# Text classification of municipality documents
# A comparison between Naive Bayes and CNN's

Axel Hirschel, 10656146

March 4, 2018

**Abstract**

The municipalities within the Netherlands produce a large amount of official documents. However, the content of these documents often remain unlabelled. This research is aimed at alleviating this problem using Machine Learning, however, there is no training data available with documents of the local municipalities. Therefore a training set of the central government is used as training data.
This research contains a comparable studies between two methodologies of categorizing the data. The first method is a Naive Bayes implementation using bag-of-words to categorize the data. Secondly, a convolutional neural network using Word2Vec is used for categorizing the data. Both algorithms are first trained and tested on the data of the central government, and then a final evaluation is done on a manually labelled test set of the municipalities.

## 1 Personal details

Mail Axel Hirschel: AxelHirschel@Gmail.com
Mail internal supervisor Maarten Marx: M.J.Marx@UvA.nl
Mail external supervisor Tom Kunzler: Tom@OpenState.eu
Github of thesis: `https://github.com/DeWvanAxel/thesis`

## 2 Research question

Do convolutional neural networks outperform Naive Bayes on classifying the content of municipality documents?

1. What is the performance of both algorithms on the training set, consisting of documents labelled by the Dutch national government?
   (a) How does the amount of topics influence their performance?
   (b) Since many documents are multi-labelled, what threshold should be used to select relevant categories?
2. How well do both algorithms generalize their classification on a set of documents of Dutch municipalities?

# 3 Related Literature

Text classification has been an important field of studies within information retrieval. The task is to select labels for documents given the experience on a test-set. A often used metric for evaluation is the F-1 score of the models. [1]
One of the most prominent ways for text classification is the Naive Bayes classifier. The Naive Bayes classifier learns the probability words occur within a certain topic. New documents can be classified using the distribution of words within that document. However, the Naive Bayes assumes that items occur independently of each other, and this assumption often does not hold. Still, the Naive Bayes can perform relatively well. [1]
Two distinct ways of Naive Bayes can be found: Multi-variate Bernoulli model and multinomial model. The main difference is the input for the model. Within the multi-variate Bernoulli a binary vector representing the presence or absence of words in the vocabulary is used whereas the multinomial model uses bag-of-words. This means that the amount of occurrences is also tracked. [6]
For both models its learning process is relatively similar. For each word it is calculated how many times it occurs in each class. Then this number plus one is divided by the amount of unique words plus the amount of words within documents of that class. This function yields the probability of a document containing that word belongs to a certain class. To classify new documents the probabilities for the words within that document can be calculated. Using these probabilities the relative relative probability of the document to belong to a certain class can be calculated. [6]
Although Naive Bayes algorithms are performing quite well they do have some performance issues. To belong to certain classes it is necessary that a document has multiple attributes. Due to assumption of feature independence Naive Bayes is not able to find these class restrictions. Moreover, due to the bag-of-words structure within text is lost. Lastly, the semantic meaning of words is lost, which means that with Naive Bayes you can not detect text with similar meaning that have different words.
Recently new methods have been developed, and these for a large part deal with the problems of Naive Bayes. Mikolov et al. have developed a word embedding framework called Word2Vec. With Word2Vec multi-dimensional word vectors can be created that capture the semantic meaning of words. [7]
There are two regularly used ways of training Word2Vec models: continious bag of words (CBOW) and Skip-Gram. The Skip-gram model is trained by maximizing the probability of words in the neighbourhood of the central word. Which words are considered in the neighbourhood is selected as a parameter of the model. In contrast, the CBOW tries to predict the current word using the words surrounding it. [7]
The great benefit of these implementations is that the semantic meaning is captured. This means that words with a similar meaning are close to each other in the multi-dimensional space. This thus solves the problem of Naive Bayes in which it can not see the similarity between words like "car" and "automobile". These Word embeddings can be fed to convolutional neural nets (CNN). CNN are developed for computer vision, but recently great results have been achieved in text classification. Kim (2014) and Kalchbrenner (2014) have both used this, although they have used different CNN architectures. Both models were specifically aimed at categorizing sentences. Kim outperforms Kalchbrenner due to a

different architecture, which shows the need for testing with the CNN. [3, 4]
The word vectors of the words within a sentence are put after each other. This enables us to do convolutions with linear filters. This linear filter runs over the word vectors and the dot product between the filter an word matrix is taken. Multiple of these filters can be done over the same region, to extract different features. After these filters a bias term is added and one of the activation functions is applied. [9]
Since sentences differ in length these need to be converted to similar length vectors. This can be done with 1-max-pooling filters, which select the most important features per part of the vectors. The different inputs of the various layers is the input for the softmax-function which predicts the outcome. [9]

