



UNIVERSITY OF AMSTERDAM

MSC ARTIFICIAL INTELLIGENCE  
MASTER THESIS

---

# Enriching Textual Data with Document Structure For Sentence Classification

---

by  
MAARTEN DE JONGE  
10002109

May 13, 2018

42 EC  
01 July 2017, 31 December 2017

*Supervisor:*

Dr MAARTEN MARX

*Assessor:*

Dr TO BE DETERMINED



INFORMATION AND LANGUAGE PROCESSING SYSTEMS GROUP  
INFORMATICS INSTITUTE

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Problem Statement</b>	<b>2</b>
2.1	Dataset . . . . .	3
2.2	Research Question . . . . .	7
<b>3</b>	<b>Related Work</b>	<b>8</b>
<b>4</b>	<b>Methodology</b>	<b>9</b>
4.1	Unsupervised . . . . .	9
4.1.1	Hierarchical Agglomerative Clustering . . . . .	9
4.1.2	K-Means . . . . .	12
4.2	Supervised . . . . .	12
4.2.1	Convolutional Neural Networks . . . . .	13
4.2.2	Difference between convolutional and recurrent neural networks	16
<b>5</b>	<b>Experimental Setup</b>	<b>16</b>
5.1	Dataset . . . . .	17
5.2	Testing performance . . . . .	17
5.3	CNN performance . . . . .	17
5.4	Generalisation . . . . .	17
<b>6</b>	<b>Evaluation</b>	<b>18</b>
6.1	400 samples, local max pooling . . . . .	18
6.2	400 samples, 1-max pooling . . . . .	18
<b>7</b>	<b>Conclusion</b>	<b>23</b>

# 1 Introduction

The Political Mashup project<sup>1</sup> aims to digitize the world’s political proceedings in order to make them easily accessible and searchable. Unfortunately, the published documents are often primarily intended to be human-readable, without the embedded semantic structure required to properly index this data in a digital way. This semantic information is currently recovered using rule-based methods. Since the data gets transcribed by a human typist, compiled to a PDF, and then goes back into an imperfect PDF decompiler, there is a lot of room for minor variations in the output even though the layout of the document itself is consistent. Dealing with this in a rule-based system entails using either broad rules that lead to a larger probability of false positives, or a large amount of narrow rules which can quickly lead to a spaghetti-like mess of special cases and is very fragile to unseen issues.

I propose that by using a small number of manually annotated documents as a dataset, a machine learning algorithm can learn to classify sentences in a way that allows it to segment a document into its constituent parts, while being more robust to noise than its rule-based counterpart. The common ways to do sentence classification (e.g. convolutional neural networks [1], recurrent neural networks or the simpler bag-of-words models) operate on sentences in a vacuum, considering only their linguistic contents and ignoring any contextual information that might be present. This is to be expected considering that most of the common datasets in this area really *are* just small bits of text in a vacuum; often-used datasets involve Twitter messages or short product reviews. In this case however, the sentences come from a document with a rich structure providing a lot of context. Anecdotally, as a human it is trivial to discern section headers in a document even when the document is in a foreign language; simply the fact that the section header might be printed in bold and centered rather than left-aligned gives it away. Incorporating this structural data into the learning process will hopefully increase the performance of the system, either by simply scoring better on the used metrics, or perhaps more indirectly by requiring less data or training time to achieve the same score.

## 2 Problem Statement

The German parliament, called the *Bundestag*, publishes the proceedings of their meetings, as an effort to open up the political process to the common people. These proceedings have been continuously published starting in 1949. Figure 1 shows a sample page from one of these proceedings; the left column contains a continuation of a speech from the previous page as well as two moderately sized speeches, while the right column contains a large number of very short speeches.

---

<sup>1</sup><http://search.politicalmashup.nl/about.html>

37 Having a large corpus of political proceedings like this is wonderful and opens a lot  
38 of doors for research regarding political discourse. Figure 2 shows the same page  
39 seen in Figure 1, but with a number of regions of interest (manually) colored in.  
40 Unfortunately this data is only available as PDF files, which in terms of internal  
41 representation are entirely unstructured. That means that none of the rich structure  
42 highlighted in Figure 2 is actually present in a computer-usable way.

43 The central problem in this thesis is the extraction of speeches from the docu-  
44 ments, transforming each document into a structured series of speeches that can  
45 be serve as a useful entry point into further research. As a first step, the structural  
46 layout (as in Figure 2) will be extracted using unsupervised clustering algorithms.  
47 The textual contents of the document are then fed into supervised text classifier,  
48 where each piece of text is augmented with the corresponding layout information  
49 previously obtained. More details on this process will be supplied in Section 4.

50 As annotating data by hand is an expensive process, a big focus is on limiting  
51 the amount of required training data as much as possible; it would be preferable  
52 if the system was able to learn sufficiently from a handful (say, less than 5) of  
53 hand-annotated files.

## 54 2.1 Dataset

55 PDF files are unfortunately rather difficult to work with; being a vector-based  
56 format, they have no internal concept of words or sentences. All that’s available  
57 are instructions for drawing a certain character at a certain position. This means  
58 that even something as seemingly trivial as obtaining the lines of text from a PDF  
59 requires some fairly involved logic and heuristics (for instance, one would think  
60 that simply taking all characters on a page with the same *y* coordinate would be  
61 sufficient until realizing that many documents have a layout with two columns of  
62 text). This is dealt with by using the `pdftohtml` script from the Poppler PDF  
63 rendering library<sup>2</sup>. This script converts a PDF file to an XML file containing logical  
64 lines of text along with the coordinates and size of the line. Figure 3 shows an  
65 example of a portion of a PDF file and the corresponding XML. The contents of  
66 the `text` elements forms the plaintext that is fed into the classification algorithm.  
67 Although the layout information contained in the properties (`top`, `left`, `width`,  
68 `height` and `font`) could be used for the layout clustering, this would effectively tie  
69 the performance of the clustering to the performance of `pdftohtml`’s line-extraction  
70 heuristics. Instead, a separate system is used (the Apache `pdfbox`<sup>3</sup> library for Java).  
71 By using the bounding boxes of individual characters as the base for clustering,  
72 the troubles of line-extraction can be fully bypassed.

---

<sup>2</sup><https://poppler.freedesktop.org/>

<sup>3</sup><https://pdfbox.apache.org/>

**Präsident Dr. Norbert Lammert**

- (A) Ich darf bereits jetzt darauf aufmerksam machen, dass ich nach Schließen des Wahlgangs die Sitzung für die Auszählung der Stimmen unterbrechen werde. Stellen Sie sich bitte darauf ein, dass das etwa eine Stunde dauern kann, weil ja ein doch relativ komplexer Wahlgang ausgezählt werden muss.

Ich eröffne die Wahl.

Liebe Kolleginnen und Kollegen, darf ich fragen, ob jemand im Saal ist, der seine Stimme noch nicht abgegeben hat? Oder hat jemand einen gesehen, den er dann nicht mehr gesehen hat und der seine Stimme noch abgeben könnte? – Dann schließe ich diesen Wahlgang und unterbreche die Sitzung bis zur Bekanntgabe des Ergebnisses der Wahl. Wir werden den Wiederbeginn der Sitzung rechtzeitig durch entsprechende akustische und optische Signale in den Immobilien des Bundestages ankündigen. Stellen Sie sich bitte darauf ein, dass es etwa eine Stunde dauern kann, bis wir diesen ja doch umfangreichen Wahlgang mit der gebotenen Sorgfalt ausgezählt haben.

