Classifying Dutch municaplity documents using questions from parliament

A comparative study on the domain adaptivity of Convolutional Neural Networks, Support Vector Machines, Random Forest and Par2Vec

submitted in partial fulfillment for the degree of master of science

Axel Hirschel
10656146

master information studies
data science
faculty of science
university of Amsterdam

2018-06-21

	Internal Supervisor	External Supervisor
Title, Name	Dr Maarten Marx	Tom Kunzler
Affiliation	UvA, FNWI, IvI	Open State Foundation
Email	maartenmarx@uva.nl	tom@openstate.eu

Universiteit van Amsterdam

Contents

Abstract

Thesis requirements

- Your thesis is written in ACM style with two columns (`documentclass[sigconf]acmart`).
- It is maximally 10 pages long, excluding the title page and the appendix, but including references, figures, etc

1 Introduction

Since 2014 a part of all documents produced by Dutch municipalities are published as open data on http://zoek.openraadsinformatie.nl/, a website dedicated to this cause. Over time more municipalities have joined, and in 2018 some provinces have also decided to participate. Although it is possible to search on specific words, more could be done in order to allow users to effectively query relevant information. This research is therefore dedicated to classifying documents with labels which describe their content. Users can select labels and retrieve documents belonging to that label.

These municipality documents have not been classified before, which means that it is not possible to train classifiers on municipality data. The classifiers are thus trained on documents from a related domain, namely questions asked within the Dutch national parlement. Although this data is similar, still classifiers have to adapt to the discrepencies in style, length and vocabulary between the two types of documents. Multiple classifiers are examined in order to evaluate how well the classifiers are suited for domain adaptation.

Many of these classifiers use the bag-of-words (BoW) representation of documents as input. Within BoW documents are represented as vectors containing the counts of words within those documents. Examples of classifers that employ BoW are Support Vector Machines (SVM), Naive Bayes (NB), Logistic Regression (LG) and Random Forest (RF) (Aggarwal & Zhai, 2012).

Other document representations have also been developed which instead create multidimensional vectors for words and paragraphs which capture their semantic meaning. Examples of these multidimensional vectors are Word2Vec (Mikolov, Chen, Corrado, & Dean, 2013) and Paragraph2Vec (Le & Mikolov, 2014). This type of representation is also known as word embeddings and they allow the use of deep neural networks, such as Convolutional Neural Networks (CNN), on text as well. The CNNs employ convolutional filters, which shift over the documents and detect patterns within documents (Kim, 2014) (Kalchbrenner, Grefenstette, & Blunsom, 2014).

The mentioned classifiers are all evaluated in how well they adapt to the new domain. Previous research suggested that algorithms based on word embeddings perform better because the embeddings capture the more nuanced meaning of words and can also use words that are not within the training data (Nguyen & Grishman, 2015) (Mou et al., 2016). Still methods are employed to further increase performance of those classifier after domain shifts. The most suitable technique is importance sampling, which assigns weights to samples during the training based on how much the samples are similar to the test data (Pan & Yang, 2010). The effect of this technique on all classifiers is also studied which leads to the following research questions:

- How can Dutch municipality documents be classified with machine learning techniques?
  - How well can the various classifiers categorize data of parliament?
  - How well does that performance generalize to the municipality dataset?
  - What is the influence of importance sampling on the domain adaptivity?
  - How well do the best methods perform on relatable domain shifts?

Within the next chapter, the related work, relevant literature to this research is examined. Afterwards the methodology is discussed, which also provides information on how the research questions are answered. The results are described next, and provide a detailed overview of the performance of all algorithms. Lastly, the answers to the research questions are formulated and the conclusions are discussed.

2 Related Work

Within this section related research is discussed to show how this research relates to it. The task of classification is discussed first, together with the regularly used classification algorithms. Secondly, it is explained what multilabel classification is and how the algorithms should be adapted. Lastly, the relevance of domain adaptation for this research is examined.

2.1 Classification

Within classification a training set D of length N is present, and each sample has a label from a limited set of labels L assigned to it. This data is used to construct a classification model (also known as a classifier) using classification algorithms, which relate features of samples within D to labels in L. The goal of classification is correctly predict labels for documents based on its content. To evaluate how well the classifiers can do this, a part of D is used as test data which the models attempt to predict accurately (Aggarwal & Zhai, 2012).

