Modular Multimodal Machine Learning for Extraction of Theorems and Proofs in Long Scientific Documents (Extended Version)

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

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

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

Abstract

We address the extraction of mathematical statements and their proofs from scholarly PDF articles as a multimodal classification problem, utilizing text, font features, and bitmap image renderings of PDFs as distinct modalities. We propose a modular sequential multimodal machine learning approach specifically designed for extracting theorem-like environments and proofs. This is based on a cross-modal attention mechanism to generate multimodal paragraph embeddings, which are then fed into our novel multimodal sliding window transformer architecture to capture sequential information across paragraphs. Our document AI methodology stands out as it eliminates the need for OCR preprocessing, LaTeX sources during inference, or custom pre-training on specialized losses to understand cross-modality relationships. Unlike many conventional approaches that operate at a single-page level, ours can be directly applied to multi-page PDFs and seamlessly handles the page breaks often found in lengthy scientific mathematical documents. Our approach demonstrates performance improvements obtained by transitioning from unimodality to multimodality, and finally by incorporating sequential modeling over paragraphs.

CCS Concepts

• **Information systems** → **Information extraction**.

1 Introduction

Context. Scholarly articles in mathematical fields typically include theorems (and other theorem-like environments) along with their proofs. This paper builds upon our previous work [31], which aimed to transform scientific literature from a collection of PDF articles into an open knowledge base (KB) centered around theorems.

The objective of [31] was to enable new ways of exploring mathematical results, such as searching for all theorems that depend on a specific result or identifying all proofs that include a particular feature.

For example, such a knowledge base would allow the following:

(1) **Navigating through the scientific literature:** Currently, the only way to navigate through the scientific literature is through search engines such as Google or Google Scholar that index the full-text of papers, or by navigating through citation links. These approaches do not allow indexing of individual mathematical results, which is the main object of interest of mathematicians and theoretical computer scientists. With a KB of scientific results, one would be able to find, e.g., all NP-hardness results involving the vertex cover problem (and not just all papers that contain both the terms "NP-hard" and "vertex cover").

(2) **Identifying the impact of errors in theorems:** Another useful application of such a knowledge base is to determine which theorems are used in the proof of another theorem. This would be of tremendous use, for instance, to determine which results become invalidated or need to be revisited when one of the theorems they depend on is shown to be false.

In [31], we outlined the comprehensive scope of our project and presented preliminary evaluations on two fundamental subtasks:

(i) extraction of information pertaining to proofs and theorems;
(ii) linkage of mathematical results across various papers.

In this paper, we concentrate primarily on the extraction aspect of the pipeline introduced in [31]. We conduct an in-depth exploration of diverse multimodal methodologies and assess the impact of modeling long-term paragraph sequences. This is particularly advantageous for the identification of mathematical results, as it utilizes the contextual information surrounding the paragraphs covering length proofs.

Problem definition. As a first step towards this ambitious goal of building a knowledge base of mathematical results, it is necessary to develop information extraction methods that automatically identify theorem-like environments and proofs in PDF scientific articles.

A human being would typically be able to perform this task by relying on the formatting of the text, on specific keywords identifying the environments, and on other visual clues: including keywords such as "Theorem" or "Proof" in bold or italics, the fact that an entire block of text might be in italics, the comparatively high proportion of mathematical characters, the presence of a QED symbol at the end of a proof, etc. However, precise formatting depends on document formats; a classifier that would only use such kinds of hard-coded rules does not generalize well for arbitrary formats, or for proofs that span multiple paragraphs.

To clarify, in the whole of this paper we use *theorem* in the same sense as it is used in LaTeX (say, by the `\newtheorem` command): a theorem-like environment is a structured statement, possibly numbered, formatted in a specific way and used to represent a formal (usually mathematical) statement: it can be a theorem, a lemma, a proposition, etc., but also a definition, a formal remark or an example. By *theorem* we mean any statement of this kind. By *proof* we mean what would typically be rendered in LaTeX in a `proof` environment: a proof or proof sketch of a result.

We propose to approach the theorem–proof identification problem by designing an approach based on multimodal machine learning that classifies each paragraph of an article into *basic*, *theorem*, and *proof* labels, based on the scientific language, on typographical information, and on visual rendering of PDF documents. Additionally, we take into account information about the *sequence* of paragraph blocks, normalised spatial coordinates and page numbers along with page breaks, to exploit the fact that the label of a paragraph heavily relies on that of the preceding (and possibly following) ones.

Methodology and contributions. To design a multimodal approach to the theorem–proof identification problem, we take inspiration from how a human being would solve the task, i.e., with the help of:

(1) Understanding of the scientific vocabulary and how mathematical writing is organized: it might be possible to recognize a proof or a theorem by the presence of phrases as "We conclude by" or "Assume … Then …".
(2) Visual features such as symbols and the use of bold or italic fonts: some document classes, for instance, format the content of a theorem all in italics and end all proofs with a QED symbol.
(3) Use of different font types and sizes in order within paragraphs: starting a paragraph with a word in bold or in italics.
(4) Sequential organization of blocks within a document: For example, if we know the label of both previous and next paragraphs are *proof*, it is likely that of the current paragraph is also *proof* (or possibly a *theorem*; *basic* is unlikely); this is even more relevant, if vertical spacing between these blocks is small.

This suggests, respectively, the use of a language model to capture text-level information; the use of a computer-vision approach to capture visual features; the use of styling information to capture font-based information; and the use of a sequential model to capture information from block sequences. In addition, we want to be able to combine all these features in a unified multimodal approach.

We provide the following contributions in this paper, summarized in Figure 1: (i) Three unimodal (vision, text, font information) models for the theorem–proof identification problem relying on modern machine learning techniques (CNNs, transformers, LSTMs) with a focus on reasonably efficient models as opposed to very large ones; note that the text modality approach relies on pretraining a language model specific to our corpus, which may have applications beyond our task. (ii) A multimodal late fusion model that combines the features of all three modalities. (iii) A block sequential approach, based on a transformer model, that can be used to improve the performance of any unimodal and multimodal model by capturing dependencies between blocks. (iv) An experimental evaluation on a dataset of roughly 200k English-language papers from arXiv, with a separate validation dataset of 3.5k papers (amounting to 529k paragraph blocks).

Outline. After discussing related work in Section 2, we present in Section 3 the three unimodal models. We then discuss in Section 4 how to combine them into a multimodal model, and how to add support for information about block sequences. We further provide a description of our dataset in Section 5. Experimental results on all unimodal and multimodal models is presented in in Section 6. For brevity, additional materials are provided as supplementary content. Detailed information, including design choices, architecture diagrams, confusion matrices for all classes, data pipeline diagrams, explainability, large-scale societal impact, and other critical aspects of the project, are comprehensively discussed in the PhD thesis of the first author [30]. The code, data, and models supporting this paper are accessible at https://github.com/mv96/mm_extraction.

2 Related Work

We discuss now related work about extraction of theorems and proofs from the scientific literature, and more broadly about document datasets. We shall discuss further related work relevant to unimodal or multimodal approaches when discussing individual models.

Extraction of theorems and proofs. The theorem–proof extraction problem has received little interest in past research, though we now discuss two highly relevant works [10, 32].

Ginev and Miller [10] proposed the task of identifying proofs and theorem-like environments from arXiv[1] articles using their HTML rendering via LaTeXML. Their approach involves detecting mathematical statements (along with other regions such as abstract and acknowledgements), introduced as a 50-class classification problem. They show that there is some link in the contextual information among paragraphs, which is then exploited by the textual modality over a BiLSTM-based encoder/decoder approach. This approach has two major limitations, which make it unsuitable for our needs: (i) Their approach does not operate on raw PDFs but on HTML renderings, which makes it only applicable when LaTeX source code is available.[2] (ii) Their approach is only evaluated on the first logical paragraph within a marked-up environment belonging to the label set (e.g., only the first paragraph of every proof), which makes the task much simpler, since the first word is highly indicative of the label in most cases; in contrast, we aim at differentiating such environments from regular text, and we aim at classifying all paragraphs within an environment, not just the first one. In addition, note that accessing the dataset of [10] requires signing an NDA[3] whose terms prevent free use for research.

In our prior work [32], we built a proof of concept system evaluating various unimodal approaches based on different evaluation metrics using NLP, computer vision, and a mix of heuristics (detection of specific keywords) and font-based information to identify mathematical regions of interest. The problem was posed as a 3-class classification problem operated on text *lines* extracted from raw PDFs obtained using pdfalto[4]. This work had some important limitations: (i) Text lines do not usually contain entire sentences and offer little context, which means they are hard to classify. (ii) The computer vision approach was framed as an object detection challenge, utilizing an Intersection Over Union (IOU) based metric,

[1] https://arxiv.org/
[2] Note that in such settings, extracting theorems and proofs from the source, as we do in Section 5 to build a labeled dataset, seems a better alternative.
[3] https://sigmathling.kwarc.info/resources/arxmliv-statements-082018/
[4] https://github.com/kermitt2/pdfalto

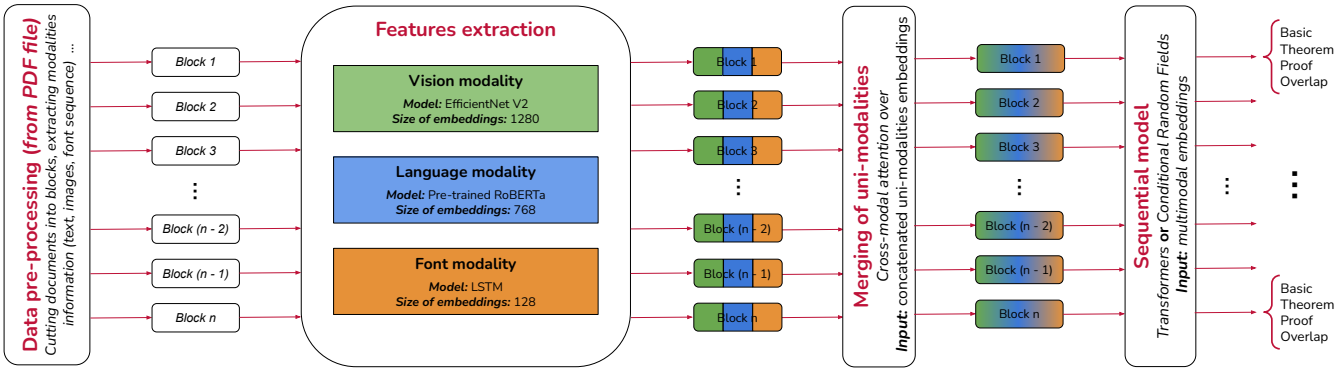

Figure 1: Overall model Inference pipeline

poorly suited for identifying text blocks, as alterations in the threshold can affect the detection score. Additionally, this discrepancy complicates the integration of the computer vision approach with other modalities that function at the text line level. (iii) The third modality, focusing on heuristics and font features, primarily utilized hand-crafted characteristics, including checks for whether the first word is bold or italic. (iv) Moreover, Mishra et al. [32] do not present a consistent method for comparing the three modalities, as each modality relies on a distinct segment of information. In our study, we strive to establish a standardized approach for evaluating the performance of various modalities, as well as combining them together. This study builds upon [32], addressing and overcoming the four primary limitations identified therein. We introduce a modular, multimodal framework that facilitates a consistent methodology for the comparison and integration of the three modalities, eliminating the necessity for LaTeX source files during inference.