Die Sitzung ist unterbrochen.

(Unterbrechung von 13.42 bis 14.52 Uhr)

**Präsident Dr. Norbert Lammert:**

Die unterbrochene Sitzung ist wieder eröffnet.

- (B) Liebe Kolleginnen und Kollegen, ich kann Ihnen das Ergebnis der Wahl der Stellvertreterinnen und Stellvertreter des Präsidenten bekannt geben: abgegebene Stimmkarten 626. Alle abgegebenen Stimmen waren gültig.

Von den abgegebenen Stimmen sind entfallen auf Peter Hintze 449 Jastimmen, 122 Neinstimmen und 51 Enthaltungen. In diesem Falle, was mich ein bisschen überrascht, waren 4 Stimmen ungültig. Das heißt, es gibt keine Stimmkarte, die insgesamt ungültig war, was ja doch auf eine gewisse Pfiffigkeit der neuen wie der alten Kollegen schließen lässt, aber bei einzelnen Wahlgängen ist das offenkundig anders. Noch einmal: 449 Jastimmen, 122 Neinstimmen, 51 Enthaltungen. Ich darf das mit Ihrem Einverständnis gleich mit der Frage an die jeweiligen Kolleginnen und Kollegen verbinden, ob sie die Wahl annehmen. Ich darf den Kollegen Hintze, der damit die notwendige Mehrheit erkennbar erreicht hat, fragen, ob er die Wahl annimmt.

**Peter Hintze (CDU/CSU):**

Ich bedanke mich. Ich nehme die Wahl an.

(Beifall bei der CDU/CSU sowie bei Abgeordneten der SPD und des BÜNDNISSES 90/DIE GRÜNEN)

Auf den Kollegen Johannes Singhammer sind bei 6 ungültigen Stimmen 442 Jastimmen, 115 Neinstimmen und 63 Enthaltungen entfallen. Auch er hat damit die notwendige Mehrheit eindeutig und klar erreicht. Ich darf ihn fragen, ob er die Wahl annimmt.

**Johannes Singhammer (CDU/CSU):**

(C) Ich danke für den Vertrauensvorschuss und nehme die Wahl gerne an.

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Die Kollegin Edelgard Bulmahn hat bei wiederum 6 ungültigen Stimmen 534 Jastimmen erhalten.

(Beifall im ganzen Hause)

50 Kolleginnen und Kollegen haben mit Nein gestimmt, 36 haben sich der Stimme enthalten. Frau Bulmahn, ich darf auch Sie fragen, ob Sie die Wahl annehmen.

**Edelgard Bulmahn (SPD):**

Auch ich bedanke mich für das Vertrauen, und ich nehme die Wahl gerne an.

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Auf die vorgeschlagene Kandidatin Ulla Schmidt sind 520 Jastimmen entfallen.

(Beifall im ganzen Hause)

66 Kollegen oder Kolleginnen haben mit Nein gestimmt, 35 haben sich der Stimme enthalten. 5 Stimmen waren ungültig. Ich bin zuversichtlich, Frau Schmidt, dass Sie die Frage ähnlich beantworten wie die bisher angesprochenen Kolleginnen und Kollegen.

**Ulla Schmidt (Aachen) (SPD):**

(D) Herr Präsident, Sie haben wie meistens recht. Ich nehme die Wahl an und bedanke mich für das große Vertrauen. Danke schön!

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Auf Petra Pau sind 451 Jastimmen entfallen,

(Beifall im ganzen Hause)

bei 113 Neinstimmen und 45 Enthaltungen. 17 Stimmen waren in diesem Wahlvorgang ungültig. Ich darf Frau Pau fragen, ob sie die Wahl annimmt.

**Petra Pau (DIE LINKE):**

Ja, Herr Präsident, ich nehme die Wahl gern an, und, liebe Kolleginnen und Kollegen, ich freue mich auf die weitere Zusammenarbeit.

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Schließlich darf ich noch das Wahlergebnis für Claudia Roth bekannt geben. Bei 14 ungültigen Stimmen hat sie 415 Jastimmen erhalten. Es gab 128 Neinstimmen und 69 Enthaltungen. Sie ist damit gewählt.

(Beifall im ganzen Hause – Claudia Roth [Augsburg] [BÜNDNIS 90/DIE GRÜNEN]:

**Figure 1** – A sample page from one of the Bundestag proceedings.

(A) **Präsident Dr. Norbert Lammert**

Ich darf bereits jetzt darauf aufmerksam machen, dass ich nach Schließen des Wahlgangs die Sitzung für die Auszählung der Stimmen unterbrechen werde. Stellen Sie sich bitte darauf ein, dass das etwa eine Stunde dauern kann, weil ja ein doch relativ komplexer Wahlgang ausgezählt werden muss.

Ich eröffne die Wahl.

Liebe Kolleginnen und Kollegen, darf ich fragen, ob jemand im Saal ist, der seine Stimme noch nicht abgegeben hat? Oder hat jemand einen gesehen, den er dann nicht mehr gesehen hat und der seine Stimme noch abgeben könnte? – Dann schließe ich diesen Wahlgang und unterbreche die Sitzung bis zur Bekanntgabe des Ergebnisses der Wahl. Wir werden den Wiederbeginn der Sitzung rechtzeitig durch entsprechende akustische und optische Signale in den Immobilien des Bundestages ankündigen. Stellen Sie sich bitte darauf ein, dass es etwa eine Stunde dauern kann, bis wir diesen ja doch umfangreichen Wahlgang mit der gebotenen Sorgfalt ausgezählt haben.

Die Sitzung ist unterbrochen.

(Unterbrechung von 13.42 bis 14.52 Uhr)

**Präsident Dr. Norbert Lammert:**

Die unterbrochene Sitzung ist wieder eröffnet.

Liebe Kolleginnen und Kollegen, ich kann Ihnen das Ergebnis der Wahl der Stellvertreterinnen und Stellvertreter des Präsidenten bekannt geben: abgegebene Stimmkarten 626. Alle abgegebenen Stimmen waren gültig.

Von den abgegebenen Stimmen sind entfallen auf Peter Hintze 449 Jastimmen, 122 Neinstimmen und 51 Enthaltungen. In diesem Falle, was mich ein bisschen überrascht, waren 4 Stimmen ungültig. Das heißt, es gibt keine Stimmkarte, die insgesamt ungültig war, was ja doch auf eine gewisse Pfiffigkeit der neuen wie der alten Kollegen schließen lässt, aber bei einzelnen Wahlgängen ist das offenkundig anders. Noch einmal: 449 Jastimmen, 122 Neinstimmen, 51 Enthaltungen. Ich darf das mit Ihrem Einverständnis gleich mit der Frage an die jeweiligen Kolleginnen und Kollegen verbinden, ob sie die Wahl annehmen. Ich darf den Kollegen Hintze, der damit die notwendige Mehrheit erkennbar erreicht hat, fragen, ob er die Wahl annimmt.

**Peter Hintze (CDU/CSU):**

Ich bedanke mich. Ich nehme die Wahl an.

