalities, we aim to classify and link mathematical results robustly across various scientific literature.

4.2 Linking Results to Papers

We employed Grobid, a machine-learning tool designed to convert PDFs of scientific papers into structured XML files. This conversion crucially extracts bibliographic data (including titles, authors' names, and publication years of cited articles) from the article, with references to each bibliographic entry. We also integrated data from the S2ORC

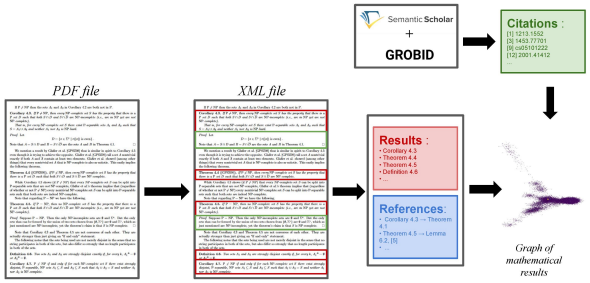

Figure 3: **Flow of the algorithm**. We convert a PDF file into an XML file with identified theorems and proofs following our extraction methodology. Finally, we associate references with papers in the corpus to build a graph of mathematical results

dataset (Lo et al., 2020) of Semantic Scholar, which maps paper references to arXiv IDs.

To systematically map connections from papers to papers within our dataset, we developed a high-level graph extraction algorithm summarized in Figure 3. This algorithm efficiently identifies and represents the relationships between papers based on their citations, creating a navigable and interactive graph.

Our graph extraction algorithm is designed to map the citations within and across papers, creating a network that can be navigated and analyzed. We proceed as follows. First, we extract theorems and proofs from PDF articles as previously described.[11] Second, within every identified proof, we search for explicit references to formal results (potentially from other papers), with simple regular expression patterns capturing strings such as "Theorem 1", "Lemma 2.54.2", "Problem A.1", "Proposition E.7.1.3.1". We also determine whether such a referenced link is internal (within the same paper) or external (pointing to another document), looking for references to bibliographic entries in the immediate context (such as "Theorem 1 of [2]" or "[27], Proposition 26,"). We primarily focus on these external references for building our citation graph, ignoring internal citations (i.e., references within the same paper), and retrieving the corresponding bibliographic entry. In cases where multiple results are referenced together, such as "Theorems 1, 5 from [2]", we split these into separate citations for clarity. Additionally, we filter out false bibliographic links by checking for specific keywords that indicate non-referential usage. We

[11] To evaluate independently both phases of our extract–connect approach, we can also start from the dataset with ground truth labels for the position of theorems and proofs.

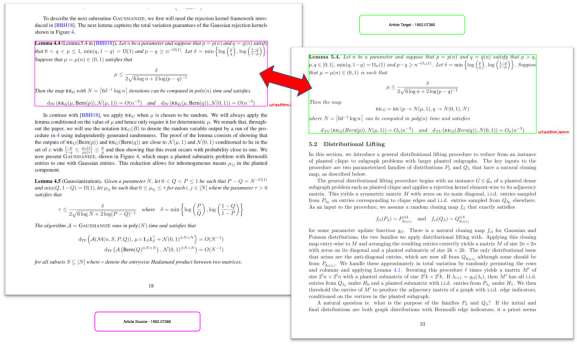

Figure 4: A clear, obvious link where the author did specify the connecting result.

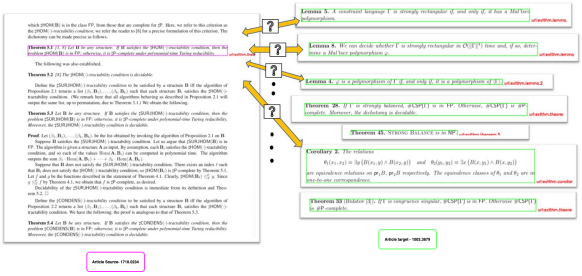

Figure 5: An ambiguous link where the author did not specify the connecting result

finally merged the Grobid-produced bibliographic entries with the S2ORC ones, allowing us to retrieve the arXiv ID of each bibliographic reference if it exists. Note that we encountered some occasional mismatches between our arXiv dataset and the S2ORC one, coming from the fact that an arXiv paper may have multiple versions, only the most recent of which is available in the bulk data dump. This means the tag used for the bibliographic entry may not match across these two datasets but we resolved inconsistencies by looking for a match in the whole bibliographic entry itself.

4.3 Linking Results to Results

A crucial step in constructing a knowledge graph of mathematical results is accurately linking specific results between papers. Thus, when citing Theorem 42 of reference [X] in the proof of a result, when restating Theorem 42 of reference [X] with another name in another paper (see Figure 4 for an example), or, simply, when using reference [X] in the proof of a result or when restating it without an explicit reference to a specific result, it is important to determine which result from the paper referenced by [X] is actually mentioned (see Figure 5 for an example of this case).