Document datasets. In this research, we utilize Grobid [27] to extract textual content and bounding boxes from paragraph-rendered bitmap images, a foundational step for training our models. This work shares similarities with the objectives outlined in the Publaynet study [47], particularly in our focus on identifying "Proofs" and "Theorems" within scholarly articles. Unlike Publaynet, which categorizes document sections into Text, Tables, Figures, etc., at the document level, our study extends to the analysis of academic writings, leveraging LaTeX sources from arXiv submissions for ground truth generation, in contrast to Publaynet's use of the National Library of Medicine (NLM)[5] schema for journal articles.

At its core, Grobid covers a broad array of document segments, including lists, figures, titles, and bibliographic entries, akin to the scope of Publaynet [47]. It distinguishes itself by semantically parsing texts into sentences and paragraphs and accurately identifying block coordinates, thus supporting a text-based modality in our analysis. Notably, Grobid preserves mathematical content within textual segments, a feature not prioritized by Publaynet, which omits certain XML tree nodes like tex-math and disp formula (as stated by the PMCOA XML on Page 2 of [47]). This distinction underscores the relevance of our approach to the specific requirements of our study and its broader goals. While Publaynet prioritizes visual classification among diverse labels such as Text, Figures, and Tables, our analysis, as evidenced in Table 1, highlights the indispensable role of textual modalities in distinguishing proofs across paragraphs, underscoring the multimodal nature of our challenge where textual analysis is paramount.

The debut of Docbank [24] marks a notable advancement beyond Publaynet [47], offering a comprehensive dataset of 500K document images tailored for training and testing applications. Unlike Publaynet's emphasis on the medical field, Docbank encompasses a wider array of academic areas, including Physics, Mathematics, and Computer Science. This diversity introduces a rich variety of mathematical formulas to the dataset. Docbank's distinctive feature is its dual-level annotations – both token and segment – rendering it highly applicable for a broad spectrum of tasks in computer vision and natural language processing. Despite its potential for aiding proof identification projects, Docbank's broad annotation scope, encompassing author names, abstracts, titles, equations, and paragraphs, may dilute its applicability for our focused research on identifying proofs and theorems within texts. A limitation arises from Docbank's lack of specific labels for proofs or theorems, complicating its use for our problem, given our dataset's focus on documents that definitively contain proofs and theorems.

A noteworthy feature of Docbank is its sourcing of documents from arXiv, associating each with an arXiv ID. This linkage permits access to the LaTeX sources of the papers, enabling the application of our preprocessing script for ground truth annotation (proofs and theorems) within the Docbank dataset, thereby broadening its utility. The adoption of the `\begin` command for annotations by Docbank's authors parallels the methodology utilized in our research for marking structural segments in scientific documents, illustrating a shared approach in identifying and analyzing document components like proofs and theorems.

Doclaynet [36], paralleling the efforts of Docbank [24] and Publaynet [47], targets layout detection in documents with a focused dataset of 80K instances. This dataset extends beyond scientific articles to include a broader range of paper layouts, aiming to achieve the detection precision of models like FASTRCNN [39] and YOLOv5 [19]. A notable challenge identified in Doclaynet is the presence of overlapping labels, where blocks share intersecting

[5] https://dtd.nlm.nih.gov/

labels, a complexity also acknowledged in our research. To mitigate this, both studies prioritize the analysis of non-overlapping blocks for evaluation. 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 suffices to enhance performance on the validation data, underscoring the efficiency of our unimodal approaches.

3 Unimodal Models

We now present the methodology of our three unimodal models: a pretrained transformer (RoBERTa-based) language model for text extracted for each paragraph of the PDF; an EfficientNetv2M [40] CNN for vision on the bitmap rendering of each PDF paragraph; and an LSTM model trained on font information sequences within each paragraph. For a technical reason explained in Section 5, the problem is formulated as a four-class classification: in addition to the three target *basic text*, *theorem*, *proof*, we employ a reject *overlap* class.

3.1 Text Modality

Pretraining language models. The intricacy of scientific terminology presents challenges for natural language processing models, necessitating domain-specific pretraining to improve their understanding of scientific language. Following insights from [12], which demonstrated performance gains through additional pretraining on diverse datasets [3], we adopt a tailored approach. Instead of extending an existing model like RoBERTa, we pretrain our model from scratch using a corpus of mathematical articles, aiming for a direct comparison with models trained on general English. Note that final performance of the language model at our task is not our only target: we are also interested in models that, after pretraining, require fewer samples to fine-tune.

Related work (text modality). Several existing works have built a domain-specific language model for scientific papers, such as SciBERT [2], BioBERT [23], and MathBERT [35]. MathBERT is pretrained on mathematical texts and formulas, showing notable efficacy in tasks like mathematical information retrieval and formula classification, albeit necessitating access to LaTeX sources, unlike our PDF-based approach. BioBERT focuses on medical science, diverging from our focus, while SciBERT covers a broader spectrum, including computer science, making it a relevant baseline for our experiments. This comparison aims to assess the effectiveness of domain-specific pretraining in enhancing model performance with potentially less data, setting the stage for future exploration into more extensive, multi-billion parameter models.

Previous research [29] has also underscored the significance of pretraining, showing that models trained on just 4 GB of web-crawled data can outperform those trained on over 130 GB, especially on domain-specific tasks.

Methodology. We pretrain a language model from scratch on a 50k vocabulary size (with byte-pair encoding), similar to the configuration of RoBERTa base (124M) [25]. While masking 15% of tokens we kept the configuration similar to original RoBERTa ($L = 12, H = 768, A = 12$), but on a different vocabulary. The model used dynamic masking and was trained on masked language modeling loss. After pretraining, the model is fine-tuned for our classification task.

3.2 Vision Modality

Related work (vision modality). Architecture design significantly influences model performance, evolving substantially from the early Lenet-5 [22].

ResNet [14] pioneered the use of skip connections, a concept expanded upon by DenseNet [17], which connected each layer to all its predecessors. Following this, NASNet [48] leveraged neural architecture search (NAS) for optimal architecture selection via reinforcement learning. Advancements continued with EfficientNet [40], which employed NAS to fine-tune hyperparameters and introduced compound scaling, significantly enhancing efficiency and performance over predecessors like NASNet, making it suitable for multimodal frameworks by minimizing computational demands. The debut of vision transformers [9] challenged CNNs' supremacy, demonstrating superior results with ample data. However, due to computational constraints, our focus remains on CNNs, particularly after studies [26, 41, 44] showed recent CNNs, including EfficientNetv2 [41], achieving comparable performance to transformers at a much lesser computational cost.

Indeed, EfficientNetv2 introduces a model family that significantly outpaces its predecessor, EfficientNet, in training speed and parameter efficiency on various datasets. It surpasses the Vision Transformer (ViT) [9] in performance while maintaining a considerably smaller size. This efficiency makes EfficientNetv2M — our chosen model — ideal for use as a network backbone. Key to its performance is the adoption of Fused-MBConv in early network stages, replacing depth-wise convolutions (MBConv) and optimizing for accuracy, parameter, and training efficiency through training-aware NAS. This approach, focusing on a 3×3 kernel size while adding depth, ensures EfficientNetv2's effectiveness even with the smaller image sizes of the ImageNet dataset, which typically challenges ViT models in training time and memory usage.

Methodology. CNNs, pivotal in image classification and as backbones in visual-language tasks, typically benchmark on ImageNet and CIFAR for top-1% accuracy. Our project, targeting the identification of mathematical symbols and the layout of paragraph blocks to discern proofs and theorems, necessitates model training from scratch. Distinct markers like the term "Proof" in unique fonts and the QED symbol, crucial yet overlooked by text modalities, guide our focus.

One specificity of vision approach for classification block is that images come in widely different **aspect ratios**. Traditional interpolation methods, though prevalent for adjusting natural images to a uniform resolution, unsuitably modify the geometry of text, symbols, and fonts in our context. Based on corpus analysis, we establish a fixed resolution of (400×1400) pixels. This size accommodates over 80% of our paragraphs, with larger images being cropped and smaller ones padded to maintain this standard without altering their intrinsic visual properties. This approach aligns with recommendations against scale variance [43] and parallels the preprocessing strategy used in the Nougat paper [4], which

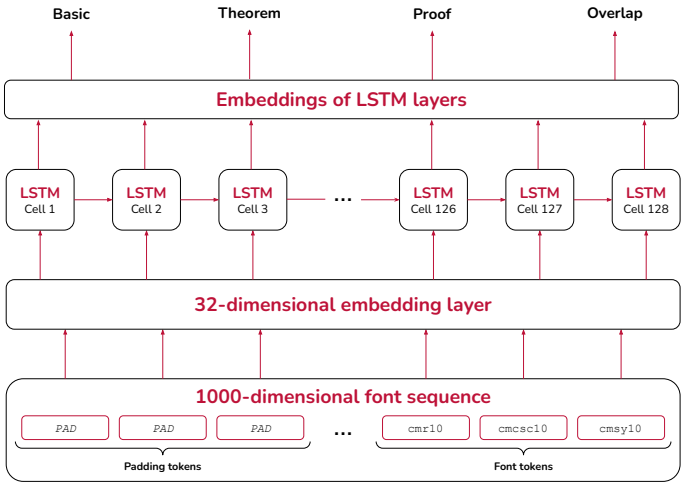

Figure 2: LSTM model for the font modality

also maintains a constant aspect ratio to suit specific model inputs. Our method ensures the preservation of textual image integrity by avoiding the pitfalls of resizing, opting instead for cropping or padding to fit our predetermined resolution criteria.

To counteract the issue of **white backgrounds** in scientific texts, which can hinder CNN performance as noted by studies [16], we invert image colors to mimic the MNIST dataset's white-on-black text presentation. This approach prevents max-pooling operations in CNNs from mistakenly prioritizing the background, thereby maintaining focus on the textual content.

EfficientNet comes with several variants (B0–B7) where B7 has the largest receptive field due to compound scaling. We select in our experiments a base network (B0), a medium-sized network (B4) and the largest network (B7). EfficientNetV2 also comes with different sizes. We focused on the small (EfficientNetV2s) and medium-sized (EfficientNetV2m) models.

