Theorem View: A framework to Extract Theorem like environments from raw PDFs

Shrey Mishra[1 [0000−0000−0000−0000]], Neil Sharma[2 [0000−0000−0000−0000]], Antoine Gauquier[1 [0000−0000−0000−0000]], and Pierre Senellart[1,3 [0000−0000−0000−0000]]

[1] DI ENS, ENS, CNRS, PSL University, Inria, Paris, France
shrey.mishra@ens.psl.eu, antoine.gauquier@ens.psl.eu, pierre@senellart.com
[2] Malaviya National Institute of Technology
neil.sharma3000@gmail.com
[3] Institut Universitaire de France (IUF)

Abstract. This paper presents TheoremView, a novel modular multimodal framework designed to extract proofs and theorems from scientific papers in their raw PDF format, without requiring access to LaTeX source files. Our approach leverages three distinct modalities—font, text, and vision—to accurately identify and classify proof and theorem-like environments. We incorporate a sequential component to capture long-term dependencies, including page breaks and layout information. TheoremView employs a combination of specialized models trained to recognize and extract these critical elements from academic documents. By eliminating the need for OCR preprocessing, our method significantly reduces computational overhead, making it feasible for real-time applications. This innovative approach offers a robust solution for automated theorem extraction, potentially enhancing accessibility and analysis of mathematical content in scientific literature.

Keywords: Theorem extraction · Multimodal learning · Document analysis · Machine learning · Natural language processing

1 Introduction

1.1 Motivation for Theorem Extraction

In contemporary scientific research, articles are primarily published as PDFs, and many search engines index entire papers instead of specific scientific results. This paper contributes to TheoremKB [**?**], a project focused on building a knowledge base of mathematical results across different fields of science. The objective is to improve the accessibility of relevant information for researchers, allowing for more effective retrieval and utilization of scientific knowledge.

Having such a knowledge base can significantly impact the way researchers find information. Some advantages of this system include:

1. **Enhanced Accessibility**: TheoremKB streamlines the retrieval of specific proofs and theorems, allowing for quick access to targeted mathematical results, in contrast to traditional search engines that index full-text papers.
2. **Facilitated Knowledge Discovery**: This knowledge base assists researchers in uncovering connections between disparate mathematical results and their applications, thereby enhancing the exploration of specific results, such as NP-hardness in relation to the vertex cover problem.
3. **Identification of Theorem Interdependencies**: TheoremKB helps determine which theorems are used in the proofs of others, which is essential for assessing the impact of errors in foundational results.
4. **Support for Automated Reasoning**: The knowledge base provides a foundation for developing AI systems capable of automated theorem proving, promoting innovative approaches to mathematical problem-solving.

1.2 Prior work on theorem/proof Extraction

Previous attempts to address this task include the work presented in [**?**], which focused on initial explorations of extraction from PDFs framed as object detection and text classification problems. This approach utilized font visuals and text modalities but operated only at the textline level. Subsequent research, such as [**?**], refined the methodology by incorporating contextual information surrounding paragraphs and employing multimodal systems to unify the extraction model.

The TheoremView framework offers a user interface to visualize the results extracted by various models in an end-to-end system that directly takes PDFs as input and displays the extracted results. It is designed modularly, allowing users to select which model to utilize for extraction, thereby leveraging different modalities that highlight the strengths and weaknesses of each approach. This flexibility enables users to run models on low-compute hardware, such as systems without GPU instances, for inference. The primary objective of this paper is to present an easy-to-use interface that facilitates preprocessing and inference in a modular manner.

2 Methodology

We propose a modular approach to extract raw information from PDFs. We utilize Grobid and pdfalto to convert the documents into valid XML formats. The XML generated by Grobid organizes the content into paragraphs, while the XML from pdfalto provides details segmented into text lines along with associated font information. We then employ a merging script to correlate the font information with each paragraph extracted by Grobid. This process yields a CSV file structured by paragraphs, where each row includes the spatial location of the paragraph on the page (indicating page number and vertical and horizontal coordinates), the textual content extracted from Grobid, and the font information used within those paragraphs.

Importantly, we do not rely on any OCR preprocessing; instead, we leverage PDFALTO and Grobid, which are specialized tools that efficiently extract information from scientific articles. It is crucial to note that while we do not use LaTeX sources during inference, they are utilized solely for generating ground truths during model training.

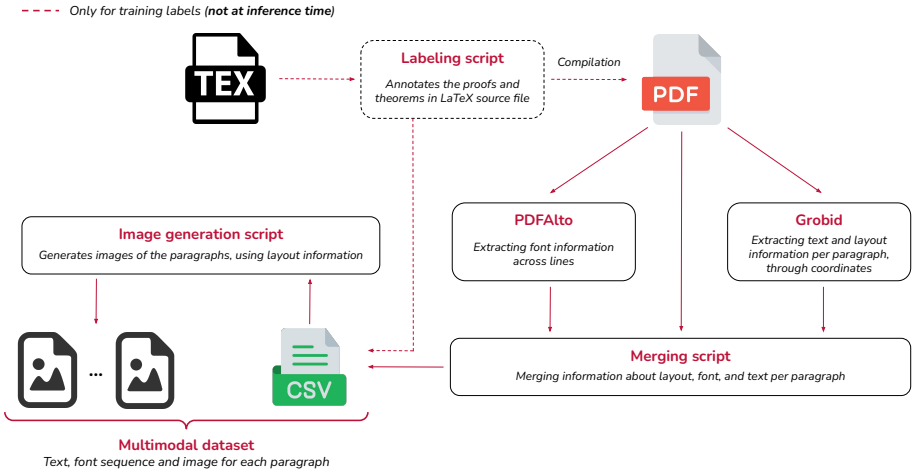


Fig. 1: Data pipeline for extracting and processing information from PDFs.