2.1.1 Bag-of-words classifiers

All classifiers are thus contingent on labeled train data, which is used to train the classifiers. Often, the data is pre-processed in order to engineer a document representation which contains content which can be used by the classifiers to distinguish relevant features. A common representation is the BoW representation, which is used to transform documents in D into vectors which indicate how many times words of the vocabulary occur within the documents (Aggarwal & Zhai, 2012).
One of the algorithms that uses bag-of-words as input is the multinominal naive bayes classifier, which uses probabilities of words occuring within a specific topic in order to classify unseen documents (Aggarwal & Zhai, 2012).
Another frequently used algorithm for text classification is the decision tree classifier, which is an algorithm that establishes a hierarcal division based on textual features. These splits are based on feature spaces which have a more skewed distribution of the classes. Multiple trees can be created as an ensemble, each on part of the data, to prevent overfitting to the train data and these are called random forest classifiers. Although the algorithm and its outcome are easily understandable its performance is often worse than other methods (Li & Jain, 1998).
Logistic regression is also used as classifier, and it employs a statistical approach for classification. Its prediction is based on the multiplication of parameters and variables. During training these parameters are optimized with gradient descent in such a way that as many samples are classified correctly with high probability. To predict unseen samples it takes the log of the multiplication, which

when rounded ends up as a 0 or 1 (Aggarwal & Zhai, 2012).
The last method based on the bag-of-words implementation is the SVM, which has been the state-of-the-art for many years. Within SVM hyperplanes are constructed that split datapoints within the multidimensional space. It is argued that SVM can perform well on textual data, since few features are relevant though those which are relevant correlate. This allows SVM classifiers to easily distinquish between various classes (Joachims, 2001).
Although SVM and logistic regression perform well, they are naturally binary classifiers. This means that instead of having any amount in labels within L merely two labels can be used. However, this problem can be easily solved by constructing an one-vs-rest classifier, which produces a new binary model for each label against all other labels. This means that all samples of one label are considered as positive samples and all the other samples without the label as negative samples for the training of the model. Eventually labels are assigned by applying all classifiers to new samples and predicting a label from L for the classifier which reports the highest score (Aggarwal & Zhai, 2012).
The algorithms based on bag-of-words have performed well on many classification tasks. However, the bag-of-words representation has a number of detriments. Foremost, the representation is unable to see the relation between "strong" and "powerful" as these words are equally far apart as they are different from any other word such as "flower". Moreover, the representation treats documents as a collection of words instead of an ordered sequence of words. These two problems refrain algorithms to identify specific patterns and nuances in texts.

2.1.2 Word embedding classifiers

Recently, new document representations have been developed which use word embeddings. Word embeddings are multidimensional vectors that represent the semantical meaning of words. These embeddings are created using a neighbourhood approach, and therefore words that appear in similar contexts are located near each other within the multidimensional space. (Mikolov et al., 2013) Documents can then be represented by ordering these word vectors in the same sequence as original sentences.
This representation allows successful employment of deep neural networks for text classification, and two distinct types can be distinquished: Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). Although both types have achieved state-of-the-art results within various natural language processing tasks, CNNs are expected to have better performance on classification tasks where pattern recognition is vital (Yin, Kann, Yu, & Schütze, 2017). Moreover, CNNs can be trained faster than RNNs, because it can be parallized on the GPU. Since keyphrases and similar structures seem to be important for this specific task, and because computational speed is important, CNNs have been further explored.
CNNs employ filters that are applied to local features and this has originally been used for processing image data. However, recent research has shown that CNNs can also be used for natural language processing tasks. Kim et al. (2014) show that CNNs with Word2Vec-embeddings achieve excellent results, which suggests that the embeddings are good feature extractors. Their architecture consist of multiple pooling layers and 1D-convolutional layers and then two