3.3 Font Modality

The last modality we consider is styling information present in the PDF in terms of the sequence of fonts (font family and font size) used in a specific paragraph. This information can be obtained using the pdfalto tool[4], which produces a list of fonts used in a given document, and associates each text token to a particular font. Fonts are usually standard LaTeX fonts, such as `cmr10` for Computer Modern Roman in 10 point.

From the training data, we build a font vocabulary of 4 031 unique fonts including their sizes, and represent every paragraph block as a sequence of font identifiers. To match input dimensions among training samples, we apply left padding with a maximum length of 1 000. We then feed the entire sequence to a simple 128-cell LSTM [15] network to monitor the loss, represented in Figure 2. The choice of the model is purely to capture sequential information within fonts that can be used to identify the label of the paragraphs.

4 Multimodal and Sequential Models

We now go beyond unimodal models by showing how all three modalities can be combined into a single late-fusion multimodal model, and how block sequence information can be captured.

Related work (Multimodality). Multimodal machine learning for document AI has seen a surge in interest. However, existing models often fall short in addressing the unique aspects of scientific articles, such as font features, scientific terminology, and the structure of lengthy documents. Most research focuses on benchmarks like FUNSD [11], CORD [34], and RVL-CDIP [13], which deal with simpler document types like invoices and forms assuming dependencies within the same page.

Several architectures have been proposed as multimodal transformers which try to jointly model different modalities in a single transformer model early on during the input stage and try to capture modality interactions such as LayoutLM [18, 45, 46], which takes into account 2D positional embedding via masked visual-language model loss and multi-label document classification loss. An alternative approach is late fusion (after feature extraction) such as CLIP [37]. In CLIP, both text and visual features are projected to a latent space with identical dimensions and a contrastive loss is applied to zero shot learning with supervision from language models. One of the big advantages of CLIP is its ability to upgrade and replace the backbone on the fly.

We adopt a late fusion based approach (similar to CLIP) instead of early fusion based approaches such as LayoutLM for the following reasons: (i) **Modular backbone integration:** Our late fusion approach is driven by the flexibility to integrate various backbones in a modular way, enhancing performance and scalability without being constrained by fixed architecture dimensionality. (ii) **Reevaluating cross-modality capture:** the specialized losses for cross-modal interactions, like those in LayoutLMv2 and LayoutLMv3, are claimed to enhance cross-modality relationship understanding. However, this assumption warrants further scrutiny. Specifically, the specificities of LayoutLM architectures mean that they cannot be compared to straightforward multimodal fusion strategies employing identical backbones. Here, we conduct direct comparisons across multiple fusion methods, focusing on raw features and employing only cross-entropy classification loss, while maintaining consistent backbones as in unimodal setups.

Methodology. We compare different modalities for late fusion: bilinear gated units [20], EmbraceNet [6], gated multimodal units (GMU) [1], and attention mechanisms such as those of ViLBERT [28], typically applied to dual modalities but extendable to multiple. These methods, focusing on feature-level fusion, ensure modularity and adaptability across architectures.

Our comparison to simple fusion methods is narrowed to concatenation, identified as the most effective among basic fusion strategies, allowing direct comparison with our unimodal baselines. These comparisons solely rely on cross-entropy loss for classification tasks, omitting additional losses like contrastive loss. Importantly, the feature backbones are frozen during fusion, preventing weight updates and situating multimodal fusion as an augmentation to our unimodal framework.

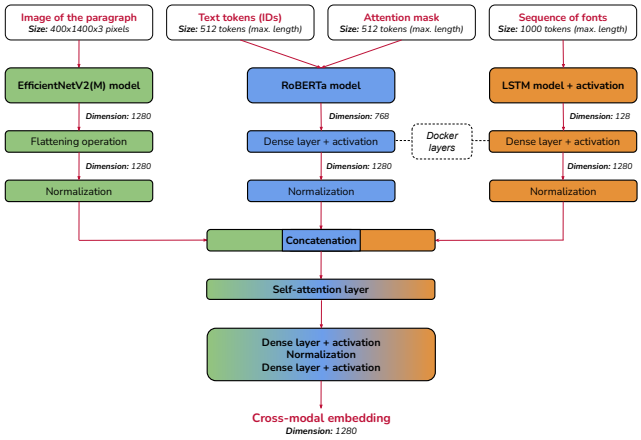

Figure 3: Cross-modal attention architecture

Cross-modal attention architecture. The main multimodal model we use is a cross-modal attention model, inspired by ViLBERT's attention mechanism. We show its full architecture in Figure 3.

Sequential approach. In addition to modalities, considering the sequencing of the blocks, i.e., the order in which they appear in the document, allows us to determine with greater confidence the class of each block. For example, if one seeks to predict the class of a block that is itself framed by two blocks that have been classified as *proof*, then there is a good chance that this block is itself a *proof*. We consider how to integrate unimodal and multimodal models into a sequential prediction model.

To classify scientific paragraphs spanning multiple pages, long document classification methods like Tobert, Bigbird, Longformer, and Hierarchical Attention Transformer (HAT) are more relevant. HAT outperforms Bigbird and Longformer (see Table 5 of HAT paper) and efficiently handles long documents, including page breaks. However, these models lack multimodal capabilities and document-specific information, such as coordinate data used in LayoutLM and LILT. Our approach combines the strengths of both classes while using the same backbones. It is computationally efficient, relying on the SW mechanism instead of full attention, and saves nearly half the parameters per encoder by reducing the feedforward dimension.

We propose two approaches to do this: First, using a simple linear-chain order-one Conditional Random Fields model (CRFs) [21]. Second, we introduce a novel transformer-based BERT-like encoder architecture (also more efficient for our task) to process multimodal features, using a sliding window (SW) of size $k = 16$, whose architecture is presented in Figure 4. We also investigate the impact of long sequential relationships by employing interleaving architecture found in Hierarchical Attention Transformers (HATs) [5]. The architecture is modified to be adapted in a multimodal setting such as ours.

The CRF and SW models use the following features, on top of frozen unimodal or multimodal model: unimodal text, vision, and font models respectively bring 768, 1280, and 128 features; the multimodal approach includes 1280 joint features; we incorporate four additional geometrical features to describe block positions: normalized page number, indicating a block's page relative to the total pages; normalized horizontal and vertical distances from the block's bounding box corners; and a binary feature indicating if a block and its predecessor are on the same page.

In order to determine whether long-distance dependencies are also useful to capture for our task, we also implement HATs, relying on the same Sliding Window transformer encoder architecture used as as a segment-wise encoder. We then expanded it to learn about connections between different context windows (using cross segment encoder) taking only the Multimodal [CLS] token of every segment. Out of the many versions proposed in the original HAT paper [5], we tested the best-performing one, i.e., with interleaving layers. See Figure 5 for the corresponding architecture.

5 Dataset and Setup

We use Grobid[6] [27], which is the state of the art for information extraction from scholarly documents to parse a PDF document and interpret it into a succession of paragraph blocks.

Our dataset, encompassing all arXiv papers (around 1.7 million 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 LaTeX sources. This involved crafting a LaTeX package to instrument commands such as `\newtheorem` for precise identification in the compiled PDFs (≈ 460k papers). See Figure 6. We filtered articles from the dataset to only keep those in English, for which LaTeX source is available (according to arXiv's policy, all those that have been produced using LaTeX), that were compilable on a modern LaTeX distribution, that contained at least a theorem or a proof environment, and 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. This resulted in a final dataset of ≈ 197k papers. We stress that LaTeX 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 category (say, *basic* and *theorem*). We label such paragraphs as *overlap*, exclusively used for such outliers.

Our validation set comprises approximately 500 000 paragraph blocks from 3 682 randomly selected PDF articles. The remaining articles formed the training dataset, used entirely for pretraining our language model after filtering potential personal information such as author names and institutions from Grobid extractions to minimize privacy concerns. 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 Section 4. The dataset is heavily imbalanced, with the number of paragraphs labeled as *basic*: 314 501, *proof*: 125 524, *theorem*: 85 801, and *overlap*: 3 470.

All experiments were run on a supercomputer with access at any point to 4 NVIDIA (V100 or A100) GPUs. We estimate to 8 000

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

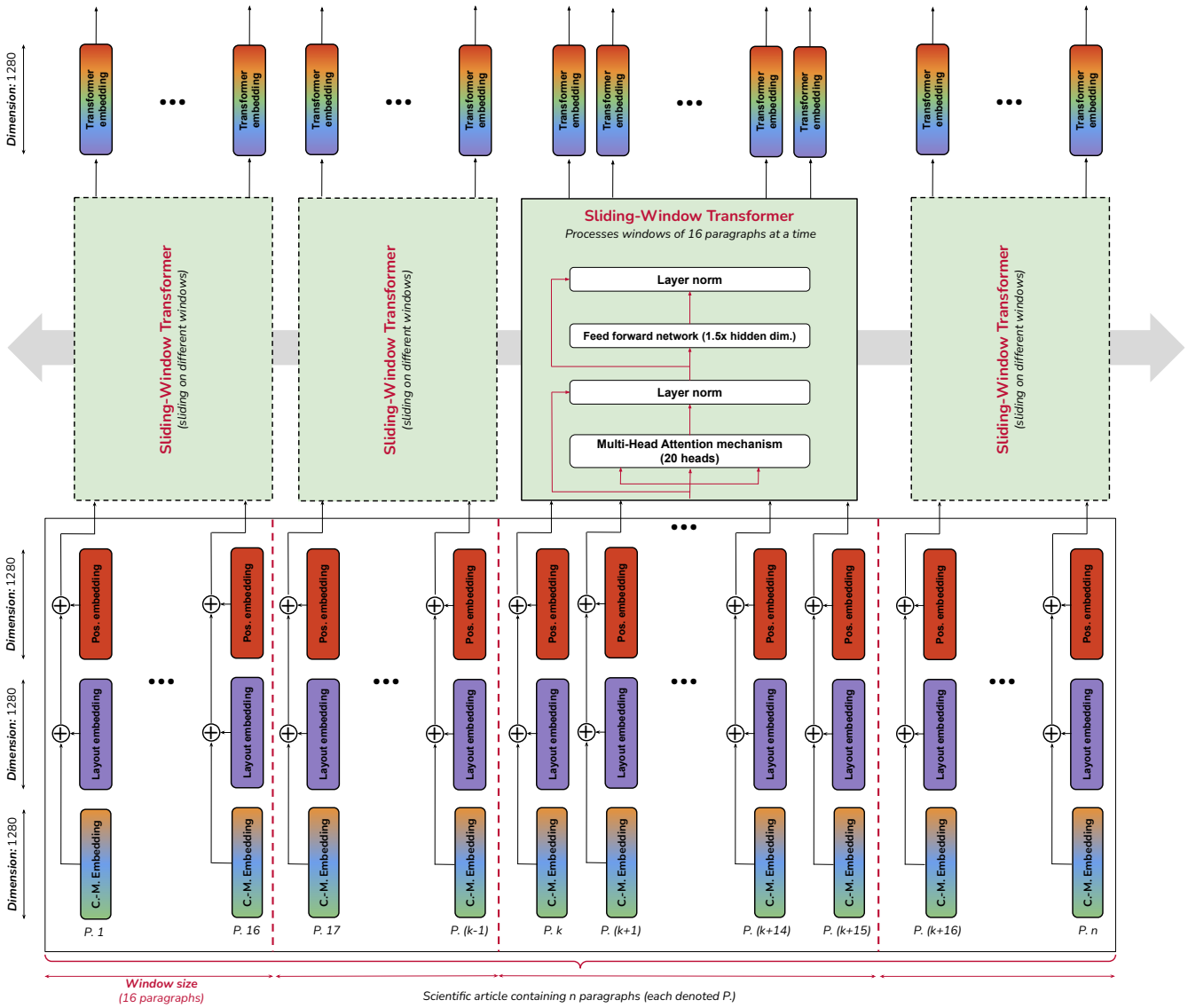

Figure 4: Sequential model based on a sliding-window (SW) transformer architecture

