FinTOC-2019 Shared Task: Finding Title in Text Blocks

Hanna Abi Akl
Yseop
habi-akl@yseop.com

Anubhav Gupta
Yseop
agupta@yseop.com

Dominique Mariko
Yseop
dmariko@yseop.com

Abstract

As part of FNP Workshop Series, "Title Detection" is one of the two shared tasks proposed on Financial Document Structure Extraction. The objective of the task was to classify a given text block, that had been extracted from financial prospectuses in pdf format, as a title. Our DNN-based approach scored a weighted F1 of 97.16% on the test data.

1 Introduction

The Portable Document Format, also known as pdf, is an electronic document format from Adobe Inc. Launched in early 1990s, this format has now become the de-facto means of sharing information across the Internet. However, given the lack of "basic high level logical structure information" (Chao and Fan, 2004), the process of layout and content extraction from a pdf is difficult. The obstacles in extracting information from a pdf, as enumerated by Hu and Liu (2014), are:

- absence of information regarding structure
- disagreement of the render order with the "reading order"
- overlap of object blocks
- myriad layouts and fonts

Consider the process of automating the information extraction from financial documents. Not being able to recognize where a paragraph begins or ends in a financial report can be a strain on many levels: a) it not only blurs important information, but b) can also be misleading in decisions taken based on said report. Though a number of open-source and commercial tools are available, the goal of establishing correctly a semantic unit (paragraphs, tables) and ascertaining its role (title, header) is far from being complete (Bast and Korzen, 2017).

The **FinTOC-2019 Shared Task: "Financial Document Structure Extraction"** (Juge et al., 2019) of **Financial Narrative Processing Workshop** comprises two shared-tasks:

- Title detection
- TOC structure extraction

In this paper, we propose and evaluate three methods to detect titles (Shared Task 1). Since text comes in many different formats, this problem can become exponentially heavy to treat. For that purpose, our experiment is concerned with identifying only titles in pdf reports.

2 Experiments

Our first approach was to use a standard SVM classifier (Cortes and Vapnik, 1995) to understand the data and evaluate their non-linearity. The second and third approaches are based on deep learning classifiers. The second design is inspired by a BiLSTM recurrent neural network (RNN) model with attention (Zhou et al., 2016). The third model is a convolutional neural network (CNN) with character embedding (Zhang et al., 2015).

2.1 Data

The training data was extracted from 44 documents using the Poppler utility libraries (poppler-utils) and converted into xml files. The contents of the xml were transformed into a csv file with the help of various heuristics by the organisers. The resulting file had 75 625 text blocks and 7 features describing each of them. Since these features were automatically generated they were noisy. For example, some of the text blocks that begin with determiner such as 'a', dates

Proceedings of the Second Financial Narrative Processing Workshop (FNP 2019), pages 58–62

and addresses were erroneously marked as *begins_with_numbering*. Even though the original pdf files were provided we decided to work only with the provided xml and csv files because a) the result needed to be submitted in the provided csv format and b) the heuristic to transform xml into text blocks (as presented in csv) was unknown.

2.2 Approach

The first model we evaluated was a SVM classifier. It was trained on a combination of features, compiled from the existing features presented within the original csv file as well as additional features extracted from pre-processing work on the xml files (similar to format and linguistic features of Hu et al. (2006)). These features capture layout and form of the text, which can play a deciding part in the classification. In addition to the provided features, we computed:

1. **top:** integer indicating the placement of the text with respect to the top of the document page
2. **left:** integer indicating the placement of the text with respect to the left of the document page
3. **width:** integer indicating the horizontal space occupied by the text
4. **height:** integer indicating the vertical space occupied by the text
5. **font:** integer indicating the size of the text
6. **number of dots:** integer indicating the total number of dots in the text
7. **is last character dot:** binary indicating if the text ends with a dot (1) or not (0)
8. **count of capital letters:** integer indicating the total number of capital characters in the text
9. **character count:** integer indicating the total number of characters in the text
10. **word count:** integer indicating the total number of words in the text
11. **average word length:** float indicating the average number of characters per word in the text
12. **number count:** integer indicating the total number of digits in the text
13. **count of words in capital:** integer indicating the total number of capitalized words in the text
14. **similarly avg word length:** integer indicating average token length
15. **cnn output** binary prediction provided by the CNN model (described later).

The observed variance and results were very close with and without the CNN outputs added as input features, so after evaluation of the noise in the features (as predictors), we decided to foster two deep learning approaches based on raw text entries and to focus on regularisation methods to prevent high variance observed in the SVM results.