(Beifall bei der CDU/CSU sowie bei Abgeordneten der SPD und des BÜNDNISSES 90/DIE GRÜNEN)

Auf den Kollegen Johannes Singhammer sind bei 6 ungültigen Stimmen 442 Jastimmen, 115 Neinstimmen und 63 Enthaltungen entfallen. Auch er hat damit die notwendige Mehrheit eindeutig und klar erreicht. Ich darf ihn fragen, ob er die Wahl annimmt.

(C) **Johannes Singhammer (CDU/CSU):**

Ich danke für den Vertrauensvorschuss und nehme die Wahl gerne an.

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Die Kollegin Edelgard Bulmahn hat bei wiederum 6 ungültigen Stimmen 534 Jastimmen erhalten.

(Beifall im ganzen Hause)

50 Kolleginnen und Kollegen haben mit Nein gestimmt, 36 haben sich der Stimme enthalten. Frau Bulmahn, ich darf auch Sie fragen, ob Sie die Wahl annehmen.

**Edelgard Bulmahn (SPD):**

Auch ich bedanke mich für das Vertrauen, und ich nehme die Wahl gerne an.

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Auf die vorgeschlagene Kandidatin Ulla Schmidt sind 520 Jastimmen entfallen.

(Beifall im ganzen Hause)

66 Kollegen oder Kolleginnen haben mit Nein gestimmt, 35 haben sich der Stimme enthalten. 5 Stimmen waren ungültig. Ich bin zuversichtlich, Frau Schmidt, dass Sie die Frage ähnlich beantworten wie die bisher angesprochenen Kolleginnen und Kollegen.

(D) **Ulla Schmidt (Aachen) (SPD):**

Herr Präsident, Sie haben wie meistens recht. Ich nehme die Wahl an und bedanke mich für das große Vertrauen. Danke schön!

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Auf Petra Pau sind 451 Jastimmen entfallen,

(Beifall im ganzen Hause)

bei 113 Neinstimmen und 45 Enthaltungen. 17 Stimmen waren in diesem Wahlvorgang ungültig. Ich darf Frau Pau fragen, ob sie die Wahl annimmt.

**Petra Pau (DIE LINKE):**

Ja, Herr Präsident, ich nehme die Wahl gern an, und, liebe Kolleginnen und Kollegen, ich freue mich auf die weitere Zusammenarbeit.

(Beifall im ganzen Hause)

**Präsident Dr. Norbert Lammert:**

Schließlich darf ich noch das Wahlergebnis für Claudia Roth bekannt geben. Bei 14 ungültigen Stimmen hat sie 415 Jastimmen erhalten. Es gab 128 Neinstimmen und 69 Enthaltungen. Sie ist damit gewählt.

(Beifall im ganzen Hause – Claudia Roth [Augsburg] [BÜNDNIS 90/DIE GRÜNEN])

**Figure 2** – The same page as in Figure 1, hand-annotated with interesting regions that play a significant role in understanding the layout. The red regions are the headers that signify that someone is starting a speech, and contain information about who the speakers is. The blue regions are little interruptions (*Heiterkeits*), often signifying approval or displeasure (the regularly seen *Beifall* means applause). The green regions indicate plain blocks of text.

**Dr. Norbert Lammert (CDU/CSU):**  
Herr Alterspräsident, lieber Kollege Riesenhuber, ich  
nehme die Wahl gerne an.

(Beifall im ganzen Hause – Abgeordnete aller  
Fraktionen gratulieren dem Präsidenten)

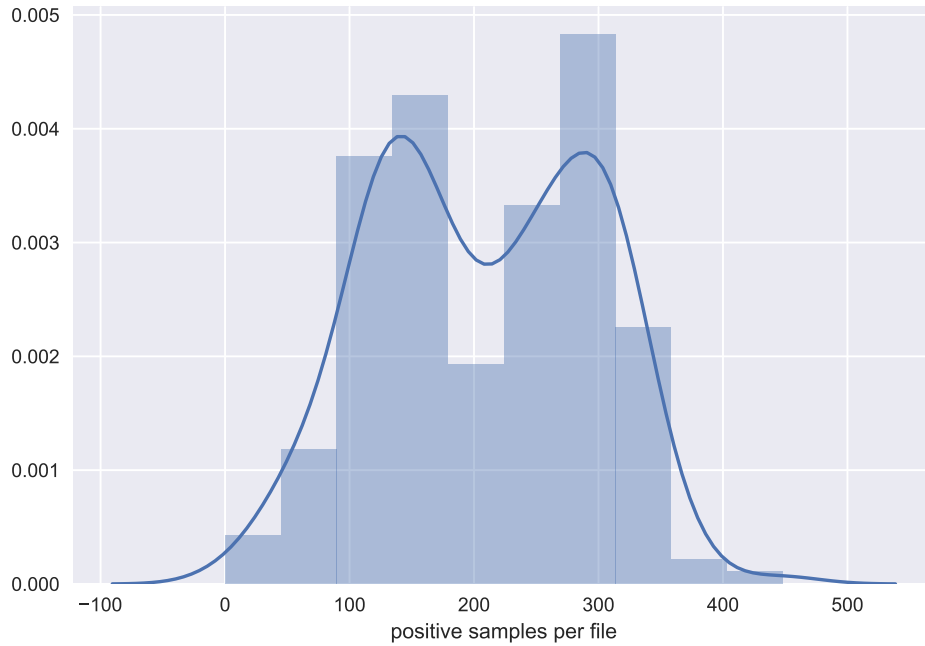
(a) A portion of the source PDF.

```
<text top="122" left="125" width="143" height="16" font="3">  
  <b>Dr. Norbert Lammert </b>  
</text>  
<text top="122" left="269" width="83" height="17" font="4">  
  (CDU/CSU) :  
</text>  
<text top="142" left="125" width="328" height="17" font="4">  
  Herr Alterspräsident, lieber Kollege Riesenhuber, ich  
</text>  
<text top="158" left="108" width="156" height="17" font="4">  
  nehme die Wahl gerne an.  
</text>  
<text top="186" left="141" width="278" height="17" font="4">  
  (Beifall im ganzen Hause      Abgeordnete aller  
</text>  
<text top="203" left="158" width="242" height="17" font="4">  
  Fraktionen gratulieren dem Präsidenten)  
</text>
```

(b) XML created by running `pdftohtml`, corresponding to the PDF excerpt in Figure 3a. The contents of the `text` elements are used as inputs for the classification algorithm; the layout data contained in the properties is not used, as a separate software pipeline is used for the unsupervised clustering.

**Figure 3** – A sample excerpt from a source PDF, along with its XML representation created by `pdftohtml`.

73 The dataset was obtained through a rule-based system as described in the  
 74 introduction, which annotates each `<text>` element of the XML files with a boolean  
 75 flag indicating whether said element starts a new speech. The system was run  
 76 on documents from the 18th electoral period of the *Bundestag*, consisting of 211  
 77 documents dating from 2013 to 2017. Together these documents contain 43,252  
 78 `<text>` elements indicating the start of speeches (that is, positive training samples),  
 79 and 2,602,793 other elements (negative samples). This adds to a total of 2,646,045  
 80 training samples, taking up 503 MiB. This is a rather lopsided distribution (there  
 81 are roughly 60 negative samples for each positive sample), which will have to  
 82 be accounted for by, for instance, using stratified sampling. Figure 4 shows the  
 83 distribution of the number of positive samples per file, giving a guideline as to  
 84 how many files would have to be annotated to reach a desired amount of positive  
 85 samples.