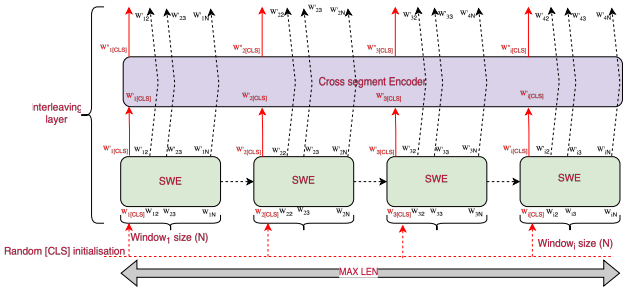

Figure 5: HAT network visualization (with 1 interleaving layer)

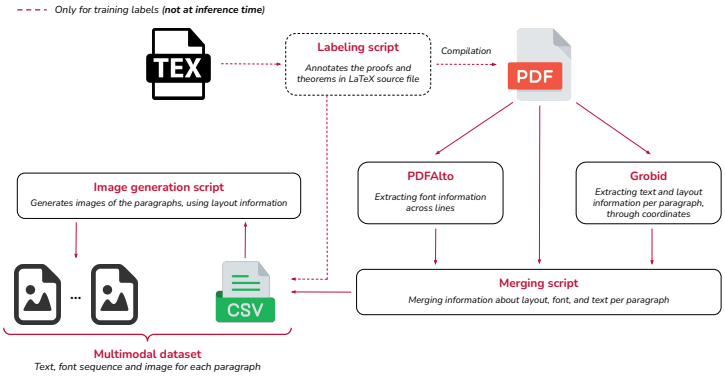

Figure 6: Dataset preparation pipeline

GPU hours the computational cost of the entire prototyping, hyperparameter tuning, training, validation, and evaluation pipeline.

6 Experimental Results

We now report experimental results on the *basic–theorem–proof* classification problem, first comparing representative unimodal classifiers, with and without the article paragraphs fed to the sequential approach, followed by the multimodal classifier. We then delve into more specific details of every unimodal classifier.

We are interested in two main performance metrics: *accuracy* measures the raw accuracy of the classifier on the validation dataset (disjoint with the training dataset); and (unweighted arithmetic) *mean F_1-measure* of the *basic*, *theorem*, and *proof* classes, which summarizes the precision and recall over each class assigning the same weight to every class. As *basic* is the most common class in the dataset, a *dummy* classifier that would always predict the *basic* class would have an accuracy of 59.41%; but its recall would be 100% on *basic* and 0% on the other classes, while its precision would be 59.41% on *basic* and 0% on the other classes, resulting in a mean F_1 of $\frac{1}{3} \times \frac{2 \times 59.41\%}{59.41\% + 100\%} \approx 24.85\%$. This gives an important comparison point for all other methods; accuracy measures how well the classifier works on the actual unbalanced data, while mean F_1 favors methods performing well to identify all three classes, arguably a better metric.

Drawing inspiration from two related works, albeit applied in slightly different settings, we evaluate two straightforward baselines: (1) Top-k first words: This method, which echoes the approach used in [10] focusing on the first paragraph of marked environments, constructs a vocabulary of the top-k unique words for each class. Labels are assigned based on the first word of a text and whether it matches any word within the class-specific vocabulary. For instance, if the first word is within {*theorem, lemma, proposition, definition*}, the text is labeled as a theorem. (2) Text classifier from [32]: We reuse the text classifier that was fine-tuned in [32], which processes text lines (not paragraphs) extracted from pdfalto. Note this classifier does not identify the *overlap* class.

Overall Results. The results we obtain for the different modalities, with and without the use of either CRF or sliding-window block sequence model, are shown in Table 1. The following lessons can be drawn from these results:

(1) This is a hard task, as the best performance reached is 88% for accuracy and 87% for mean F_1. Indeed, it can be hard even to a human to determine whether a block is part of a proof or theorem environment, especially in the middle of it, so it is unsurprising that we cannot reach near-perfect results.

(2) Looking at unimodal models: the font-based model performs rather poorly, though still beating (at least in terms of mean F_1) the three baselines; the text-based model is the best performing one, suggesting that textual clues impact more than visual ones for this task.

(3) The multimodal model outperforms every unimodal model, though the margin with the text model is somewhat low.

(4) Including the Sequential model (both CRF, SW transformer, or HAT) greatly increases the performance of every unimodal or multimodal model, by 5 to 10 points of accuracy or mean F_1. The importance of the use of an approach modeling block sequences is thus clear. Long-distance dependencies captured by HATs do not seem to matter.

Multimodal approach. To look more in detail at the impact of the choice of multimodal fusion strategies, we report the performance of a variety of them in Table 2 with the cross-modal attention technique described in Section 4 highlighted in bold. We note that results of most multimodal approaches are actually quite close to each other, which hints at the robustness of our observation that adding a multimodal model on top of our three unimodal models improves in all cases the performance on our classification task.

Individual modalities. We now discuss the performance of different unimodal models.

To measure the performance of various language models (our language model pretrained on our corpus, RoBERTa, and SciBERT), we evaluate their accuracy as shown in Table 3 on the validation dataset. All three have similar numbers of parameters (obviously the same for our pretrained and the base version of RoBERTa), have similar inference time, and reach similar levels of accuracy (76.45% to 76.89%) and mean F_1 (71.66% to 73.00%) and converge after training on 20 batches. SciBert does have slightly higher performance. Table 3 shows another side of the picture: to reach a target level of accuracy (say, 65% or 70%), our pretrained model needs much fewer fine-tuning data than the RoBERTa model (trained on a corpus of 15 times more text data). Although SciBERT performs better than our pretrained model on this metric as well, note that it has been trained on 5.5 times more scientific papers than our pretrained model.

The performance of a wide variety of vision-based models is displayed in Table 5. For the simplest model, we experiment with different forms of pooling for the last convolutional layer: none, max pooling, or average pooling. We see that no pooling yields performance, on a small model that is quite close and comparable to much larger models. We also notice that average pooling works quite well in most cases, while also cutting the number of parameters to nearly a third.

For font sequence information, in addition to our model formed of an LSTM with 128 cells, and in order to investigate potential further gains, we try switching LSTM cells to GRU [7]; and using a Bidirectional LSTM to capture sequential information across both forward and backward axis. Our results from Table 6 indicate that the bidirectional component in fonts alone does not have a huge impact in deciding the label of the blocks, even if modest gains are observed.

We finally show in Table 7 a partial classification report, for the best model in each class.

7 Conclusion

Summarizing the results obtained in the previous section, we put forward our multimodal model with block-sequential sliding-window transformer model as a state-of-the-art candidate for identification of theorems and proofs for scientific articles. The level of accuracy and mean F_1 reached, if not perfect, is acceptable for automatic processing of articles and construction of a knowledge base of theorems, which may need to be further manually cleaned and curated.

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)

Modality	Model chosen	Seq. approach	#Batches	#Params (M)	Accuracy (%)	Mean F_1 (%)
Dummy	always predicts *basic*	—	—	—	59.41	24.85
Top-*k* first word	use only first word	—	—	—	52.84	44.20
Line-based [32]	Bert (fine-tuned)	—	—	110	57.31	55.71
Font	LSTM 128 cells	-	11	2	64.93	45.48
		CRF	11+8	2	71.50	64.51
		SW Transformer	11+8	2	76.22	71.77
Vision	EfficientNetV2m_avg	-	9	53	69.44	60.33
		CRF	9+8	53	74.63	70.82
		SW Transformer	9+8	65	79.59	77.66
Text	Pretrained RoBERTa-like	-	20	124	76.45	72.33
		CRF	20+8	124	83.10	80.99
		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**	**87.18**
		HAT	2+8	232	87.52	86.58

Table 2: Performance comparison of multimodal fusion techniques (with @dimensions and model architecture)

Model Architecture	#Params (Total/Trainable)	Accuracy (%)	Mean F_1 (%)
Concatenated raw features(@2176)	179M/8K	77.90	74.34
docker layers(@1280) + concat(@3840)	180M/1M	78.11	74.95
docker layers(@1280) +fusion (@768)	183M/4M	78.50	75.38
docker layers(@1280) +fusion (@1280)	185M/6M	78.43	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	75.11
docker layers(@1280) +GMU mechanism (@1280)	185M/6M	78.11	75.52
docker layers (@1280) +attention mechanism (@1280)	**185M/6M**	**78.50**	**75.37**
docker layers(@1280) +multihead attention (@1280, 8 heads)	244M/65M	78.33	75.26
EmbraceNet mechanism (@1280) (balanced prob +docker layers (@1280) incl)	182M/3M	77.73	74.70
EmbraceNet mechanism (@1280)(weighted prob +docker layers (@1280) incl)	182M/3M	77.73	74.55
docker layers(@1280) +fusion (@2304) +fusion (@768)	191M/12M	78.50	75.32
docker layer (@1280) +Cross-modal attention (@1280) +fusion (@768)	186M/7M	78.45	75.24
docker layer (@1280) +GMU mechanism (@1280)+fusion (@768)	186M/7M	78.33	75.28

Table 3: Performance comparison of text models

Model	#Batches	Inf. time (ms/step)	Accuracy (%)	Mean F_1 (%)	#Params
Dummy	—	—	59.41	24.85	—
RoBERTa base	20	23	76.61	71.66	124M
Pretrained	20	23	76.45	72.33	124M
SciBERT base	20	23	**76.89**	**73.00**	110M

Table 4: 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

Text stands out as the most effective single modality for our analysis, surpassing vision and font sequence in performance, though the latter boasts the highest efficiency and minimal parameter usage, approximately 70 times less than our language model.