To address this linking challenge (determining

which result from the paper referenced by [X] is cited in the current paper), we employ a dual-representation strategy to establish the most likely matches between cited results in the source paper and possible references in the target paper. This involves the following steps:

1. **Structural Representation via TF-IDF Vectorization:** We vectorize paragraph content using a TF-IDF word vectorizer (Jones, 2004). This technique transforms the textual content into a high-dimensional vector space where each dimension corresponds to a term's frequency-inverse document frequency score. The resulting vectors capture the structural aspects of the text, such as the usage of specific terms and syntactic patterns often indicative of mathematical content.

2. **Semantic Representation with DistilBERT:** Parallel to the structural approach, we process the text using a ***pre-trained DistilBERT model*** (Sanh et al., 2019). This smaller, more efficient version of *BERT* is fine-tuned on a generic classification task to categorize text as proof, theorem, or introductory text. We use the **[CLS]** token embedding from the last layer of DistilBERT as the comprehensive vector representation of each paragraph. This semantic vector captures deeper linguistic and contextual nuances, identifying conceptual similarities between paragraphs that the TF-IDF approach might miss.

Both vectorization methods are applied in a zero-shot learning context, meaning no additional fine-tuning is done for this specific task. There is no training on specialized losses such as contrastive loss, which could otherwise enhance the model's ability to distinguish connected text segments. We compute the cosine similarity between the source paragraph vector from the source paper and all paragraph vectors from the target paper. The pair of paragraphs (one from the source and one from the target) with the highest cosine similarity is considered the likely match.

5 Experimental results

This section presents experimental results derived from implementing our methodologies across three specific subtasks: extraction, connecting papers, and linking specific results. We stress that results from the latter two tasks are preliminary at this stage.

5.1 Extraction

The extraction task involved training multiple machine learning classifiers on different modalities to recognize and categorize paragraph-level information content from academic papers. Table 1 summarizes the accuracy achieved by each classifier with or without sequential paragraph information, compared to two baselines: a dummy classifier that always predicts the most frequent *Basic* class, and the textual model, line-based, approach from (Mishra et al., 2021), reevaluated on the same dataset. We report standard accuracy, as well as mean-F_1 over the three classes *Basic*, *Theorem*, *Proof*.

The best-performing approach is the multimodal approach, which captures sequential information using the CRF model. However, the gains are modest with respect to the pure text-only approach augmented with the CRF model. Extra experiments are described in (Mishra et al., 2023).

5.2 Connecting results to papers

For connecting papers, we visualized the graph of linked results across the **entire arXiv corpus**. Figure 6 illustrates the interconnectedness of various papers through citations of famous theorems and lemmas. The visualization underscores the dense network of academic work, reinforcing the importance of effective result linking to navigate scholarly literature efficiently. For more details, see (Delemazure, 2020).

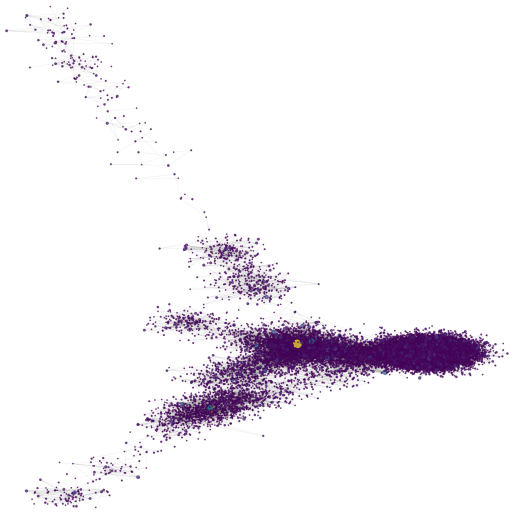

Figure 6: Largest connected component of the aggregated ArXiv graph generated with the networkx Python library, using a spectral layout.

Modality	Model chosen	Seq. approach	#Batches	#Params (M)	Accuracy (%)	Mean F_1 (%)
Dummy	always predict *Basic*	—	—	—	59.41	24.85
Line-based (Mishra et al., 2021)	Bert (fine-tuned)	—	—	110	57.31	55.71
Font	LSTM 128 cells	— CRF	11 11+1	2 2	64.93 71.50	45.48 64.51
Vision	EfficientNetV2M_avg	— CRF	9 9+1	53 53	69.44 74.63	60.33 70.82
Text	Pretrained RoBERTa-like	— CRF	20 20+1	124 124	76.45 83.10	72.33 80.99
Multimodal	GMU	— CRF	10 10+1	185 185	76.86 **84.19**	73.87 **82.91**

Table 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).

5.3 Linking results

In the final subtask of linking specific results between papers, we evaluated the effectiveness of two vectorization-based approaches: TF-IDF and DistilBERT embeddings on a small dataset of 45 theorems with manually labeled matches in other papers. The outcomes of these approaches are summarized below:

TF-IDF Vectors: This approach successfully matched **17/45 (38%)** results across different papers. TF-IDF vectors, which emphasize term frequency within the document context, proved particularly effective in matching results that use similar terminologies.