**Figure 4** – Distribution of the number of positive samples per file

## 86 **2.2 Research Question**

87 As the application of supervised text classification is fairly basic and well-studied,  
 88 academically speaking the interesting portion of this problem is figuring out what



benefit is added by the unsupervised layout clustering, if any. A potential benefit could manifest itself in a couple of different ways:

- The performance (using a metric such as the F1 score or average precision, to be further elaborated on in Section 4) after training is increased.
- The peak performance is equal, but less training data is required to reach said peak.
- The peak performance is equal and a similar amount of training data is required, but the classifier needs less iterations to reach this performance.

This leads to the research question that is central to this thesis:

**Research Question 1** *Does augmenting a text classification system with layout information obtained by unsupervised clustering of the input data improve the classifier with regards to:*

- *F1 score and average precision*
- *required volume of training data*
- *training speed*

### 3 Related Work

Todo: Expand section

Various forms of convolutional neural networks are commonly used for text classification. The most basic architecture is described by Kim [1], where the input words are tokenized and embedded before passing them to the convolutional neural network. Additional exploration of the parameter space and its effect on various datasets is done by Zhang & Wallace [2]. Comparable results are achieved by Zhang *et al.* [3] by operating on the character-level rather than the word-level, bypassing the issues overhead of using word embedding (either in extra training time or in finding suitable pretrained embeddings). All the previously mentioned architectures use a single convolutional layer; this is somewhat contrary to current trends in computer vision, where popular models such as ResNet[4] go as deep as 152 layers. This difference is explored by Conneau *et al.* [5], who take a character-level CNN and show that adding more layers improves performance, before leveling out at 29 layers. They hypothesize that the difference in effective depth between computer vision and language processing might be due to the difference in datasets. The common ImageNet dataset used in computer vision deals with 1000 classes; in

contrast, sentiment analysis datasets vary between 2 and 25 classes. In addition, they note that the deeper networks do require a larger amount of data to train.

In terms of analyzing document structure, Klampfl *et al.* [6] introduce a method to analyze scientific articles, detecting blocks of text, labeling them (as e.g. section headers, tables or references) and determining the reading order — all in an unsupervised manner. While their approach to block detection forms an integral part of this thesis, the rest is too specifically tied to the format of scientific articles to be applicable in this scenario.

## 4 Methodology

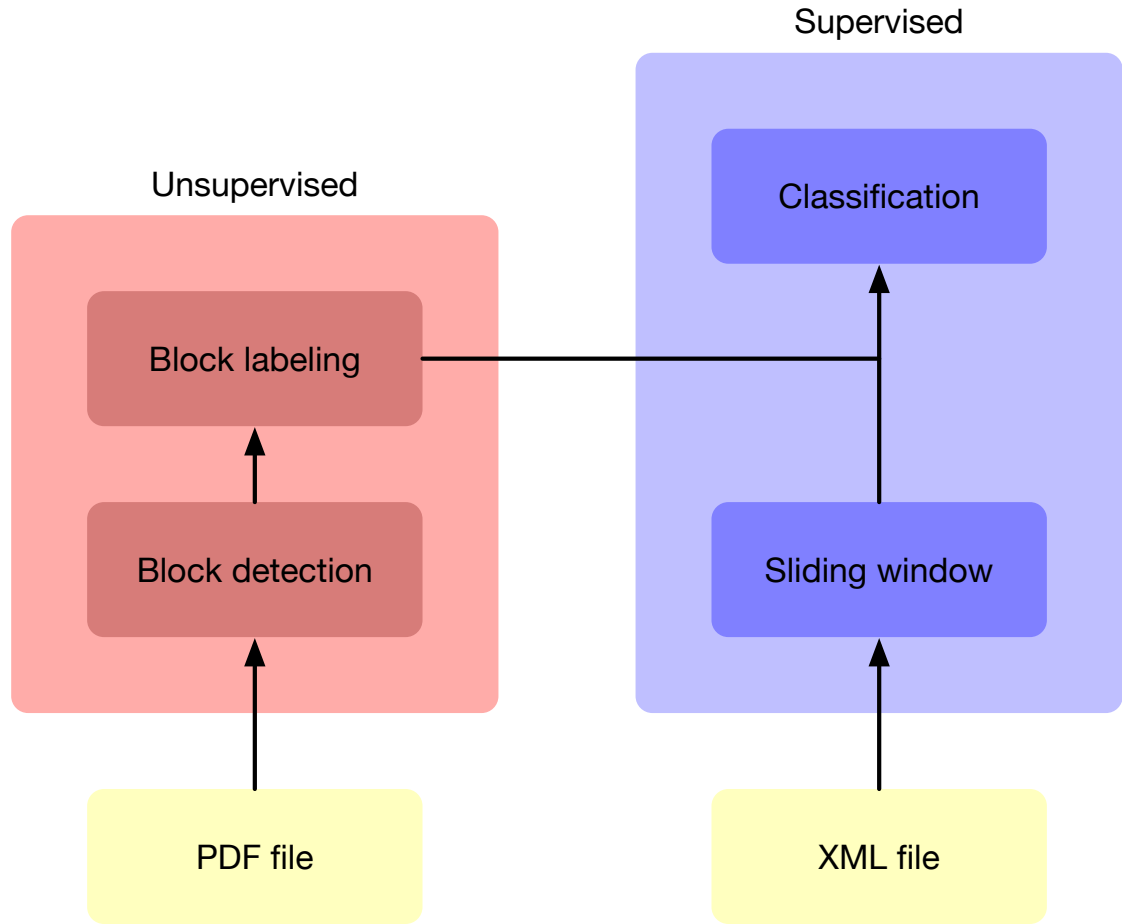
As described in Section 2, the input data comes in the form of a PDF file which can be converted into plaintext XML data containing lines of text along with each line’s bounding box. There are two systems in play here; the main system is a convolutional neural network classifier that acts on the XML data. Further details on this are provided in Section 4.2. The second system is the unsupervised preprocessor which acts on the PDF data and writes its output as an additional property into the XML data, which is elaborated upon in Section 4.1. Figure 5 shows a high-level overview of the two systems and how they interact.

### 4.1 Unsupervised

The unsupervised algorithm intends to detect and classify blocks of text in the PDF file; Figure 6 shows an example. This approach is based on work by Klampfl *et al.* [6], and consists of two separate clustering steps. First, individual characters (the fundamental objects available in a PDF file) are clustered together into blocks of semantically relevant text. These would be paragraphs, section headers, page decoration, etc. By using the bounding boxes of the blocks, they can be clustered based on their shape and some additional metadata (e.g. occurrence of font types and sizes). The next two subsections will go into details on the two clustering steps.