An important distinctive feature of our research is that we focus on analyzing scientific documents spanning multiple pages, unlike typical document AI methods designed for single-page documents (e.g., receipts, bills) and simpler tasks (e.g., total bill calculation, document type classification).

Here are some advantages of our approach:

Table 5: Performance comparison of vision models

Model	#Batches	Inf. time (ms/step)	Accuracy (%)	Mean F_1 (%)	#Params
Dummy	-	-	59.41	24.85	-
EfficientNetB0	5	29	65.27	46.00	6.9M
EfficientNetB0_max	5	35	58.22	25.00	4.0M
EfficientNetB0_avg	5	34	62.93	39.66	4.0M
EfficientNetB4_avg	5	61	65.87	47.33	17.6M
EfficientNetB7_avg	5	145	61.22	42.33	64.1M
EfficientNetV2s_avg	5	70	59.41	25.00	20.3M
EfficientNetV2m_avg	5	94	64.02	42.66	53.2M
EfficientNetB4_avg	9	88	68.47	54.33	17.6M
EfficientNetV2s_avg	9	71	59.81	27.00	20.3M
EfficientNetV2m_avg	9	92	**69.44**	**60.33**	53.2M

Table 6: Performance comparison of font models

Model	#Batches	Inf. time (ms/step)	Accuracy (%)	Mean F_1 (%)	#Params
Dummy	-	-	59.41	24.85	-
LSTM (128)	11	14	64.93	45.48	1.72M
GRU (128)	11	14	60.59	42.71	1.72M
BiLSTM (128)	11	26	64.71	45.66	1.82M

Table 7: Class-wise precision and recall scores for best unimodal, multimodal, and sequential models

	Font		*Vision*		*Text*		*Multimodal*		*Sequence CRF*		*Sequence Transformers*	
	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

• ***Processing entire PDFs while capturing sequential dependencies:*** Our model processes entire PDFs, generating labels for each paragraph in a single forward pass and capturing sequential dependencies across pages. For reference, our 198M model can process an entire PDF at once, whereas LayoutLMv3 [18] (368M) and Nougat (250M) [4] can only process one page at a time.

• ***Modular and multimodal:*** Our approach is both multimodal and modular. We demonstrate this by integrating a custom pretrained Roberta model, which will allow us to switch to different text backbones to extend this work (e.g., LLAMA [42], SPECTER [8], ORCA-MATH [33]) without redesigning the entire architecture as for future versions of the LayoutLM family. This flexibility mirrors the CLIP [38] model's approach (see Table 10 of the CLIP paper) and is not possible with other models like LayoutLM.

• ***Scalability and speed:*** Unlike LayoutLM models that require OCR (adding to inference time, see α parameter in table 1 of Donut paper that factors the OCR time when comparing to LayoutLM). While Nougat and Donut are OCR-free and faster, they are still slower than our approach, to be used in a real-world setting to help researchers. For reference, Nougat takes 19.6 seconds for 6 pages on an Nvidia A10G, whereas Grobid processes 10.6 PDFs/sec on a CPU (see limitation section of Nougat paper). This efficiency is crucial to evaluate our dataset of 200k papers, which would take several months. We also cut down the number of parameters by switching the vanilla transformer encoder block to a more efficient sliding window encoder block that reduces the number of parameters leading to a reduced inference time and memory usage.

• ***Alternate and lighter models:*** Unlike typical model sizes offering small and large variants of the same model (in terms of parameter count) models, we provide models at different modality levels. This allows for the integration of extremely lighter alternatives, such as a 2M parameter CRF model trained on fonts, which achieves 71% accuracy for an extremely low-resource setting, see table 7 for per class modality performance.

One limitation of our work is that we have only trained and evaluated our model on English-language mathematical articles. Though they represent a significant portion of the mathematical literature, articles in other languages do exist and one would need to check whether the approach proposed extends to, say, Russian- or Arabic-language articles. This can be extended by removing the English language filter in the preprocessing step.

Our approach is a first building block towards building a knowledge base of theorems from the raw PDFs, but further research is required before one will be able to provide such an application.

8 Ethical Considerations

We do not envision any major ethical concerns from the development of machine learning models to extract mathematical statements and proofs from mathematical articles.

As for all imperfect methods, results should not be blindly used in settings where perfect accuracy is required.

The training of machine learning models in general, and deep learning models in particular, requires a significant amount of computation time and energy consumption, which contributes to the production of greenhouse gases and global warming. We attempted to somewhat mitigate this by focusing on models with good performance but reasonable numbers of parameters. We also note that the supercomputer used for the computation is powered with low-carbon electricity, and that residual heat produced by the computing power is used as part of an urban heat distribution network.

Finally, note that though we used publicly available research articles from arXiv, acquired respecting arXiv's terms and conditions, redistribution of the dataset is not allowed by the arXiv licensing agreement (except for the few papers that are explicitly marked with a Creative Commons license). Distribution of the model learned from this publicly available dataset is somewhat of a grey legal area (as is the release of all machine learning models trained on publicly available data without a specific license of use, such as most openly available large language models). To allow reproducibility while respecting licensing terms, we provide at https://github.com/mv96/mm_extraction: full instructions on how to rebuild the same dataset by retrieving data from arXiv; all ground truth annotations (label of each paragraph block of each article in the dataset); trained models; full code to train them, released as free software.

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 allocation 2020-AD011012097 made by GENCI (Jean Zay supercomputer).

References

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

[2] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *EMNLP/IJCNLP*.

[3] Steven Bird, Robert Dale, Bonnie J. Dorr, Bryan Gibson, Mark T. Joseph, Min-Yen Kan, Dongwon Lee, Brett Powley, Dragomir R. Radev, and Yee Fan Tan. 2008. The ACL Anthology Reference Corpus: A Reference Dataset for Bibliographic Research in Computational Linguistics. In *LREC*.

[4] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. 2023. Nougat: Neural optical understanding for academic documents. *arXiv preprint arXiv:2308.13418* (2023).

[5] Ilias Chalkidis, Xiang Dai, Manos Fergadiotis, Prodromos Malakasiotis, and Desmond Elliott. 2022. An exploration of hierarchical attention transformers for efficient long document classification. *arXiv preprint arXiv:2210.05529* (2022).

[6] Jun-Ho Choi and Jong-Seok Lee. 2019. EmbraceNet: A robust deep learning architecture for multimodal classification. *Information Fusion* 51 (2019), 259–270.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555* (2014).

[8] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 2270–2282. https://doi.org/10.18653/V1/2020.ACL-MAIN.207

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

[10] Deyan Ginev and Bruce R. Miller. 2020. Scientific Statement Classification over arXiv.org. In *LREC*.

[11] Jean-Philippe Thiran Guillaume Jaume, Hazim Kemal Ekenel. 2019. FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents. In *OST@ICDAR*.

[12] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: adapt language models to domains and tasks. In *ACL*.

[13] Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. 2015. Evaluation of Deep Convolutional Nets for Document Image Classification and Retrieval. In *ICDAR*.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

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

[16] Hossein Hosseini, Baicen Xiao, Mayoore Jaiswal, and Radha Poovendran. 2017. On the limitation of convolutional neural networks in recognizing negative images. In *ICMLA*.

[17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *CVPR*.

[18] 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*.

[19] Glenn Jocher. 2020. ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements.

[20] Douwe Kiela, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2018. Efficient large-scale multi-modal classification. In *AAAI*.

[21] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*.

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998).

[23] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (2020).

[24] Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. 2020. DocBank: A benchmark dataset for document layout analysis. *arXiv preprint arXiv:2006.01038* (2020).

[25] 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. *arXiv:1907.11692* (2019).

[26] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A ConvNet for the 2020s. In *CVPR*.

[27] Patrice Lopez. 2009. GROBID: Combining Automatic Bibliographic Data Recognition and Term Extraction for Scholarship Publications. In *ECDL*, Vol. 5714. 473–474.

[28] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems* 32 (2019).

[29] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, É ric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a tasty French language model. In *ACL*.

[30] Shrey Mishra. 2024. *Multimodal Extraction of Proofs and Theorems from the Scientific Literature*. Ph. D. Dissertation. Université Paris Sciences & Lettres.

[31] Shrey Mishra, Yacine Brihmouche, Theo Delemazure, Antoine Gauquier, and Pierre Senellart. 2024. First steps in building a knowledge base of mathematical results. In *Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024)*. 165–174.

[32] Shrey Mishra, Lucas Pluvinage, and Pierre Senellart. 2021. Towards extraction of theorems and proofs in scholarly articles. In *DocEng*.

[33] Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-Math: Unlocking the potential of SLMs in Grade School Math. *CoRR* abs/2402.14830 (2024). https://doi.org/10.48550/ARXIV.2402.14830 arXiv:2402.14830

[34] Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. 2019. CORD: A Consolidated Receipt Dataset for Post-OCR Parsing. In *DI@NeurIPS*.

[35] Shuai Peng, Ke Yuan, Liangcai Gao, and Zhi Tang. 2021. MathBERT: A pre-trained model for mathematical formula understanding. *arXiv:2105.00377* (2021).

[36] Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S Nassar, and Peter Staar. 2022. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3743–3751.

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 8748–8763. http://proceedings.mlr.press/v139/radford21a.html

[39] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28 (2015).

[40] Mingxing Tan and Quoc Le. 2019. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*.

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

[42] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[43] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. 2019. Fixing the train-test resolution discrepancy. *Advances in neural information processing systems* 32 (2019).

[44] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. 2023. ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders. *arXiv:2301.00808* (2023).

[45] 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*.

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

[47] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 1015–1022.

[48] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. 2018. Learning transferable architectures for scalable image recognition. In *CVPR*.

A Extra Material for Section 3 (Unimodal Models)

A.1 Text Modality

We investigate the difference in base vocabulary of different language models: Figure 7 compares the overlap of vocabulary between various language models – the one we are proposing (`trained_tokenizer` in the figure) has a maximum of 33% overlap with others, including SciBERT which is trained on scientific text, suggesting the relevance of a pretrained model with vocabulary specific to our corpus.