# 4 Methodology

There are multiple data sources being used. Firstly, the train data of the central government is retrieved from the website www.zoek.officielebekendmakingen.nl. Although the implementation of the scraping process is not yet finished, there is code online to scrape this information from the VU and a manual from the Dutch government. [2, 8]
This means it is not entirely clear what the output of the scrapers will be, however, both the documents itself and the meta data of the documents can be exported as XML files. Within the meta data files the topic is always explicitly within tags. These topics always adhere the taxonomy of the TaxonomyBeleidsagenda from the Dutch central government. This taxonomy consists of topics and subtopics, and the entire experimental set-up is done with both to see what the influence is of more detailed topics.
The second part of data is that retrieved from www.zoek.openraadsinformatie.nl. This website allows Dutch municipalities to publish their documents online. Moreover, the website allows effective querying of this data. This data is also accessible using an API provided by the creators of the website. However, these files do not contain topics. Therefore a subset of a 1000 files will bee manually annotated during the project.
To do the research the files from the central government are pre-processed by transforming every character to lowercase. Moreover, the stopwords, XML- and HTML-tags are removed. Lastly, the documents are lemmatized using the Dutch lemmatizer from NLTK. These files are basically the input for the two different classifiers. The same steps are done on municipality data.
The Word2Vec is now trained using the Gensim implementation and it is trained on all files. Since Word2Vec is unsupervised it is able to also use more than just 1000 of the files from the municipality set. This is done to ensure that the words specifically used by municipalities are also well-presented in the vector. In principle the Word2Vec implementation is only tested with set parameters, but if time allows it multiple settings are experimented with.
For the Naive Bayes algorithm the implementation of SK-Learn is used. Both implementation of Naive Bayes, being Bernoulli Naive Bayes and Multinomial Naive Bayes are tested. Moreover, for both implementation the smoothing parameter is tested with various values.
The architecture of the CNN is to be experimented with. However the ideas of designing CNN architectures of Zhang  Wallace (2016) are taken into account.

They explain the various parameters that should be experimented with, such as the filter region size, activation functions and regularization type. Also, the implementation of these CNN's is done using the Keras library. [9]
Since the research of Kim and Kalchbrenner is specifically aimed at sentences, a methodology needs to be developed to transform the ideas to entire documents. This can be done by splitting up the documents into sentences. Each sentence in the documents is considered as an example of the class of the document. Documents in the test set are predicted a label by taking the topic of how most sentences were predicted.
The two different algorithms are trained on 80 percent of the data. The last 20 percent is used to evaluate performance of the central government data set. The evaluation is done on the basis of effectiveness. This means that for each class the accuracy, recall, precision and F1-measure are calculated. The micro- and macro-average of these metrics are taken among the various classes. This evaluation can be used to compare the functionality of both algorithms. [5]
The last part of the evaluation is used to see how well the knowledge of the training set translates to a different data set. Therefore the models also predict the classes of the 1000 manually labelled set of the municipality data.

# 5 Risk assessment

The following risks could occur during the project:

- **The algorithms both have a significant worse performance on the second data set compared to the first.**
  If this is the case then I should focus on learning how to do transfer learning. This is a scientific field which is specifically aimed at using labeled data sets from another field and transfer that knowledge to unlabeled data. If necessary, this might help.

- **My computer is to slow to process the amount of data or crashes.**
  Luckily the company that I work for can provide a better computer than I have. Even if this does not work I can try to use cluster computing to fix this problem. By making regular back-ups the results of crashes can be minimal.

- **The threshold to categorize is not flexible enough in order to deal with different length of documents, new words, etc.**
  To deal with this the threshold can be made more variable, for example by including document length or type as extra input to select the most logical threshold. Moreover, I use different ways of smoothing and Word2Vec training to make sure new words do not have a big influence on the performance.

- **Since the documents of municipalities are not split into small documents with one topic, the model is not really helpful for categorization.**
  Firstly, this might not be a problem, as if my model can help with one topic documents that is already a good result. Moreover, if I do want it to deal with multitopic documents better, I can try to work on automatically

splitting up documents with technology like topic tilling. This way it can show what parts of the document belong to which topic.

For all these risks, they would take up much extra time. However, as can be seen in the project plan, I do have some time left to spend on these problems.

# 6 Project plan

Below is an overview of the tasks within my thesis. As can be seen, most of the programming tasks are scheduled to be done before the end of May. This means that there is room for delays if necessary. If everything is going according to schedule, then in June I can try to help with the implementation of this research on the website of www.zoek.openraadsinformatie.nl

Table 1: Timetable of the thesis

| **Week** | **Finish** | **Dependence** |
|---|---|---|
| **2/4-8/4** | 1. Loading dataset from central government | - |
| | 2. Loading dataset from municipalities | - |
| **9/4-15/4** | 3. Pre-processing of datasets | 1,2 |
| | 4. Splitting dataset in 2 | 3 |
| **16/4-22/4** | 5. A working implementation of Naive Bayes | 4 |
| **23/4-29/4** | 6. A trained implementation of Word2Vec | 3 |
| **30/4-6/5** | 7. Manually labeling Municipality data | - |
| | 8. A working implementation of CNN | 4,6 |
| **7/5-13/5** | 9. Parameter optimization Naive Bayes finished | 5 |
| | 10. Methodology, literature and introduction section | - |
| **14/5-20/5** | 11. Mid-term progress and evaluation report | 9, 10 |
| **21/5-27/5** | 12. Parameter optimization for CNN | 8 |
| **28/5-3/6** | 13. Final evaluation on municipality data set | 7,9,12 |
| **4/6-10/6** | 14. Results section | 9,12,13 |
| **11/6-17/6** | 15. Discussion and future work section | 14 |
| | 16. Spell-check, lay-out and abstract section | 10,14,15 |
| **18/6-24/6** | 17. Thesis | 16 |
| **25/6-1/7** | 18. Defence of thesis | |

# References


[1] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*, 2017.

[2] The Dutch Government. *Open data webservice van overheid.nl*, 2016.

[3] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[4] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[5] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[6] Andrew McCallum, Kamal Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[8] Kasper Welbers. *Officiele Bekendmakingen scraper*, 2015.

[9] Ye Zhang and Byron Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015.