#### 4.1.1 Hierarchical Agglomerative Clustering

The first step is performed using hierarchical agglomerative clustering (HAC), an unsupervised bottom-up clustering algorithm that constructs a hierarchical tree of clusters (in this context referred to as a *dendrogram*). An example is shown in Figure 7. The algorithm gets fed the individual characters present in the PDF files, then iteratively groups the two closest clusters (the initial inputs being regarded as clusters of one element) together until only a single cluster remains. This process involves two parameters:



**Figure 5** – A high-level overview of the system. The unsupervised block augments the input to the classifier.

- 155 1. The distance function between two characters.
  - 156 2. The distance function between two clusters of characters.
- 157 The first parameter is trivially chosen to be the Euclidian distance between the  
158 coordinates of the two characters. The second parameter is called the *linkage* and  
159 has several common options, the most basic of which are:
- 160 • Single-linkage: The distance between clusters is based on the closest two  
161 elements:

$$d(A, B) = \min\{d(a, b) : a \in A, b \in B\}$$

- 162 • Maximum-linkage: The distance between clusters is based on the furthest  
163 two elements:

$$d(A, B) = \max\{d(a, b) : a \in A, b \in B\}$$



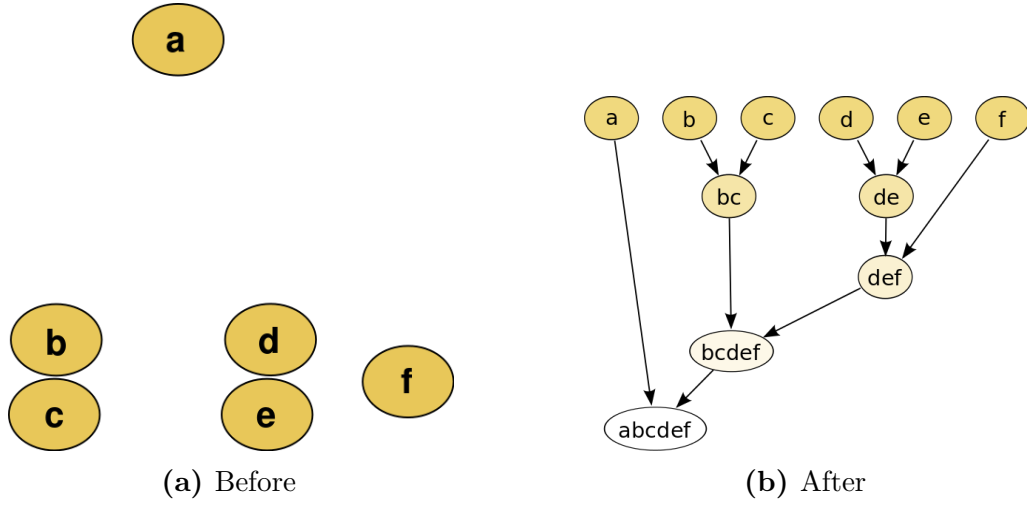
**Figure 6** – An example of clustered blocks of text. Blocks with the same outline color belong to the same cluster.

- Average-linkage: The distance between clusters is based on the average distance of its elements:

$$d(A, B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$$

As per Klampfl *et al.* [6], single-linkage clustering performs best for this task due to its tendency to form long thin clusters. This is beneficial since text is somewhat long and thin in nature (especially words and sentences). As an additional bonus, while the general time complexity for HAC is in  $\mathcal{O}(n^3)$ , single-linkage clustering can be done in  $\mathcal{O}(n^2)$  [7], making it far more usable on realistic datasets.

After the dendrogram is constructed, it has to be cut at some level to obtain the desired blocks of text. Clustering can optionally be rerun using the newly found clusters as basic elements. This way, the document can incrementally be clustered from characters into words, words into lines, and finally lines into paragraphs. Both the level at which to cut the tree and the number of times to recluster are determined by trial and error based on the particular set of documents.



**Figure 7** – An example of hierarical agglomerative clustering, where the nodes are clustered by distance.

#### 177 4.1.2 K-Means

178 The extracted blocks from the previous step are then clustered according to their  
179 similarity based on the following metrics:

- 180 • Width of the cluster
- 181 • Height of the cluster
- 182 • The ID of the most common font occurring in the cluster
- 183 • The size of the most common font occurring in the cluster

184 This is done using K-means clustering, with the value of  $k$  being varied for experi-  
185 mental purposes.

## 186 4.2 Supervised

187 After the data is augmented by the previously described clustering algorithms, it's  
188 fed into a convolutional neural network for classification. Since the source PDFs  
189 have a dual column layout with rather short lines, a sliding window is used to add  
190 valuable context, with the center element in the window supplying both the label  
191 (i.e. does or does not start a speech) and the cluster type. The text content of this  
192 window is then entered into a standard convolutional neural network architecture  
193 similar to the one proposed by Kim [1]. First an embedding layer is used to learn  
194 a high-dimensional representation of the words (no pre-trained embeddings were

195 used because of both the specialized political domain of the data as well as a lack  
196 of pre-trained German models), followed by a number of Convolutional filters with  
197 max pooling. The output of the filters is then concatenated into a single feature  
198 vector, which is amended with the cluster type as an additional feature before  
199 feeding it into a standard fully-connected neural network. This network features  
200 one hidden layer with a ReLU activation ( $\text{relu}(x) = \max(0, x)$ ) and a single output  
201 node with a sigmoid activation. The layout of this system is detailed in Figure 8  
202 along with its parameters and their baseline value.

#### 203 4.2.1 Convolutional Neural Networks

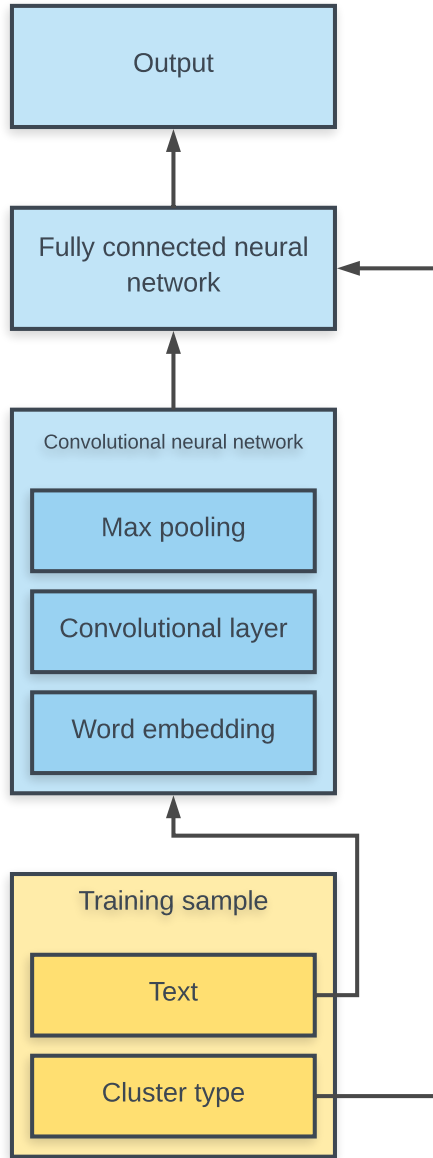
This is kind of jumbled and out of place, might need to be deleted or moved to an appendix or something.