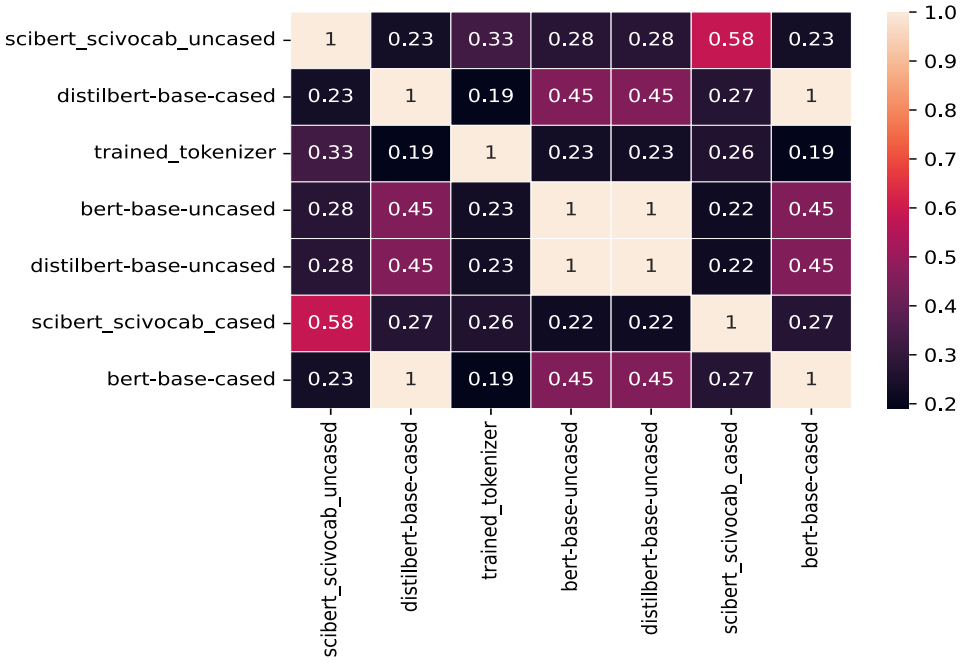

Figure 7: Vocabulary overlap among popular language models (BERT, DistilBERT, SciBERT, in cased or uncased variants) and our pretrained model (labeled as `trained_tokenizer` here)

In our pretraining, we used 11 GB of pretraining text data (196 846 scientific articles, see Section 5), trained over 11 epochs. We used the LAMB optimizer [YLR+20] and produced the results using a total batch size of 256 across 4 NVIDIA A100 GPUs with a distributed mirrored strategy and an initial learning rate of 2×10^{-5}. The total pretraining time was 176 hours.

Beyond vocabulary, the nature and size of pretraining data critically influence a language model's performance and its speed of generalization. Research, including the Chinchilla study [HBM+22], highlights the significant impact of data size, introducing the "Chinchilla scaling laws". For example, Chinchilla (70B) outperformed Gopher (280B) by 7% by quadrupling its training data, demonstrating these principles even beyond a trillion tokens, as seen with the LLAMA [TLI+23] model (7B) surpassing GPT-3 (175B [BMR+20]).

In this paper, we choose not to utilize several billion-parameter models for the reasons outlined below:

- **Focus on integration over depth:** Our primary aim is theorem–proof identification, prioritizing the combination of various modalities for a holistic approach over delving into advanced, singular modalities that may not offer comparative advantages. We start with base models, considering scalability and relevance to our scientific domain.
- **Budgetary limitations:** The computational cost of training and evaluating large models, especially across multiple modalities, necessitates a pragmatic approach. We test with base models due to their feasibility and the modular nature of our framework, which allows for flexibility in model choice and scaling.

We report pretraining results on two pretraining configurations (see Table 8): a BERT-like model in addition to the RoBERTa-like model described in the main text. As a quantitative measure of the quality of the pretraining, we report the perplexity of the pretrained language model on the MLM task, similar to Table 3 in the RoBERTa paper [LOG+19]. We show the evolution of the MLM loss in Figure 8. For a qualitative analysis, we intentionally picked up samples that require specific vocabulary understanding on the MLM task, see Table 9.

An example of use of Grad-CAM for visualization of the attention heads of a language model in Figure 9.

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 8: Pretraining configurations (on arXiv dataset)

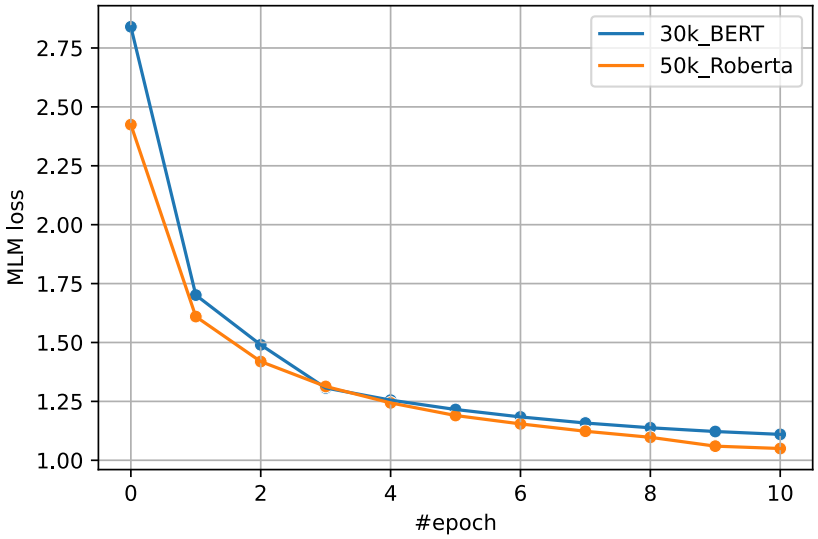

Figure 8: MLM loss for two pretrained models, as a function of the pretraining epoch

Masked sentence	Pretrained BERT-like model	BERT model
This concludes the [MASK].	proof lemma claim theorem case thesis	game story film play episode novel
We show this by [MASK].	induction . definition a lemma contradiction	ourselves accident name themselves ear hand
By [MASK]'s inequality.	jensen holder young minkowski cauchy	fourier brown russell fisher newton
The [MASK] is definite positive.	inequality case slimit sum function	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 grover tarjan newton hamilton	shannon newton taylor wilson moore
An illustration of the superiority of quantum computer is provided by [MASK]'s algorithm.	grover shor dijkstra yao kitaev	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 error errors misspecification noise	errors error this bias uncertainty

Table 9: Inference of BERT-like pretrained model on selected MLM tasks

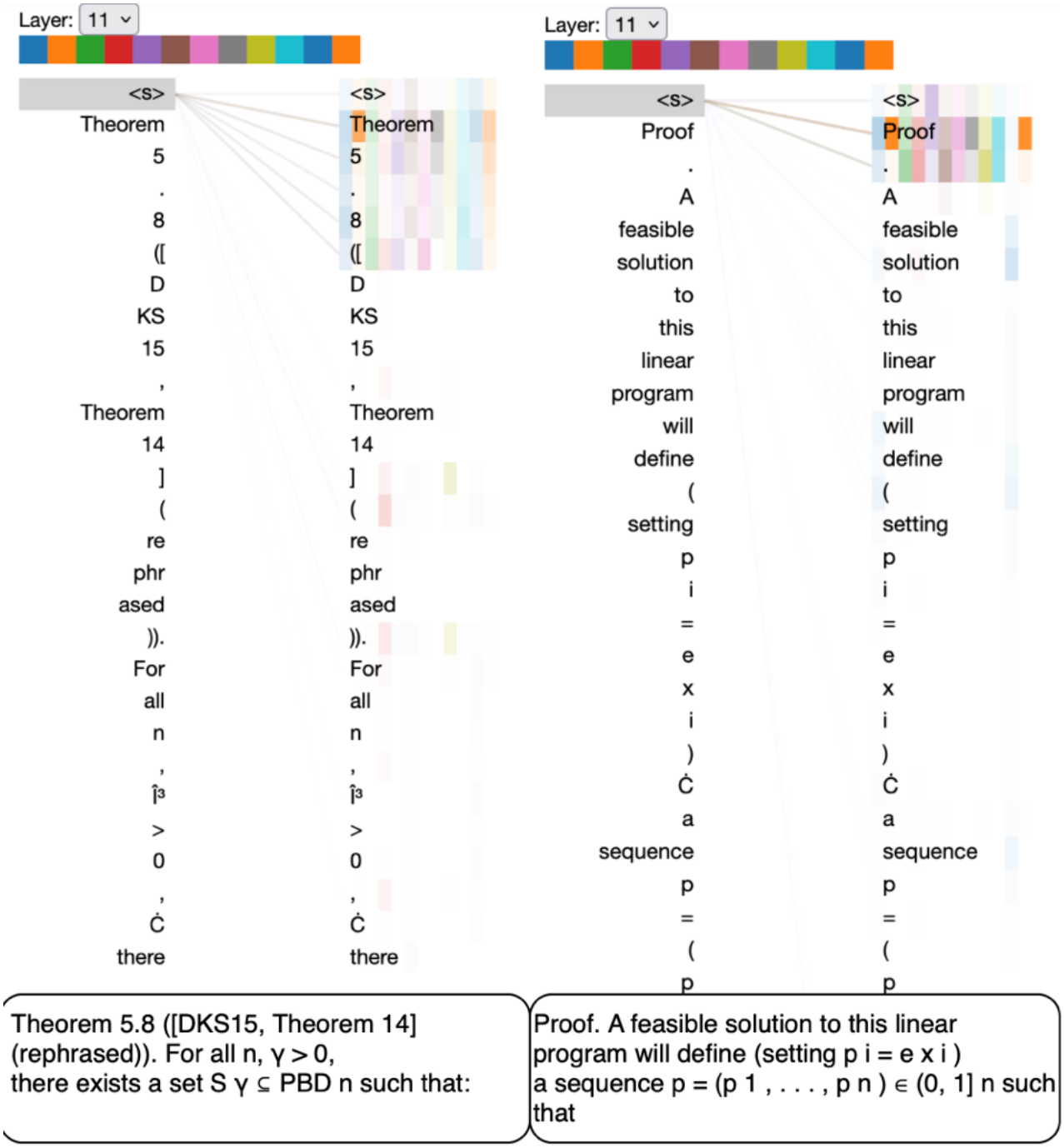

Figure 9: Visualising the attention maps of a finetuned transformer Language model

A.2 Vision Modality

Utilizing Grad-CAM [SCD+17], we visually demonstrate the visual model's attention to specific elements such as "Proof", "Theorem" keywords r the use of italics, highlighting its effectiveness in recognizing mathematical documentation nuances (see Figure 10).

