UWB@FinTOC-2019 Shared Task: Financial Document Title Detection

Tomáš Hercig
NTIS – New Technologies
for the Information Society,
Faculty of Applied Sciences,
University of West Bohemia,
Technická 8, 306 14 Plzeň
Czech Republic
tigi@kiv.zcu.cz

Pavel Král
Department of Computer
Science and Engineering,
Faculty of Applied Sciences
University of West Bohemia,
Univerzitní 8, 306 14 Plzeň
Czech Republic
pkral@kiv.zcu.cz

Abstract

This paper describes our system created for the Financial Document Structure Extraction Shared Task (FinTOC-2019) Task A: Title Detection. We rely on the XML representation of the financial prospectuses for additional layout information about the text (font type, font size, etc.). Our constrained system uses only the provided training data without any additional external resources. Our system is based on the Maximum Entropy classifier and various features including font type and font size. Our system achieves F1 score 97.2% and is ranked #3 among 10 submitted systems.

1 Introduction

Financial documents are used to report activities, financial situation, investment plans, and operational information to shareholders, investors, and financial markets. These reports are usually created on an annual basis in machine-readable formats often only with minimal structure information.

The goal of the Financial Document Structure Extraction Shared Task (FinTOC-2019) (Juge et al., 2019) is to analyse these financial prospectuses[1] and automatically extract their structure similarly to Doucet et al. (2013).

The majority of prospectuses are published without a table of content (TOC), which is usually needed to help readers navigate within the document.

2 Task

The goal of FinTOC-2019 shared task is to extract the table of content from the financial prospectuses. The shared task consists of two subtasks:

[1]Official PDF documents in which investment funds precisely describe their characteristics and investment modalities.

- Subtask A classifies given text blocks as titles or non-titles.
- Subtask B organizes provided headers into a hierarchical table of content.

We participated only in subtask A. For additional information (e.g. about subtask B) see the task description paper (Juge et al., 2019).

Systems participating in this shared task were given a sample collection of financial prospectuses with different level of structure and different lengths as training data.

We approached the title detection subtask as a binary classification task. For all experiments we use Maximum Entropy classifier with default settings from Brainy machine learning library (Konkol, 2014).

Data statistics for the title detection subtask are shown in Table 1.

Label	Test	Train
Non-title	13 928 (94.0%)	65 354 (86.4%)
Title	888 (6.0%)	10 271 (13.6%)

Table 1: Data statistics for Subtask A.

3 Dataset

The provided training collection of documents contains:

- PDF format of the documents
- XML representation of the PDFs as given by the Poppler utility libraries; this representation contains the text of the documents as well as layout information about the text (font, bold, italic, and coordinates).
- CSV file with gold labels.

Label	Test	Fixed Test	Train	Fixed Train
Non-title	13 928	12 844 (92.2%)*	65 354	60 533 (92.6%)
Title	888	821 (92.5%)*	10 271	10 209 (99.4%)
Sum	14 816	13 665 (92.2%)	75 625	70 742 (93.5%)

Table 2: Comparison of datasets with fixed issues.

The XML file consists of page elements and has essentially the following structure:

```
<page number="1" ...>
<fontspec id="0" size="11"
family="Times" color="#000000"/>
<fontspec id="1" size="9".../>
<text ...><b> </b></text>
<text ...>Man Umbrella SICAV </text>
...
</page>
...
```

The CSV file contains the following fields delimited by tabs. For more details see the task description paper (Juge et al., 2019).

- Text blocks: a list of strings computed by a heuristic algorithm; the algorithm segments the documents into homogeneous text regions according to given rules
- Begins_with_numbering: 1 if the text block begins with a numbering such as 1., A/, b), III., etc.; 0 otherwise
- Is_bold: 1 if the title appear in bold in the PDF document; 0 otherwise
- Is_italic: 1 if the title is in italic in the PDF document; 0 otherwise
- Is_all_caps: 1 if the title is all composed of capital letters; 0 otherwise
- Begins_with_cap: 1 if the title begins with a capital letter; 0 otherwise
- Xmlfile: the XML file from which the above features have been derived
- Page_nb: the page number in the PDF where the text block appears
- Label: 1 if text line is a title, 0 otherwise

According to the organizers, participants can either use the segmentation into text blocks suggested in the CSV file provided for the subtask A, or come up with their own segmentation algorithm which is highly encouraged.

We decided to use the XML file and thus needed to link the annotation labels to the original XML text representation.

4 Issues

We mentioned in previous section that the segmentation into text blocks is provided in the CSV file. However, that means that we need to find the mapping from the annotated text segments onto the original XML text representation.

We wrote an algorithm that goes through both files and tries to find the best mapping on a given page assuming the annotated text from the CSV file appears in the same order of occurrence as the text in the XML file. Unfortunately, that is not always true, thus we decided to modify the training CSV file and fix the issues, described in the following sections, that caused our algorithm to fail. We fixed only the necessary part of the dataset in order for our algorithm to work. The scale of these issues is illustrated in Table 2.

The percentage ratio in Table 2 is between the original and the fixed dataset. The star sign indicates that the labels were not known at the time and thus the issue described in Section 4.1 only eliminated duplicates not taking into consideration the assigned label, leading to the removal of more title labels compared to the train dataset.

The algorithm mentioned at the beginning of this section maps up to N text blocks from the XML file to one annotation. This is basically the reverse process to the one constructing the text blocks for the CSV file. We use the first matching text segment from the XML file to assign the font and other meta-information to the annotations.

The following example is the XML file text blocks that can be mapped to the example in Section 4.2.

```
<text ...><b>4. Stock exchange listing
 </b></text>
<text ...>The Sub-Fund ... </text>
<text ...>Details regarding ...
 ... Multi-Strategy. </text>
<text ...><b>5.   Shares </b></text>
```