204  
205 For the purposes of machine learning, text produces 3-dimensional data: there  
206 is a feature vector for each word, and some (either variable or predetermined  
207 through padding) amount of words per text. This makes each training sample a  
208 2-dimensional matrix, which then gets stacked in the depth dimension to produce  
209 3-dimensional training data. This is troublesome as the standard machine learning  
210 algorithms work on 2-dimensional data, assuming a feature vector for each sample  
211 rather than a matrix. There are three common methods to deal with this:

- 212 • Bag of words
- 213 • Convolutional neural networks
- 214 • Recurrent neural networks

215 Aside from being completely different methods, they differ in a major way in  
216 how they handle the sequential nature of text. The bag of words approach is the  
217 simplest in that it simply disregards this sequential nature, instead creating what  
218 is essentially a histogram of word occurrences. This downsamples each sample  
219 from a feature matrix to a feature vector, allowing the use of normal machine  
220 learning algorithms (commonly support vector machines). While the sequential  
221 information can be kept to some degree by use histograms of  $n$ -grams rather than  
222 words (unigrams), this causes the size of the input data to scale exponentially with  
223 the value of  $n$ .

224 Convolutional neural networks (CNNs) work by taking a number of filters  
225 (sometimes called kernels or feature maps) of a specified size and convolving these  
226 over the input data. A simplified example using one filter is shown in Figure 9.  
227 In this example, the input text is convolved with a filter with a width of 3 and a  
228 stride of 1 — that is, each application of the filter considers three subsequent input  
229 elements, after which the window is shifted one space to the right. This filter is

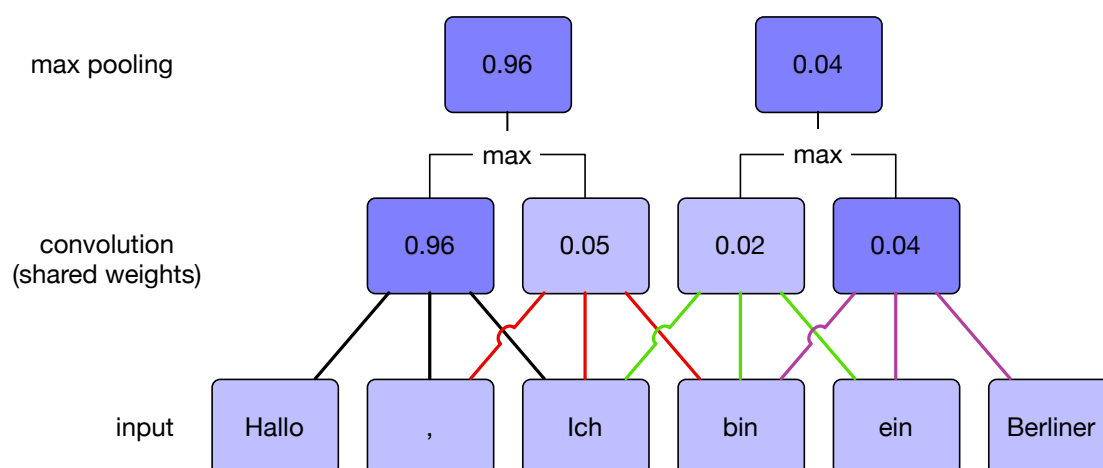


(a) The layout of the supervised portion of the system. The input data contains text and a cluster type assigned by the unsupervised portion. The text gets put into a convolutional neural network, the output of which is fed together with the cluster type into a fully connected neural network. The sigmoid function is applied to the output of this final neural network to obtain the classification.

Parameter	Default value
Window size	3
Word embedding size	300
Number of convolution filters	100
Convolution filter sizes	3, 5, 7
Convolution activation	ReLU
Pooling type	1-max
Number of hidden layers	1
Hidden layer size	50
Number of outputs	1
Output activation	Sigmoid

(b) The default parameters used in the model.

**Figure 8** – The model and its parameters.



**Figure 9** – A simplified convolutional neural network with one filter and max pooling.

essentially a small neural network mapping three items to one output value, whose weights are reused for each application of the filter. Reusing the weights in this way (weight sharing) prevents the number of parameters in the network from spiraling out of control. [8] After the application of this convolution layer, the responses of the filter form a new sequence of roughly the same size as the input (minus a few due to missing the edges). The next step is to downsample this sequence by means of a *max pooling* layer, which maps all values within a window to the maximum value amongst those values. While conceptually similar to a convolution, this step generally does not involve overlap, instead either setting the stride to the same value as the window size (usually 2) reducing the entire sequence to 1 value (1-max pooling). The reason for this is twofold:

1. It downsamples the number of inputs, reducing the amount of parameters required further on in the network.
2. It adds translation invariance to the feature detected by this filter. The example filter of Figure 9 appears to react strongly to the presence of the word “Hallo”. Without the pooling layer, changing the location of the word “Hallo” in the input would similarly change the location of the high activation in the intermediate representation; this would be *equivariance*. The more aggressively the pooling is applied, the higher the degree of invariance.

This combination of convolution followed by pooling can be repeated multiple times as desired or until there is only a single value left as output from the filter. Finally, the outputs of all filters are concatenated and fed into a standard feedforward neural network.



253 While CNN architectures in computer vision are generally very deep, they  
254 tend to be very shallow in natural language processing; commonly just a single  
255 convolution followed by 1-max pooling [2]. Since this particular task at first glance  
256 appears to be fairly reliant on word position (e.g. a colon at the end of a sentence  
257 very often indicates the start of a speech, a colon in the middle almost certainly  
258 does not), the degree of pooling will be experimented with.

#### 259 4.2.2 Difference between convolutional and recurrent neural networks

260 Recurrent neural networks (in particular LSTMs or GRUs) are seemingly the most  
261 natural fit for language processing, since they process an entire sequence and are  
262 therefore fully conditioned on the word order (as opposed to the convolutional neural  
263 networks which tend to learn translation invariant ngram features). Regardless,  
264 convolutional neural networks will be used in this research. This choice is based on  
265 two factors:

- 266 1. In practise, the performance for classification tasks does not differ between  
267 the two types of networks.[9]
- 268 2. The computations in convolutional networks are highly independent of each  
269 other, allowing for great paralellization (in particular with regards to running  
270 on a GPU). In contrast, LSTMs are bottlenecked by the fact that each  
271 calculation is dependent on the previous calculations. As a result, CNNs  
272 achieve far higher training speeds.[10]

## 273 5 Experimental Setup

274 Experiments are focused on the difference in performance between the baseline  
275 CNN model without clustering information (referred to from here on as **CNN**) and  
276 the model augmented with clustering information (which will be referred to as  
277 **CNN-cluster**). Performance will be measured with regards to the following three  
278 metrics:

- 279 1. Number of training epochs until convergence
- 280 2. The F1 score or average precision metrics on a test set (see Section 5.2)
- 281 3. The number of training samples required to attain a specific F1/average  
282 precision score

283 In each case, the experiment will be repeated 10 times by means of 10-fold cross  
284 validation followed by a Student's T-test to gauge the probability of the following  
285 null hypothesis being true:

286 **Null Hypothesis 1** *Adding clustering information to the CNN model does not*  
287 *change the performance of the model.*

## 288 5.1 Dataset

289 Referring back to Figure 4, the average document has between 100 and 300 positive  
290 samples. Since a secondary concern is to minimize the number of documents that  
291 would have to be annotated as training data, the tested dataset sizes will be very  
292 low, with the number of positive samples being one of 100, 200, 500 and 1000. Due  
293 to the relative abundance of negative samples and to prevent overfitting on the  
294 distribution of the labels, stratified sampling will be used to keep a 1:1 ratio of  
295 positive to negative samples. In addition to the size, the number of cluster types  
296 (the  $k$  in  $k$ -means) will be varied to examine its impact on the performance.

