Towards Extraction of Theorems and Proofs in Scholarly Articles

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

Lucas Pluvinage
lucas.pluvinage@ens.psl.eu
DI ENS, ENS, CNRS, PSL University
& Inria
Paris, France

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

ABSTRACT

Scholarly articles in mathematical fields often feature mathematical statements (theorems, propositions, etc.) and their proofs. In this paper, we present preliminary work for extracting such information from PDF documents, with several types of approaches: vision (using YOLO), natural language (with transformers), and styling information (with linear conditional random fields). Our main task is to identify which parts of the paper to label as theorem-like environments and proofs. We rely on a dataset collected from arXiv, with LaTeX sources of research articles used to train the models.

CCS CONCEPTS

• **Information systems → Digital libraries and archives**; • **Computing methodologies** → *Artificial intelligence*.

KEYWORDS

information extraction, scholarly articles, theorems, proofs

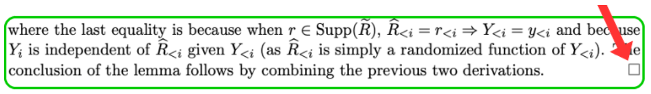
ACM Reference Format:
Shrey Mishra, Lucas Pluvinage, and Pierre Senellart. 2021. Towards Extraction of Theorems and Proofs in Scholarly Articles. In *ACM Symposium on Document Engineering 2021 (DocEng '21), August 24–27, 2021, Limerick, Ireland*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3469096.3475059

1 INTRODUCTION

Mathematical statements (theorems, propositions, definitions, etc.) and their *proofs* form some of the most important parts of scholarly articles in mathematical fields (e.g., mathematics, theoretical computer science, mathematical physics), and are usually of independent interest for scientists. However, scientific articles are often only available as PDF documents with no specific structure outlining statements and their proofs. We deal in this paper with the automatic extraction of such information.

We rely on machine learning techniques, and in this preliminary work explore several different approaches, based on computer vision, natural language, and document styles. The task is to automatically extract mathematical statements (generically referred to in the following as *theorems*) and their proofs and label them.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
DocEng '21, August 24–27, 2021, Limerick, Ireland
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8596-1/21/08…$15.00
https://doi.org/10.1145/3469096.3475059



where the last equality is because when $r \in \mathrm{Supp}(\widetilde{R})$, $\widehat{R}_{<i} = r_{<i} \Rightarrow Y_{<i} = y_{<i}$ and bec use Y_i is independent of $\widehat{R}_{<i}$ given $Y_{<i}$ (as $\widehat{R}_{<i}$ is simply a randomized function of $Y_{<i}$). e conclusion of the lemma follows by combining the previous two derivations. $\square$

Figure 1: Example proof excerpt, end-of-proof (QED) symbol

There is a rich literature on information extraction from scholarly articles [8, 15] though none on extraction of theorems and proofs.

One of the closest work might be [6] which identifies tables, lines, and formulas formulated as an object detection task using CNNs and a hybrid CRF model. This approach faces two major challenges: first, structured ordering of elements in scholarly articles is different from regular images that are the use case of popular object detection frameworks such as YOLO [11] or Faster R-CNN [12], with in particular a smooth transition of pixel values around objects; second, spacing between objects in an article vary. For the specific task of table detection, it appears that vision-based approaches can nevertheless be very successful, as demonstrated using YOLO v3 [6] and masked R-CNNs [10] with nearly perfect F_1 scores on the ICDAR2013 and 2017 datasets.

In our setting, however, theorems and proofs are often differentiated by styling information such as special fonts or special characters such as the QED symbol at the end of the proof (see Figure 1), which are hard to capture using vision-based techniques. Object detection techniques are also poorly adapted to the understanding of the sequencing of text and mathematics, which has a crucial impact on their semantics. Still, as a representative of a vision-based technique, we employ in this work YOLO v4 [3] which shows significant performance gains over YOLO v3.