Once the information is stored in CSV format, we process the font information using an LSTM model, where each font is encoded as a unique token to train the network. Simultaneously, we utilize the bitmap image rendering of each paragraph to train an EfficientNetV2 model. Additionally, we employ a language model, pretrained from scratch in our case, to make predictions based on the text modality. Subsequently, we integrate all three trained models into a multimodal architecture, freezing their weights and adding additional layers to capture intermodality interactions through mechanisms like Gated Multimodal Units (GMU) or cross-modality attention.

With a set of base features extracted from either the unimodal or multimodal approaches, we generate features for all paragraphs within the PDF. This process incorporates normalized page information, normalized coordinate data for each paragraph, and paragraph embeddings derived from the raw features just before the softmax layer. To capture sequential information across multiple paragraphs, we train a Conditional Random Field (CRF) or transformer layer atop the extracted features. The goal is to utilize relative information to contextualize each paragraph and accurately determine its label. Our model categorizes paragraphs into four major classes: (1) Proof-like, (2) Theorem-like, (3) Basic (neither proof nor theorem), and (4) an Overlap reject class that arises from preprocessing discrepancies.

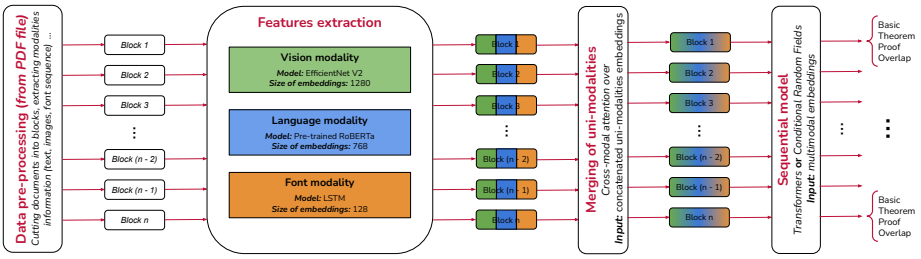


Fig. 2: Multimodal Framework for Theorem Extraction from PDFs

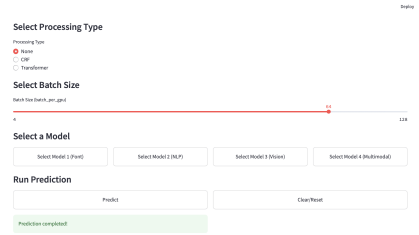
3 Demonstation scenario

The UI of the demo is organized into several segments, each serving a specific function:

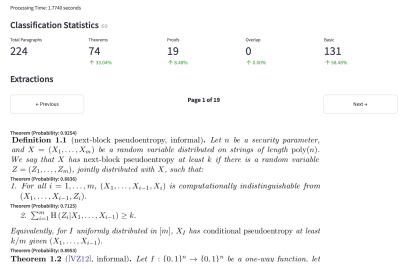
1. **Upload and Process**: Users can upload a PDF for processing or select from a list of previously cached examples. The system processes the PDF by applying Grobid and pdfalto, converting each page into a bitmap image, and merging the XML outputs from pdfalto and Grobid to generate a preprocessed `data.csv` file.
2. **Predict and Preview**: Users can choose which base model to apply, either unimodal or multimodal, along with sequential processing of paragraphs using a Conditional Random Field (CRF), transformer, or no sequential processing at all, allowing for a total of 12 possible combinations. Additionally, users have the option to preview or download the results (see Figure 3 for visualization of results).
3. **Buttons**: This section includes a visual representation of the buttons available in the UI.
4. **Summary and Statistics**: This section provides a breakdown of the inference time for the current run, as well as for any previous runs that have been cached, enabling comparative analysis (see UI 6 diagram).

References

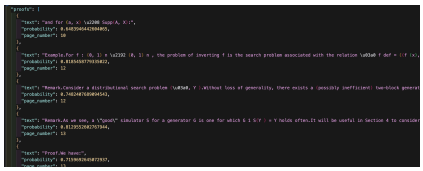
1. Author, F.: Article title. Journal **2**(5), 99–110 (2016)
2. Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016). https://doi.org/10.10007/1234567890
3. Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999)
4. Author, A.-B.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010)
5. LNCS Homepage, http://www.springer.com/lncs, last accessed 2023/10/25



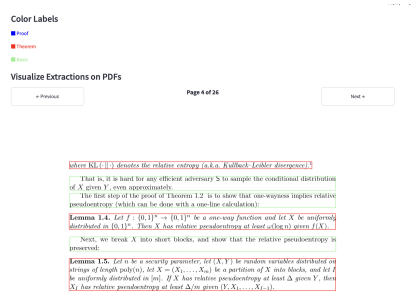
(a) UI Buttons for various actions in the demo.



(b) Extracted Information in PNG format.



(c) Extracted Information in JSON format.



(d) Visualizations overlaid on the PDF.

Fig. 3: Various visual elements in the demo.