## 297 5.2 Testing performance

298 All models will be tested on a test set containing 1000 positive samples and  
299 10000 negative samples, all of which are guaranteed not to be in the training set.  
300 Performance on this set is measured by constructing a precision-recall curve, and  
301 calculating two values:

- 302 1. The average precision (which is equivalent to the area under the curve)
- 303 2. The F1 score of the point on the curve maximizing the F1 score

## 304 5.3 CNN performance

305 Although less central to the thesis than the difference between the CNN and  
306 CNN-cluster models, some experimentation will be done with the parameters  
307 of the convolutional network in an attempt to optimize the performance. These  
308 parameters include the dimensionality of the word embeddings, the number of  
309 filters, the pooling strategy (1-max versus a smaller region) and the number of  
310 convolutional layers.

## 311 5.4 Generalisation

312 This particular dataset has the quirk that the performance of a rule-based system  
313 created based on recent documents decreases in performance when used on older  
314 documents, the older the document the worse it performs. This occurs despite the  
315 layout being visually the same all the way back to the 1950s. A number of files  
316 from old election periods has been labeled (and manually verified for correctness)  
317 in order to test

1. whether the CNN models handle this better than the rule-based system does.
2. Whether the clustering-augmented CNN model performs better on this task than the baseline CNN.

## 6 Evaluation

### 6.1 400 samples, local max pooling

Turn this into actual text

T-test probability that the AoC distributions are the same: 0.009790328758564197  
T-test probability that the F1 distributions are the same: 0.011943604700946929

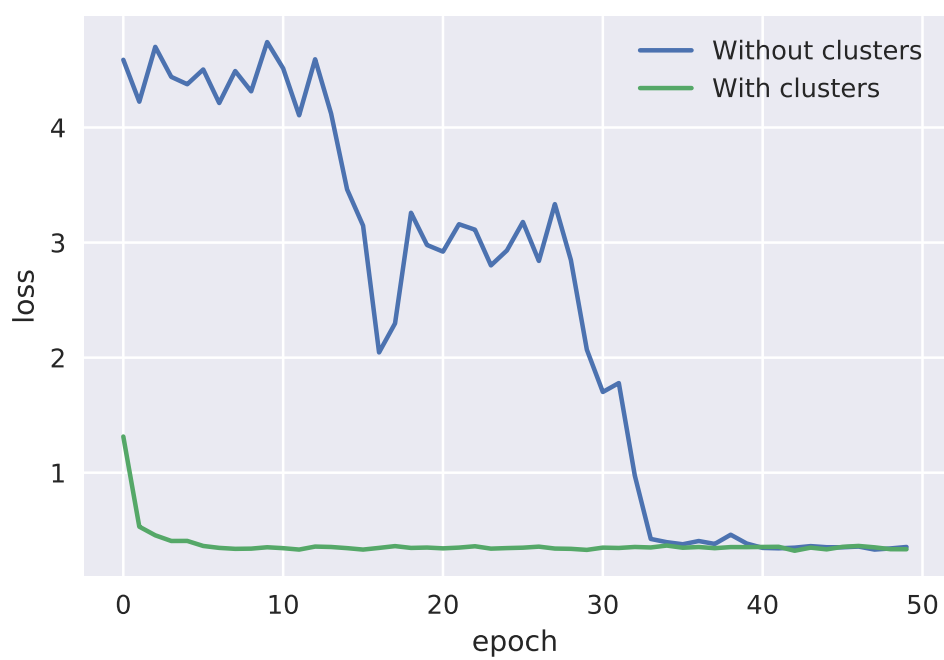
model	F1		AoC	
	mean	std	mean	std
CNN	0.79126	0.01644	0.856768	0.0219711
CNN-clusters	0.83429	0.02047	0.904359	0.0168642

**Table 1** – Performance metrics on a small dataset (200 positive samples)

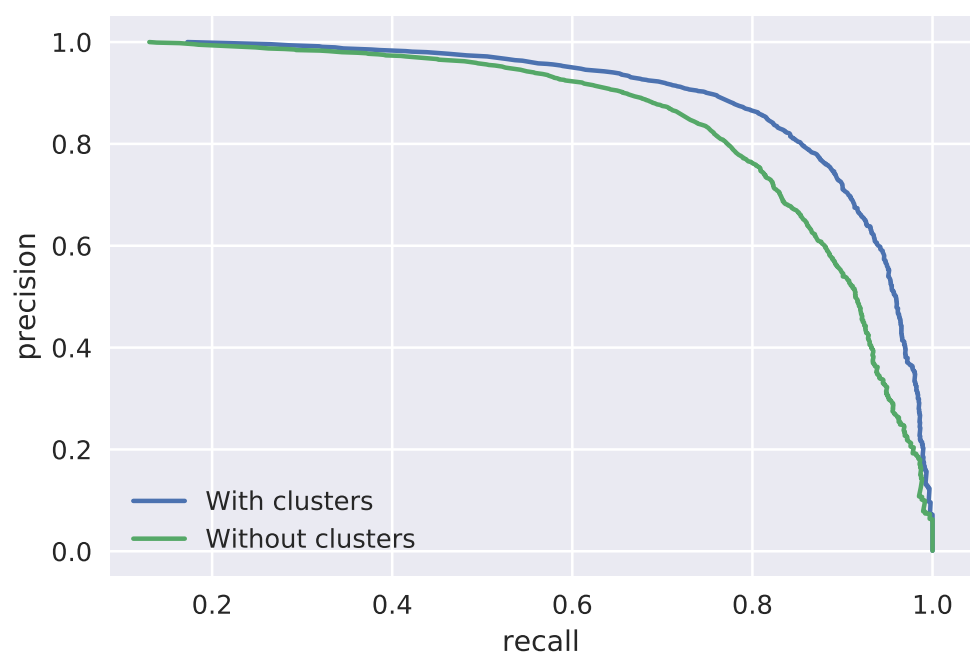
### 6.2 400 samples, 1-max pooling

model	F1		AoC	
	mean	std	mean	std
CNN	0.916071	0.00786046	0.934946	0.0183115
CNN-clusters	0.916407	0.00664848	0.931635	0.0270519

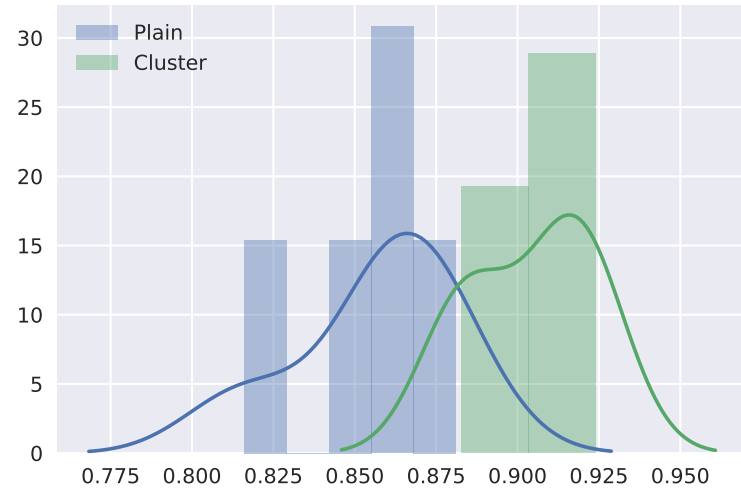
**Table 2** – Performance metrics on 200 positive and 200 negative samples with 1-max pooling. The T-test probability of the null hypothesis being true for the average precision is 0.765, for the F1 score it is 0.923. This makes it fairly save to say that for this configuration there is no benefit to the addition of clustering data.