We also consider approaches based on natural language processing, in particular attention-based mechanisms readily exploited by transformer language models. These language models help to understand context information, including for longer text lengths, which can be lost by simpler models relying on, e.g., LSTM networks [1]. The language models that we do use in this work (namely, BERT [5], DistilBERT [14], and SciBERT [2]) have been pre-trained to understand the general language rules (English here). SciBERT, in particular, has been pretrained on 1.14 million papers in the scientific corpus (18% from computer science, 82% from biomedicine), which may make the vocabulary learned more useful for our task than that of language models trained on other corpora; SciBERT [2] shows for instance better performance in sentence classification over scientific articles (ACL-ARC) in Computer Science than BERT. All these language models can be *fine-tuned* for a specific task, which is what we do in this work.

Even though language models can retain semantic information, they lack styling-based information. For example, knowing that a line starts with a "Theorem" word in bold is a very strong indicator that the line is the start of a theorem. One work that uses such information is [13], where a deep learning network is trained on features built out of the content and style of scientific articles to extract metadata of algorithms contained in the paper, reaching 78.5% accuracy (using RMSprop). In the same line, we also explore in this work a classifier trained from simple content and style-based features. Specifically, we use a conditional random field (CRF) classifier to exploit the dependency of labeling from one line to the next, without requiring too many hyperparameters as with LSTMs.

We present in Section 2 our three different approaches, then the dataset and experiments in Section 3 before discussing the results and potential for future research in Section 4. All code and a description on how to acquire the data used in this article are made available at https://github.com/PierreSenellart/theoremkb.

2 METHODOLOGY

Collecting and Labeling Data. Since there is no publicly available dataset for our task of interest, we collected publicly available articles from arXiv, using arXiv's bulk data access[1]. To automatically label this corpus, we focused on articles for which a LaTeX source is available (the vast majority of articles on arXiv in mathematical sciences, see [9]). Indeed, when such source material is present, and when this source material uses standard LaTeX practices (`\newtheorem` to define theorem-like environments as well as the `proof` environment), it is possible to automatically label lines belonging to theorems and proofs by instrumenting a LaTeX run and producing PDF annotations corresponding to these labels. We use this as a way to obtain ground truth, however this might not be completely accurate if the source code contains a different way of initializing proofs/theorems see [9] for more details; on the other hand, all information extraction techniques work on the unannotated PDF version of the papers, without access to the source material, which reflects the fact that LaTeX sources are rarely available outside arXiv.

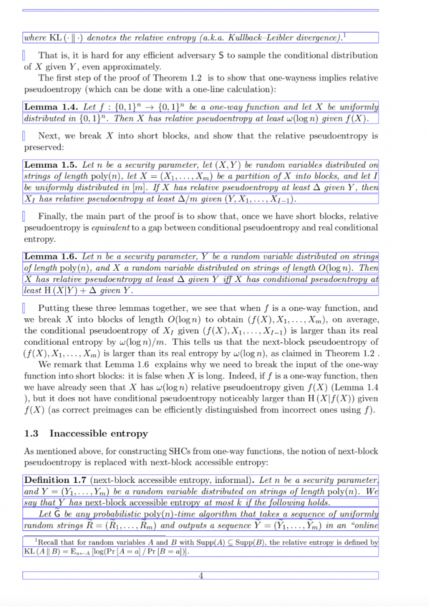
Some post-processing is required on the result of the LaTeX-based extraction of theorems and proofs (see Figure 2), in particular to filter out footers or empty annotation boxes which are an artifact of the LaTeX instrumentation. This is currently done with a hard-coded threshold that discards boxes with textual content such as page numbers, empty boxes generated (due to the rendering process) see Figure 2.

Depending on the approach used, PDF articles are then either converted to bitmap images (for vision-based approaches) or processed using pdfalto[2], which turns a PDF document into an XML file formed of a spatial index of all characters of the document organized into hierarchical blocks of text (pages, vertical blocks of lines, lines) along with styling information, as well as annotations (hyperlinks, etc.), as shown on Figure 3.