The second model is a BiLSTM–Attention model relying on word embedding. It has 2 dense layers, each composed of 64 neurons and is deployed with a batch size of 256 and 100 epochs. The purpose of this method is to evaluate the possible semantic composition of sentences (as predictors) and how relevant they are for the task. It contains five components:

1. **Input layer:** inputs text to the model
2. **Embedding layer:** parses text into words and maps each word into a low dimension vector
3. **LSTM layer:** makes use of BiLSTM to get high-level features from the previous step
4. **Attention layer:** produces a weight vector, and merges word-level features from each time step into a sentence-level feature vector, by multiplying the weight vector
5. **Output layer:** uses the sentence-level feature vector for classification

The aim of this architecture is to make use of the attention mechanism, which can automatically focus on the words that have a decisive effect on classification (in this case, the heavy constituents of a title), to capture the most important semantic information in a text block, circumventing the question of noise in the feature set. The

Experiment	Model								
	BiLSTM-ATTENTION			CNN			SVM		
Hardware	Intel Core i7 2.20GHz 16GB RAM NVIDIA GeForce GTX 1050 Ti 4Go			Intel Core i7 2.20GHz 16GB RAM NVIDIA GeForce GTX 1070 8Go			Intel Xeon 3.70GHz 64GB RAM 8 CPU cores		
	F1 (home) (%)	F1 (lboard) (%)	RunTime (hrs)	F1 (home) (%)	F1 (lboard) (%)	RunTime (hrs)	F1 (home) (%)	F1 (lboard) (%)	RunTime (hrs)
Experiment 1	96.02	91.24	7-9	-	-	-	95.96	91.13	2-4
Experiment 2	94.04	93.18	7-9	95.03	97.16	1-2	-	-	-

Table 1: Comparative table of results provided for leaderboard

hyper-parameter selection follows the findings of (Zhou et al., 2016) for the neural network. The number of layers, batch size, number of epochs, and learning rate have all been picked from a referenced model which achieved competitive results against state-of-the-art networks. The sequence length has been chosen to match the number of characters in the longest sentence. We also used dropout to regularize the network and alleviate over-fitting. Between experiments 1 and 2, we increased the number of hidden units from 32 to 64 to improve the accuracy of the model.

The third model is a CNN classifier. The purpose of this method is to evaluate the combinatorics of characters at word level (as predictors) and how relevant they are for the task. A state-of-the-art competitive word-level character embedded convolution network is used. The model follows the conclusion of Zhang et al. (2015) for tackling small dimensional problems: integrating both upper-case and lower-case letters in set of predictors might improve the results in case the data set is small (AG news case). This consideration was reinforced by the intuition that identifying both small and capital characters was relevant for the Title detection task, which was confirmed by comparing between the results driven from differentiating between upper and lower cases and the those driven by lower-case only. From a dictionary of all characters included in the training set, a character embedding operation is performed on all sentences. Word-level character embedding size of 175 is chosen for this transformation and sentences are turned into a matrix of embedded characters. This matrix is then fed to the CNN, which has the following characteristics:

1. **Convolution layers**: 2D convolution (4 convolution layers)

2. **Pooling layers**: 1D convolution reducing dimensions

3. **Fully connected layers**: transformation providing the prediction from softmax probabilities.

4. **Parameters selection**: In order to reduce the variance while training, we used a standard dropout of 0.5 at each epoch. Augmenting or reducing the dropout has showed worse results. We also added a cross validation step from a set of 1500 individuals, evaluated every ten epoch, the best epoch in this evaluation range was then chosen to become the warm start for the next 10 epochs. The number of layers chosen is the best trade off we found between a deeper network, our GPU computational capacity and runtime. The model was trained on 100 epochs, but converges quickly towards epoch 40, which eventually provided the best results.

2.3 Evaluation

For the given training data set, approximately 1 in 6 text blocks was a title. Thus, it was a case of class imbalance. A classifier which labels every text block as 'non-title' scored a weighted F1 of **80.12 %**. Our objective was to improve this score.

The performance of the three models on train and leaderboard sets are listed in Table 1. In this Table, the scores we obtained locally (F1 (home)) are compared with the final score on competition test set (F1 (lboard)). The experiment 1 corresponds to the results sent by the first deadline and the second one refers to the extended deadline submission. As the Deep Learning models originally had been trained on different GPU, we also added a comparative table on the model with best performances (CNN) with same architecture and parameters as the one provided for leaderboard results, so the influence of the architecture is clarified (Table 2).