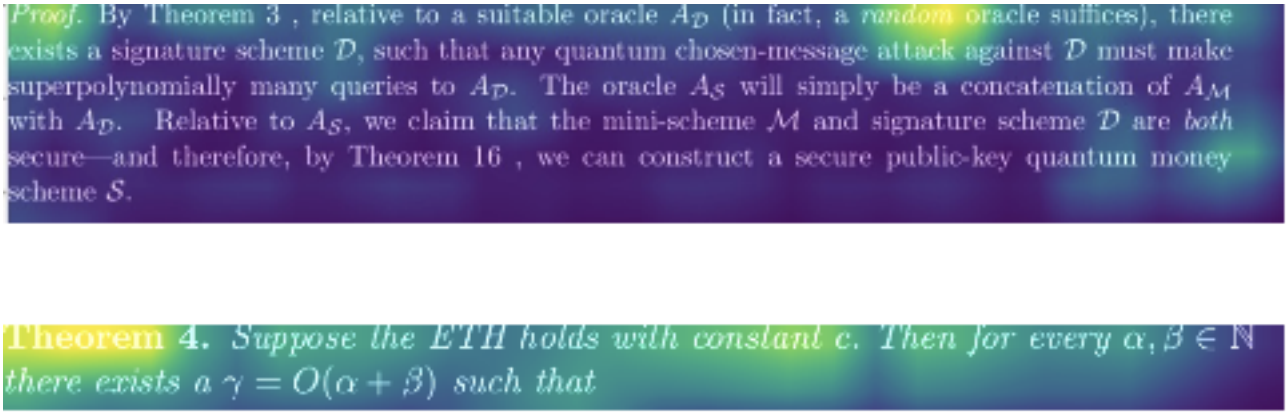

Figure 10: Grad-CAM visualizations of some sample blocks

In Figure 11, we plot the cumulative distribution of heights and widths of paragraph blocks in our dataset, which is used to fix the common target resolution of all images.

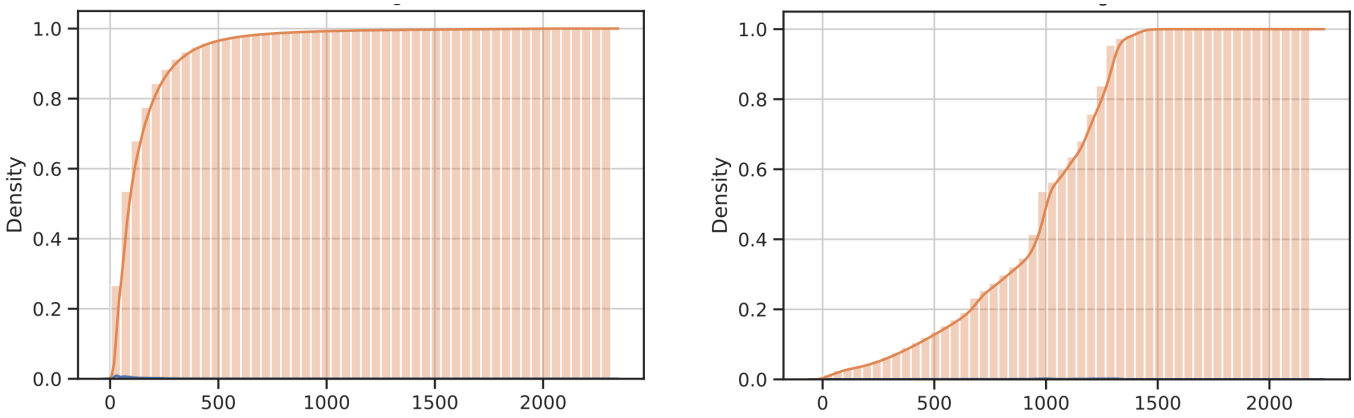

Figure 11: Cumulative distribution of heights (left) and widths (right) of paragraph blocks in our dataset

A.3 Font Modality

We illustrate in Figure 12 the output of pdfalto on an example PDF document, highlighting the font sequence information.

B Extra Material for Section 4 (Multimodal and Sequential Models)

Comparison to SOTA models (in Doc AI). LayoutLM's different versions use coordinate information at the token level after OCR (see Figure 2 of the LayoutLM paper), enabling token-level classification or single-label assignment to a page's content. This approach can't natively label specific paragraphs based on logical structure from Grobid. For tasks available with LayoutLM, see available model heads in LayoutLM's HuggingFace API documentation.

Donut and Nougat employ an encoder–decoder architecture (mainly for generative or Q&A tasks), while our approach is encoder-only (assigning labels to fixed paragraphs). This makes direct comparison challenging due to the generative nature of decoder models. We compare our models with Hierarchical Attention Transformer (HAT), specifically suited for long documents, as they are encoder-only and can label paragraphs effectively. We also provide HATs with multimodal capability.

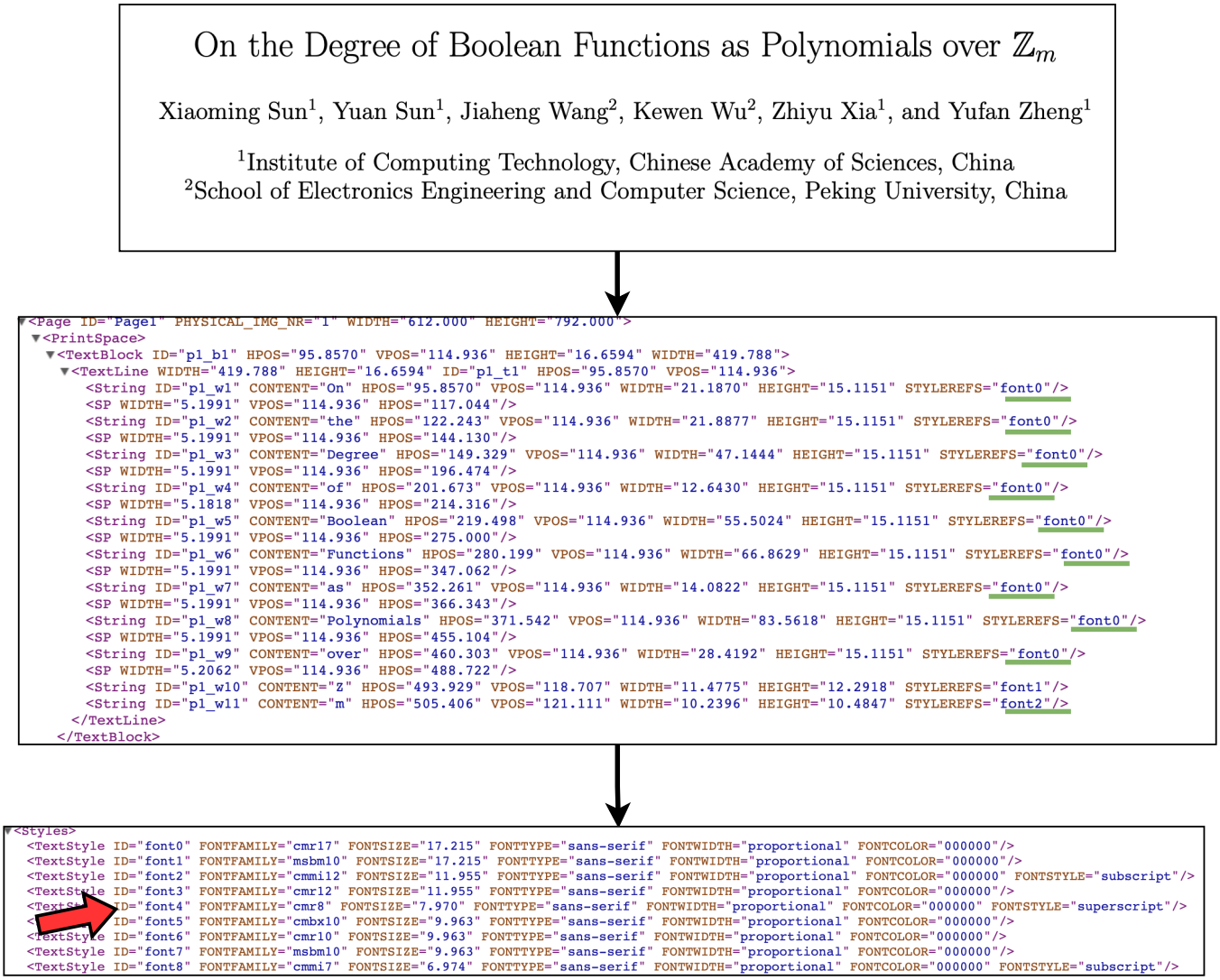

Figure 12: Font information as presented in the output of the pdfalto tool. Note that, for instance, the $\mathbb{Z}$ or m characters of the title are written in different fonts from the other tokens.

Details on LayoutLM comparisons to baselines. In the LayoutLM study, Table 5 showcases comparisons with various baselines for document classification tasks, many of which are 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 pattern of comparison 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 basic 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) closely approaches that of LayoutLM (94.42). 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 stronger baseline architectures for comparison?

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 not only enhance performance but also 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.

Details on transformer architecture for sequential approach. Our investigation leverages transformer-based BERT like architecture to process multimodal features (1280 +6 layout features), enhancing the traditional model to support the complex structure of PDF documents in our dataset. By adjusting the "maxlen" parameter, our approach accommodates very large document lengths (see data distribution, Figure 13), significantly exceeding standard transformer configurations. This modification involves splitting longer documents and integrating layout embeddings, including page information, to enhance the attention mechanism across paragraphs, thus addressing the challenge of extensive document lengths.

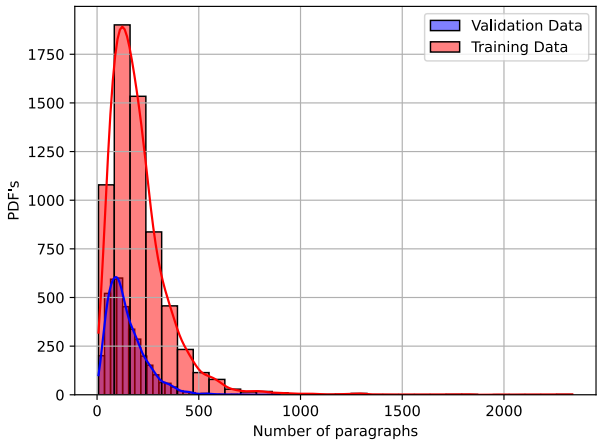

Figure 13: Overview of dataset distribution (number of paragraphs in each PDF in the training and validation data)

Empirical evidence suggests that dividing PDFs into smaller segments or applying a focused attention mechanism on a small window substantially improves model generalization (see Tables 14 and 18). Contrary to initial expectations, increasing the encoder's complexity did not proportionally enhance performance (see Table 16). A strategic reduction in the feedforward network's dimensionality, from the suggested fourfold in BERT paper to 1.5 times, not only elevated accuracy but also streamlined the model's architecture, (see Table 17), demonstrating efficiency gains alongside performance improvements.