Vision-Based Approach. Each page of the PDF image is rendered as a PNG bitmap image with line-based annotations of the ground truths corresponding to boxes. We then feed these bitmap images with (upscaled) boxes labeling objects corresponding to theorems and proofs to YOLOv4.

Previous versions of YOLO used Darknet53 as backbone network along with residual layers after every few blocks of convolution

[1]https://arxiv.org/help/bulk_data
[2]https://github.com/kermitt2/pdfalto



where KL $(\cdot \,\|\, \cdot)$ *denotes the relative entropy (a.k.a. Kullback–Leibler divergence).*[1]

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

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

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

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

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

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

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

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

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

1.3 Inaccessible entropy

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

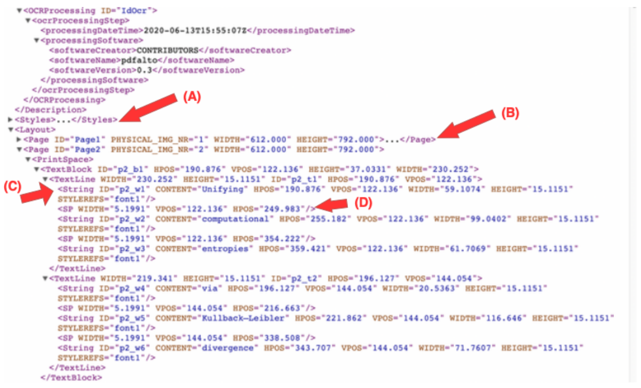
Definition 1.7 (next-block accessible entropy, informal). *Let n be a security parameter, and $Y = (Y_1,\dots,Y_m)$ be a random variable distributed on strings of length* poly(n)*. We say that Y has* next-block accessible entropy *at most k if the following holds.*

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

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

4

Figure 2: Raw annotations resulting from the LaTeX processing of theorems and proofs



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

Figure 3: Result of the processing by pdfalto: (A) style block (information about fonts); (B) page block; (C) character string; (D) spacing information

operations alongside batch normalization after every layer to combat gradient vanishing and internal covariance shift in the network. However, in YOLOv4 the authors have identified and grouped a range of different techniques that significantly increase the performance of the network into categories described as either **bag of freebies** (Cutmix, DropBlock, Mosaic Data Augmentation, label smoothing): applying these techniques increase the mAP of the model without compromising on the inference time; or **bag of specials** (Mish activation, Cross Stage Partial Connections, multi weighted input residual connections): these techniques increase the accuracy of detection with a small increase in the inference time.

The Intersection-over-Union (IoU) threshold kept for the selection of the boxes is 0.7. The problem is posed as a binary classification problem dealing with the object of interest in either proofs or

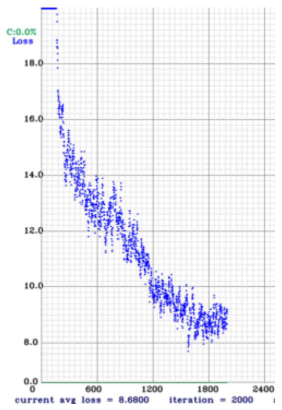


Figure 4: Loss function after 2000 iterations of YOLOv4

theorems if there exists a region of interest. The loss function can be visualized in Figure 4 after training 2000 iterations of YOLOv4.

Language-Based Approaches. Textual information from each of the lines is extracted and prepared for a binary classification task of sequence prediction. We access and evaluate the performance of several autoencoding-based models such as Bert base [5], DistilBERT base [14] and SciBERT [2]. These autoencoding-based models are trained using a word masking mechanism (masking 15% of the tokens), which we believe are well adapted to the task of theorem and proof identification.

The success of transformer-based models is heavily dependent on many crucial factors, some of them being: the quantity of data used for pretraining the model, which can be extremely large for, e.g., BERT; the number of data samples fed in for finetuning the model; the nature of content provided (e.g., SciBERT was specifically trained on a scientific corpus of papers, mostly in the biomedical domain); the size of the Encoder–Decoder stacks (BERT comes in three variants: base/small/large); the type of pretraining used (e.g., GPT-based models see everything before the next token as part of pretraining thus making them relevant for text generation tasks).