	CNN Model		
	AUC (train) (%)	F1 (home) (%)	RunTime (hrs)
Intel Core i7 2.20GHz 16GB RAM NVIDIA GeForce GTX 1070 8Go	91.41	95.03	1-2
Intel Core i7 2.20GHz 16GB RAM NVIDIA GeForce GTX 1050 Ti 4Go	90.99	95.01	5

Table 2: Compared results for CNN, GPU dependent

3 Related Works

Most of the literature deals with extraction of information from scientific documents since they are publicly available and in large quantity. Gao et al. (2011) exploited common typesetting practice (Style Consistency of page Components) in books to extract structural information from pdf documents. Their solution was based on weighted bipartite graphs and optimal matching based on Kuhn-Munkres algorithm. A similar approach was used by Klampfl et al. (2014) to analyse the structure of scientific articles. They defined a heading as a text block that appears, in reading order, before a **main** text block. Other defining features of a heading, according to them, are:

- starts with a number or a capital letter
- consists of at least one non-whitespace letter
- has at most 3 lines
- font size is not less that of surrounding text blocks
- distance to the text block is not more than a given level

According to Constantin et al. (2013), the differentiating feature of a title is font frequency. In other words, since titles occur less frequently in a document their font will also be rare with respect to other fonts present in the document.

Most of the works that process financial documents focus on obtaining tabular data from the files. Potvin et al. (2016) employs rectilinear search algorithm and Chen et al. (2017) makes use of rectangle mining (REMINE).

In summary, almost every approach utilizes the geometric data available in pdf files to analyse and extract their content.

4 Conclusions

We proposed three different approaches to tackle the problem of title identification. Designing and working with different architectures allows room for improvement. For the BiLSTM-Attention model, depth can be experimented with by adding more layers and invigorating existing layers with more neurons. Of course, a subliminal challenge here would be accommodating the necessary hardware and supplying enough computational power to run such model in a reasonable time. For the CNN model, there are many possibilities in experimenting by acting on each component separately and fine-tuning its hyper-parameters, trying to optimize the network for small dimensional sets. We have not tried initializing the weights with a specific distribution yet and though the number of possible convolutions is limited by our GPU capacity, we believe there is room for performance improvement optimising the convolution graphs to precisely fit the memory capacities on larger GPU. Another path would be to consider this task as a computer vision one and try the CNN to detect graphical areas related to titles from PDF images.

Finally, it would also be interesting to build on our efforts to examine and rank features by importance and study the most influential features on the title classification exclusively. That trail might bring up interesting patterns worth exploring and possibly even replicating over other NLP tasks.

References

Hannah Bast and Claudius Korzen. 2017. A benchmark and evaluation for text extraction from pdf. In *Proceedings of the 17th ACM/IEEE Joint Conference on Digital Libraries*, JCDL '17, pages 99–108, Piscataway, NJ, USA. IEEE Press.

Hui Chao and Jian Fan. 2004. Layout and content extraction for pdf documents. In *Document Analysis Systems VI*, pages 213–224, Berlin, Heidelberg. Springer Berlin Heidelberg.

Xilun Chen, Laura Chiticariu, Marina Danilevsky, Alexandre V. Evfimievski, and Prithviraj Sen. 2017. A rectangle mining method for understanding the semantics of financial tables. In *14th IAPR International Conference on Document Analysis and*

Recognition, ICDAR 2017, Kyoto, Japan, November 9-15, 2017, pages 268–273.

Alexandru Constantin, Steve Pettifer, and Andrei Voronkov. 2013. Pdfx: Fully-automated pdf-to-xml conversion of scientific literature. In *Proceedings of the 2013 ACM Symposium on Document Engineering*, DocEng '13, pages 177–180, New York, NY, USA. ACM.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Liangcai Gao, Zhi Tang, Xiaofan Lin, Ying Liu, Ruiheng Qiu, and Yongtao Wang. 2011. Structure extraction from pdf-based book documents. In *Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, JCDL '11, pages 11–20, New York, NY, USA. ACM.

Jianying Hu and Ying Liu. 2014. *Analysis of Documents Born Digital*, pages 775–804. Springer London, London.

Yunhua Hu, Hang Li, Yunbo Cao, Li Teng, Dmitriy Meyerzon, and Qinghua Zheng. 2006. Automatic extraction of titles from general documents using machine learning. *Information Processing Management*, 42(5):1276 – 1293.

Rémi Juge, Najah-Imane Bentabet, and Sira Ferradans. 2019. The fintoc-2019 shared task: Financial document structure extraction. In *The Second Workshop on Financial Narrative Processing of NoDalida 2019*.

Stefan Klampfl, Michael Granitzer, Kris Jack, and Roman Kern. 2014. Unsupervised document structure analysis of digital scientific articles. *International Journal on Digital Libraries*, 14(3):83–99.

Benoit Potvin, Roger Villemaire, and Ngoc-Tan Le. 2016. A position-based method for the extraction of financial information in PDF documents. In *Proceedings of the 21st Australasian Document Computing Symposium, ADCS 2016, Caulfield, VIC, Australia, December 5-7, 2016*, pages 9–16.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany. Association for Computational Linguistics.