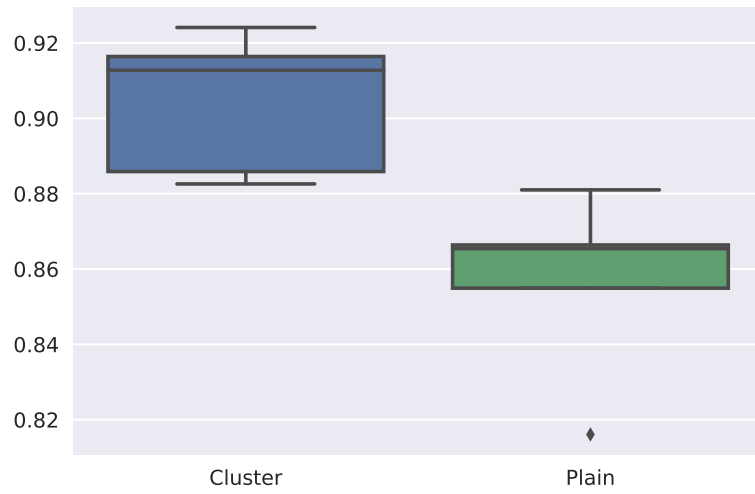
**Figure 10** – The convergence speed



**Figure 11** – The precision/recall curves

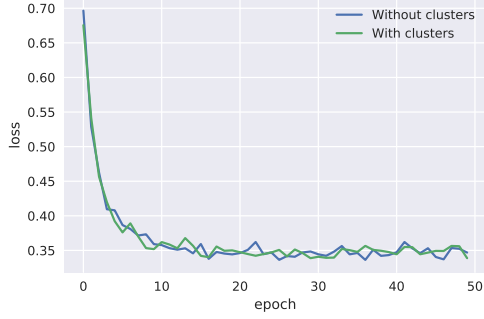


(a) Kernel density estimation

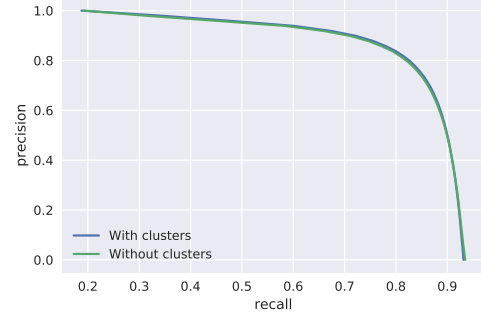


(b) Boxplot

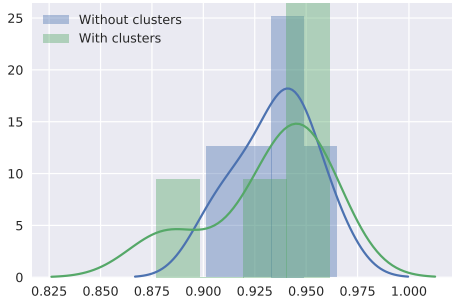
**Figure 12** – The distribution of the cross validation results



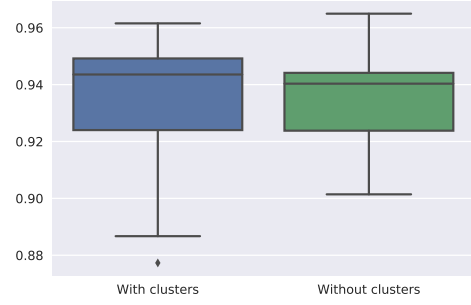
(a) Average loss at each epoch



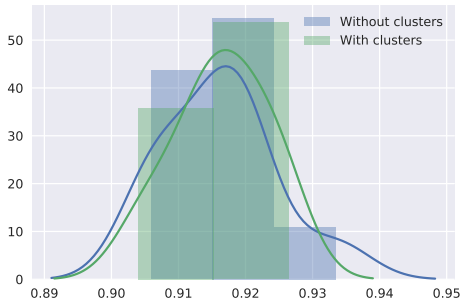
(b) Average precision/recall curve



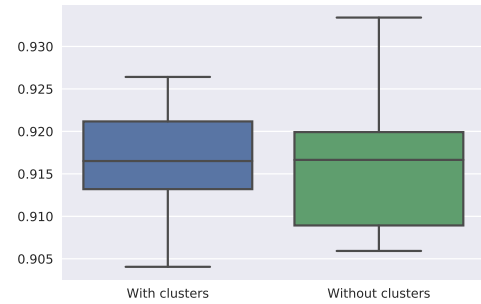
(c) Kernel density estimation of average precision



(d) Box plot of average precision



(e) Kernel density estimation of F1 score



(f) Box plot of F1 score

**Figure 13** – At 200 positive and 200 negative samples with 1-max pooling, there is no discernable difference between the two models.

## 7 Conclusion

Todo: conclusion

## References

1. Kim, Y. Convolutional Neural Networks for Sentence Classification. *CoRR* **abs/1408.5882**. arXiv: 1408.5882. <<http://arxiv.org/abs/1408.5882>> (2014).
2. Zhang, Y. & Wallace, B. C. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. *CoRR* **abs/1510.03820**. arXiv: 1510.03820. <<http://arxiv.org/abs/1510.03820>> (2015).
3. Zhang, X., Zhao, J. & LeCun, Y. *Character-level convolutional networks for text classification* in *Advances in neural information processing systems* (2015), 649–657.
4. He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. *CoRR* **abs/1512.03385**. arXiv: 1512.03385. <<http://arxiv.org/abs/1512.03385>> (2015).
5. Conneau, A., Schwenk, H., Barrault, L. & LeCun, Y. Very Deep Convolutional Networks for Natural Language Processing. *CoRR* **abs/1606.01781**. arXiv: 1606.01781. <<http://arxiv.org/abs/1606.01781>> (2016).
6. Klampfl, S., Granitzer, M., Jack, K. & Kern, R. Unsupervised document structure analysis of digital scientific articles. *International Journal on Digital Libraries* **14**, 83–99 (2014).
7. Sibson, R. SLINK: an optimally efficient algorithm for the single-link cluster method. *The computer journal* **16**, 30–34 (1973).
8. LeCun, Y., Bengio, Y., *et al.* Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* **3361**, 1995 (1995).
9. Yin, W., Kann, K., Yu, M. & Schütze, H. Comparative Study of CNN and RNN for Natural Language Processing. *CoRR* **abs/1702.01923**. arXiv: 1702.01923. <<http://arxiv.org/abs/1702.01923>> (2017).
10. Gehring, J., Auli, M., Grangier, D., Yarats, D. & Dauphin, Y. N. Convolutional Sequence to Sequence Learning. *CoRR* **abs/1705.03122**. arXiv: 1705.03122. <<http://arxiv.org/abs/1705.03122>> (2017).