To contrast with these language models, we also trained a simple LSTM model based on parallel positional encoding, using 100 cells. This model serves as a baseline to evaluate the transformer-based models in the task of identifying proofs and theorems.

Styling- and Sequence-Based Approaches. From the pdfalto output we extract features that describe the document at different hierarchical levels. <Page>-level features simply include the global position in the document (begin, inside, end). <TextBlock> also has spacing information: the distance with the last and next blocks are computed, as well as the horizontal paddings. <TextLine> features are used to detect the presence of tabulations (horizontal spacing from the side of the container block) and patterns (a pre-existing sequence of text, meaning that it occurred often in the document, such as in the header or the footer section). <String> features include the normalized word, word-wise spacing information, detect if the font is in italic or bold, font size, word-position in the sentence, and the presence of digits and special characters. This small set of features is normalized and standardized across the document so that the model can focus on *changes* instead of *values* and become robust to global style changes. We also expand the features by computing for each token the difference between the current value and the previous/next one: this is important to detect local changes. Then, for a target *resolution* we derive the linear sequence of features for the chosen document, meaning that coarser-grained features are simply copied for each item (i.e., for each line in a page, the page-level features are copied along with the individual line-level features) and finer-grained features are aggregated using the following methods: minimum value, maximum value, first value, last value, mean value.

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

Finally, we also implemented a simple baseline that used some simple information from the document, along with sequential information, referred to in the following as the *naïve* algorithm: simply look for a set of target words ("Proof", "Theorem", etc.) in bold or italic at the beginning of a line and set the label of this line and all subsequent lines until the end of a "block" accordingly.

Table 1: Inference time

Approach	Training time (h)	Model size (MB)
YOLOv4	6.42 (2000 Epochs)	244
LSTM	0.05 (4 Epochs)	19.9
LSTM (unbalanced)	0.12 (4 Epochs)	19.9
DistilBERT-base cased	3.47 (1 Epoch)	263.3
SciBERT-base cased	6.07 (1 Epoch)	440.1
BERT-base cased	6.27 (1 Epoch)	438.2
DistilBERT-base uncased	3.44 (1 Epoch)	263.3
SciBERT-base uncased	4.24 (1 Epoch)	440.1
BERT-base uncased	6.23 (1 Epoch)	438.2
Linear-chain CRF	5 (300 iterations)	0.24

All trainings were made on a Tesla P100 GPU on Google Colaboratory with 15 GB of memory except for the CRF, trained on commodity hardware.

3 DATASET AND EXPERIMENTS

The dataset[3] is formed of 4,400 PDF articles from the CS-CC category of arXiv (namely, Computer Science – Computational Complexity) category from 2010 to 2020.

In experiments, we compare the training time (see Table 1) and performance (see Tables 2, 3, 4) of our various approaches.

We stress that those are preliminary experiments with the goal to simply explore what kind of performance levels can be reached by straightforward methods. In particular, note that none of the approaches based on computer vision or natural language can capture real sequence-based information (i.e., that the label of a line is heavily correlated with that of the previous and the following line) but attempt to label visual objects or individual lines separately. Also, note that each class of approaches exploits a different segment of

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

Table 2: Performance: Computer vision

Approach	Loss	mAP/Accuracy
YOLOv4	8.68	0.416

Table 3: Performance: Natural language

Approach	Loss	mAP/Accuracy
LSTM	0.9620	0.4625
LSTM (unbalanced)	1.2571	0.4183
BERT-base cased	0.8232	0.6316
SciBERT-base cased	0.8165	0.6327
DistilBERT-base cased	0.8267	0.6302
Bert base-uncased	0.8222	0.6313
SciBERT-base uncased	0.8151	0.6357
DistilBERT-base uncased	0.8275	0.6287

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

Table 4: Performance: Styling

Approach	Precision	Recall	F_1
Linear-chain CRF	0.800	0.834	0.816
Naive Algorithm	0.832	0.370	0.487

understanding to decide on labels. A robust classifier would make a cumulative decision based on all features including vision, text, and styling information while also utilizing the sequence information across multiple lines.