DistilBERT: The DistilBERT model achieved **7/45 (15%)** perfect matches. Despite its lower performance than TF-IDF in this zero-shot setting, DistilBERT captured deeper semantic relationships that were not as transparent through lexical similarity alone.

For more details, see (Brihmouche, 2022).

6 Conclusions

This paper presents a comprehensive framework for constructing a knowledge base of mathematical results. It aims to transcend the limitations of traditional search engines through an enhanced, result-oriented navigation system within academic literature. **Through preliminary experiments on distinct subtasks, we have demonstrated the feasibility of our proposed solution, establishing a proof of concept for a robust and scalable knowledge management system**.

In data extraction, our experiments have highlighted the efficacy of a multimodal approach, leveraging a GMU architecture followed by Sequential CRF modelling. This approach proved adept at capturing sequential and contextual nuances in text, offering a slight advantage over simpler, text-only methods. The ability of this architecture to integrate various data modalities into a cohesive model illustrates its potential for comprehensive information extraction from complex scientific texts, setting a high standard for subsequent refinement and application.

The construction of a citation graph detailing the connections between papers through cited results has unveiled a rich tapestry of scholarly communication. Notably, the longest path in this citation graph extends to 13 links, underscoring the depth and complexity of academic discourse. However, the aggregated nature of this graph means we must be cautious in interpreting these paths, as they do not guarantee a direct continuation of the topic from one paper to another.

Our examination of methodologies for linking specific results across papers has underscored the importance of precise terminology in the academic discourse of mathematics. The TF-IDF vectorization approach outperformed the semantic capabilities of the DistilBERT model, suggesting that the reuse of exact terms and phrases is crucial for accurately connecting related results, though our experiments are still preliminary.

Moving forward, we consider several avenues for exploration and development. Enhanced machine learning models, specifically tailored to the nuances of mathematical language and structured problem-solving in academia, could drastically reduce the dependency on manual labelling. Ad-

ditionally, expanding the knowledge base to include more diverse fields of study and integrating more advanced semantic understanding tools will enhance the breadth and depth of the knowledge graph, providing unparalleled access to interconnected scientific knowledge.

Acknowledgments

This work was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). This work was also made possible through HPC resources of IDRIS granted under the allocation 2020-AD011012097 made by GENCI (Jean Zay supercomputer).

References

Rabah A. Al-Zaidy, Cornelia Caragea, and C. Lee Giles. 2019. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2551–2557. ACM.

John Arevalo, Thamar Solorio, Manuel Montes-y Gómez, and Fabio A González. 2020. Gated multimodal networks. *Neural Computing and Applications*, 32.

Yacine Brihmouche. 2022. TheoremKB : une base de connaissance des résultats mathématiques. Master's thesis, Paris IX Dauphine, September.

Arman Cohan, Waleed Ammar, Madeleine van Zuylen, and Field Cady. 2019. Structural scaffolds for citation intent classification in scientific publications. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3586–3596. Association for Computational Linguistics.

Theo Delemazure. 2020. A Knowledge Base of Mathematical Results. Master's thesis, Ecole Normale Supérieure (ENS), September.

M. Ganesalingam and W. T. Gowers. 2013. A fully automatic problem solver with human-style output. *CoRR*, abs/1309.4501.

Deyan Ginev and Bruce R. Miller. 2020. Scientific statement classification over arxiv.org. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 1219–1226. European Language Resources Association.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8).

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. LayoutLMv3: Pre-training for document ai with unified text and image masking. In *ACM MM*.

Karen Spärck Jones. 2004. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 60(5):493–502.

Michael Kohlhase and Andreas Franke. 2001. Mbase: Representing knowledge and context for the integration of mathematical software systems. *J. Symb. Comput.*, 32(4):365–402.

Xiao-Hui Li, Fei Yin, and Cheng-Lin Liu. 2018. Page object detection from pdf document images by deep structured prediction and supervised clustering. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 3627–3632.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Patrice Lopez. 2009. Grobid: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In *ECDL*, volume 5714, pages 473–474.

Shrey Mishra, Antoine Gauquier, and Pierre Senellart. 2023. Multimodal machine learning for extraction of theorems and proofs in the scientific literature. *CoRR*, abs/2307.09047.

Shrey Mishra, Lucas Pluvinage, and Pierre Senellart. 2021. Towards extraction of theorems and proofs in scholarly articles. In *DocEng '21: ACM Symposium on Document Engineering 2021, Limerick, Ireland, August 24-27, 2021*, pages 25:1–25:4. ACM.

Krutarth Patel and Cornelia Caragea. 2019. Exploring word embeddings in crf-based keyphrase extraction from research papers. In *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.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv:1910.01108*.

Mingxing Tan and Quoc Le. 2021. EfficientNetv2: Smaller models and faster training. In *ICML*.

Xin Wei, Lamia Salsabil, and Jian Wu. 2022. Theory entity extraction for social and behavioral sciences papers using distant supervision. In *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.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. 2021. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *ACL/IJCNLP*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. LayoutLM: Pre-training of text and layout for document image understanding. In *SIGKDD*.