In response to the discerned improvement in transformer models when handling shorter data segments, our research pivots towards employing a sliding window mechanism (as described in Figure 4). This method processes sequences of uniform size, applying padding as required, and functions via a non-overlapping approach, thereby enhancing computational efficiency by lowering the complexity of token attention from $O(N^2)$ to $O(N \times k)$ where k denotes the window size. This shift not only facilitates data preprocessing by removing the need for manual data segmentation but also introduces window size as a pivotal hyperparameter, substantially improving performance across diverse "maxlen" configurations. The implementation of this sliding window technique necessitated modifications in training duration; specifically, training epochs were doubled for each incremental doubling of "maxlen", maintaining consistent performance across varying data lengths. This strategic adaptation ensures model robustness and accuracy, even with extended document paragraphs (see Table 20). We also make comparisons with Hierarchical Attention Transformers but do not find any significant performance gains (see Tables 11 and 10) over a simple sliding-window transformer.

Hierarchical Attention Transformers. The intricate process of sequentially labeling paragraphs benefits from the nuanced understanding of adjacent textual relationships, a task adeptly managed by the sliding-window transformer model. Nevertheless, this approach may not fully account for long-term dependencies and interactions across windows, elements crucial for comprehensive document classification tasks where understanding global document context is paramount. To navigate these complexities, models like RoBERT/ToBERT [PZV+19], Longformer [BPC20], and BigBird [ZGD+20] have been developed, specifically designed to bridge this gap. RoBERT/ToBERT enhances the sliding window framework with additional layers to capture wider textual relationships, while Longformer and BigBird refine the attention mechanism to balance local and global textual insights effectively.

Building on these advancements, the Hierarchical Attention Transformer Network (HAT) [CDF+22] employs a layered approach, utilizing transformer encoders to forge a deeper connection between separated windows. Diverging from Longformer and BigBird's emphasis on modified attention mechanisms, HAT leverages a series of encoder blocks to methodically process multimodal features across stacked sliding windows, thus addressing long-term dependencies more comprehensively.

In our exploration, HAT model integrates two distinct types of encoder blocks: the Sliding Window Encoder (SWE) for encoding within-window modal information and the Context-Wise Encoder (CWE) for bridging content across windows. While SWE hones in on local attention, CWE extends its reach to encompass a broader context, employing an architecture designed to facilitate cross-window communication.

Table 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 20)

Window Size	Params (M)	Train Loss	Accuracy (%)	Mean F_1 (%)
16	47	0.3465	86.20	85.80
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 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 20)

Window Size	Params(M)	Train Loss	Accuracy (%)	Mean F_1 (%)
16	74	0.2996	86.01	85.15
32	74	0.2917	86.43	85.78
64	74	0.2758	86.70	86.05
128	74	0.2871	86.62	85.90
256	74	0.4460	80.45	78.96
512	74	0.5304	76.95	75.46

Despite the potential of HAT in enhancing model performance through structural and attentional depth, our investigations reveal that, within our multimodal framework, the added complexity of HAT does not unequivocally translate to superior performance over simpler sliding window constructs, highlighting a nuanced balance between architectural innovation and task-specific efficacy. We implement the interleaving architecture (as denoted in Figure 3b of HAT paper [CDF+22].)

To investigate the interleaving HAT we operate on variable hierarchical window sizes similar to the parameter sliding window (see Tables 10 and 11)

C Extra Material for Section 6 (Experimental Results)

C.1 Experimental Results

We show how different models (across different modalities) scale performance with the increasing data in Figures 14, 15, and 16, respectively for the text, vision, and font models.

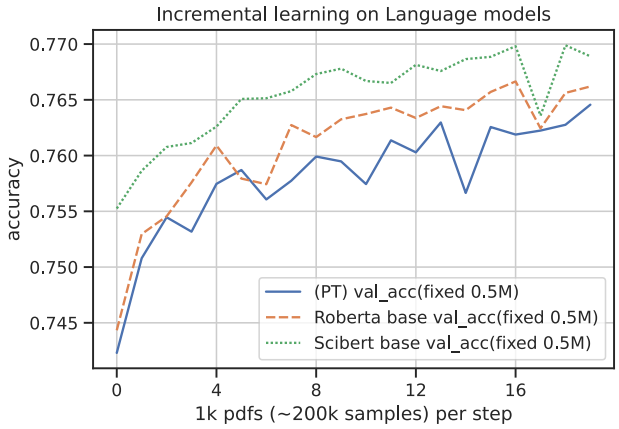

Figure 14: Accuracy of language models on fine-tuning task with respect to number of batches

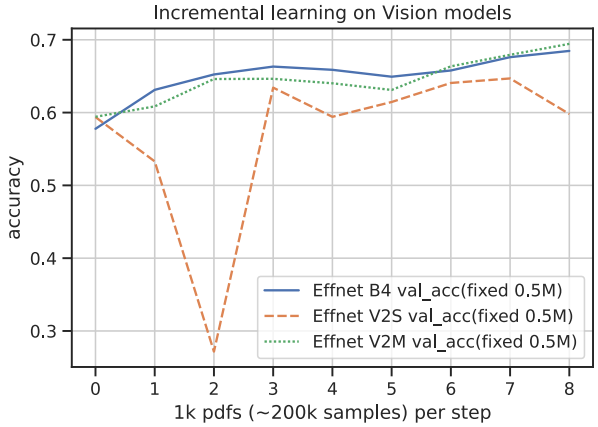

Figure 15: Accuracy of Vision models on fine-tuning task with respect to number of batches

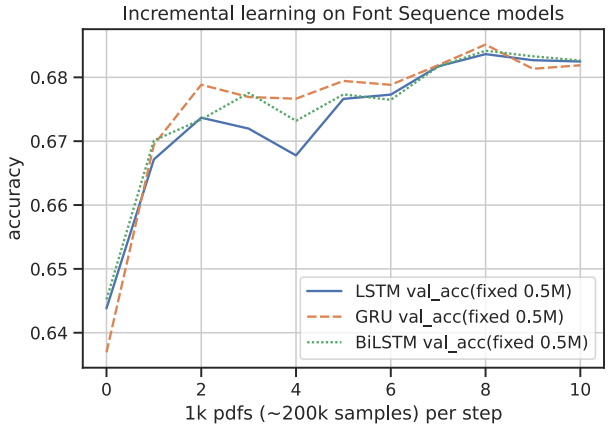

Figure 16: Accuracy of Font models on fine-tuning task with respect to number of batches

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*

$$N_k \geqslant \Omega\left(\frac{nk\log n}{\log k}\right).$$

Доказательство. Указанную оценку можно получить оценив сложность покрытия околонулевых точек функции принимающей нулевые значения на одном из слоев булева куба. ■

Figure 17: Excerpt from an example Russian-language paper [GM15] with high performance of font model

Despite the low score obtained by the font models, they can still be of use in certain situations. For example, anecdotally, Figure 17 shows an example of a Russian-language article whose blocks are correctly classified by the font-based model, while the text model is not able to use any clues as it was trained on English text.

C.2 Hyper parameter tuning of SW Transformer

In this section, we present our investigation into the hyperparameters for our sliding window (SW) transformer model. Initially, we examined a configuration akin to the original BERT architecture, featuring 16 attention heads, and varied the maximum sequence length (maxlen)[7]. Our findings, documented in Table 13, indicate that a larger maxlen detrimentally affects model performance. Consequently, we established a maxlen of 256 and proceeded to experiment with varying the number of attention heads, as detailed in Table 13. Further experimentation was conducted with even smaller maxlen values while maintaining 20 attention heads, a configuration derived from Table 12.

Subsequent investigations focused on the effect of varying the number of hidden units in the feedforward network and the number of encoder stacks, with results presented in Tables 15 and 16, respectively. Based on the insights gained from Table 15, we developed a more parameter-efficient encoder block configuration, employing 1.5 times the number of feedforward units, which demonstrated superior performance and efficiency compared to the 4x setting recommended in the original BERT paper, as shown in Table 17.

Additionally, we incorporated a sliding window mechanism into the model, as outlined in Table 18, and conducted a detailed exploration of window sizes, documented in Table 19. Upon identifying an optimal sliding window size, we increased the number of training epochs to offset the impact data size when using smaller maxlen inputs, with the outcomes reported in Table 20.

Table 12: Impact of large Max Length on Transformer Model

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 13: Impact of Attention Heads (with maxlen = 256) based on results from table 12

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

[7]It is important to note that if a document exceeds the specified maxlen, it is divided into two separate segments, each treated as an independent document with padding added to maintain dimensional consistency.

Table 14: Impact of small Max Length on Transformer Model (with heads=20) based on results from table 13

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 15: Impact of different ff-dim multipliers on Transformer Model (with maxlen=16, heads=20) based on results from table 14, 13)

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 16: Impact of Encoder Blocks (with maxlen=16, heads=20, ff-dim=1× on Transformer model based on results from table 15, 13 , 14)

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 17: 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×	0.3890	20.34M	85.17	85.01
Efficient ff-dim=1.5×	0.3864	12.15M	85.63	85.25

Table 18: Impact of SW mechanism of (window size=16) applied to encoder architecture found in table 17

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 19: Impact of Window Size (Maxlen =32) on Transformer model based on the results from table 18)

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 20: Increasing the number of Epochs to counter smaller maxlen (SW=16) based on the results from table 19

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

C.3 Interpretability

Though our models are not directly interpretable, we can get some post-hoc explanations of their performance: for the text modality, we can visualize [Vig19] the attention heads, see Figure 9 where we visualize the last layer of the language model to see what the model focuses on. Informal experiments suggest, unsurprisingly, that tokens indicative of these environments (such as the "Theorem" or "Proof" tokens, or numberings) are given a strong weight in determining the label of the paragraphs. For the vision model, we can use the Grad-CAM [SCD+17] visualization, see Figure 10, which indeed shows that some layout-based information is captured by the model.

References for the Appendix

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[BPC20] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[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.

[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.

[DPR19] Tyler Dauphinee, Nikunj Patel, and Mohammad Rashidi. Modular multimodal architecture for document classification. *arXiv preprint arXiv:1912.04376*, 2019.

[GM15] Sergey Granin and Yura Maximov. Average case complexity of DNFs and Shannon semi-effect for narrow subclasses of boolean functions. *arXiv:1501.03444*, 2015.

[HBM+22] 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.

[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.

[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.

[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.

[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.

[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.

[SZ15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[TL19] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[TL21] Mingxing Tan and Quoc Le. EfficientNetv2: Smaller models and faster training. In *ICML*, 2021.

[TLI+23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Roziè re, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[Vig19] Jesse Vig. A multiscale visualization of attention in the transformer model. In *ACL*, 2019.

[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.

[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.

[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.

[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.