4 DISCUSSION & FUTURE WORK

Discussion of results. By far the most effective approach tested in this work is training the linear-chain CRF classifier, even though this approach only utilizes sequence-based styling information at every line level. Results from Table 4 clearly indicates that styling-based semantics such as the font used and the ratio of bold or italic characters are alone a very strong indicator along with other styling-based features and sequence information. Another strong indicator of this fact is the not-too-bad performance of the naïve algorithm.

It may be surprising to see (Table 3) that language models such as Bert and SciBERT taking only information at an individual line level and disregarding any information about lines in the sequence can maintain a validation accuracy around 0.63 on undersampled data. Their main disadvantage however is the huge training time and space associated (see Table 1). A popular alternative we explored in this paper is DistilBERT [14] which is 40% smaller in size while retaining 97% of its language processing capabilities and being 60% faster than Bert. In our model experimentation, we find very similar performance while detecting theorems and proofs to Bert-based models. *Casing* (distinguishing between lowercase and uppercase characters) seems to have little effect. Usage of the *Scibert specific vocabulary* generally improves the performance on scientific corpora, though this is fairly marginal even though special vocabulary adds an extra layer of information for the classifiers. Due to the large amount of training time required for each model, a single epoch of fine-tuning was run; continuing the evaluation on more epochs could improve accuracy.

The least desirable results are generated from using a computer-vision based approach; though raw results as observed by a human being are sometimes acceptable, these techniques suffer from the evaluation metric, where the IoU score plays an important role in determining the mAP; for the line identification task, this means precisely identifying the exact edges around proofs and theorems.

Perspectives. This work is an initial step made in the direction of extracting mathematical structures from scientific articles in PDF form. We plan to perform a more comprehensive study investigating the following future directions leading to potential performance improvements: Capturing sequential information based on several text lines instead of considering independently every individual line; Cumulatively building classifiers on the features extracted by each of the three approaches; Further training models for a large number of epochs to ensure convergence. Finally, note that, in vision-based approaches, building ensembles [4] of several different types of techniques each acting as an individual weak learner can enhance mean average precision significantly especially with transformer-based vision techniques [7] topping the leaderboards on object detection datasets.

ACKNOWLEDGEMENTS

This work has been funded by the French government under management of ANR as part of the "Investissements d'avenir" program, ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

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

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. YOLO v4: Optimal Speed and Accuracy of Object Detection. *CoRR* 2004.10934 (2020).

[4] Ángela Casado-García and Jónathan Heras. 2020. Ensemble methods for object detection. In *ECAI*.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

[6] Xiao-Hui Li, Fei Yin, and Cheng-Lin Liu. 2018. Page object detection from PDF document images by deep structured prediction and supervised clustering. In *ICPR*.

[7] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR* 2103.14030 (2021).

[8] Patrice Lopez. 2009. GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In *TPDL*.

[9] Lucas Pluvinage. 2020. *A knowledge base of mathematical results.* Master's thesis. ENS, PSL University. https://hal.inria.fr/hal-02956526

[10] Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultanpure. 2020. CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents. In *CVPR Workshops*.

[11] Joseph Redmon and Ali Farhadi. 2018. YOLOv3: An incremental improvement. *CoRR* 1804.02767 (2018).

[12] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2017. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 6 (2017).

[13] Iqra Safder, Saeed-Ul Hassan, Anna Visvizi, Thanapon Noraset, Raheel Nawaz, and Suppawong Tuarob. 2020. Deep Learning-based Extraction of Algorithmic Metadata in Full-Text Scholarly Documents. *Information Processing & Management* 57, 6 (2020), 102269.

[14] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR* 1910.01108 (2019).

[15] Kyle Williams, Jian Wu, Sagnik Ray Choudhury, Madian Khabsa, and C. Lee Giles. 2014. Scholarly big data information extraction and integration in the citeseer χ digital library. In *ICDE workshops*.