fully connected layers which produce the final output. They also experiment with multiple filter sizes in the convolutional layers, which also increases performance. Their last finding is that dynamically adjusting the embeddings boosts performance in many tasks.
CNNs are limited to classifying documents with the same length, and most researchers solve this problem with padding and splitting documents. This means that a fixed length is chosen as parameter, and sentences that are longer are cut off at that point.Vectors with only zeros, and thus meaning, are appended to sentences that are shorter in order to get to the fixed length (Kim, 2014) (Zhang & Wallace, 2015).
Another approach to deal with different sentence lengths is the Dynamic CNN, which allows different sentence-sizes as input. After convolution layers a dynamic k-max-pooling layer, which computes the amount of features pooled to the next layerly based on the sentence's length. Only within the last pooling layer a fixed number of features is pooled and used within the last dense activation layer (Kalchbrenner et al., 2014).
This architecture is only suitable for sentences but has been extendend to entire documents as well. The adapted algorithm uses the output from the last layer of the Dynamic CNN employed on each sentence. Then based on the length of the document these features are then once again used in a Dynamic CNN (Denil, Demiraj, Kalchbrenner, Blunsom, & de Freitas, 2014).

2.2 Adaptions for multilabel classification

Much of the multiclass classification challenge is similar for multilabel classification too. However, instead of giving one label from the limited set L now any amount of labels can be assigned, including no label at all. This means that the training data D also contains samples with varying numbers of labels assigned to it. All previously discussed algorithms can be used for multilabel classification as well when adapted.
The outcome of the algorithms is already a predictions between 0 and 1 for each label. So instead of chosing the label with the highest score, a threshold can be defined, such as 0.5. Now samples are considered part of a class if the independent probability of it belonging to that class is higher than the threshold.
In addition to this, for CNNs the final activation of the last connected layer should be changed from "softmax" to "sigmoid" as "sigmoid" calculates independent probabilities for each label. Many people also employ a different loss function to optimize multiclass neural networks, such as binary cross-entropy instead of multiclass cross-entropy (Nam, Kim, Mencía, Gurevych, & Fürnkranz, 2014) (Sterbak, 2017).

2.3 Domain adaptation

Transfer learning is the discipline dedicated to transfering models from one task to another. One typical problem of transfer learning is transductive transfer learning or domain adaptation, which formally can be described as learning a task on the source training set D_S. The goal is to create a model that minimizes the error of the same task on a target test set D_T (Pan & Yang, 2010).
One of the methods to do domain adaptation is a transfer of feature representations, and the use of Word2Vec can already be seen as an example of this type

of adaptation (Nguyen & Grishman, 2015). Other examples of feature representation transfers are some dimensionality reduction algorithms employed on the bag-of-words representations and finding domain specific features and removing those (Pan & Yang, 2010).
Another of the feature representations transfers is the autoencoder, which is an unsupervised neural network. The idea of the autoencoder is to find a lower dimensionlaity space with which you can construct the original sample. However, the local structure is lost during the process, which is why it is less suited for combination with CNNs (Ganin & Lempitsky, 2014).
Another type of transfer suited for domain adaptation is transferring the knowledge of instances. The most common way to do this is with importance sampling, which implies that training samples in D_S that are very alike to test samples in D_T are more important during training. This is done by constructing a classifier which recognizes whether samples come from D_S or D_T. During training on D_S for the classifier of the eventual task, samples in D_S are weighted based on how much the previously constructed classifier predicts them to be part of D_T (Pan & Yang, 2010).

3 Methodology

Within this research the final goal is to construct a classifier which is able to classify municipality documents correctly. In order to do this, a multitude of algorithms are examined. Also, the influence of the domain adaptivity and importance sampling are researched. This section introduces the datasets, experiments and further detail about the implementation of the algorithms.

3.1 Description of the data

Two datasets have been used within this research which both include documents from the government of the Netherlands. The first set, called PAR_ALL, consists of over 50,000 questions and answers that have been asked within the Dutch national parliament during 2001-2017 and these questions were scraped from www.zoek.officielebekendmakingen.nl. The content of these questions and answers ranges from critical examination of proposed laws to requests of more information about ongoing affairs currently within the news. The second set consist of 20,000 documents from Dutch municipalities, such as items on the agenda and notifications of commissions. This set is retrieved with an API of www.zoek.openraadsinformatie.nl and is called MUN_ALL.
Although both sets are political and Dutch, they do vary in content. One of the big differences is that the set of the national government only entails questions with answers, whereas the municipality dataset consist of a greater variety of document types. The themes discussed are also different because within municipalities only local policy is discussed such as the construction of local infrastructure and the re-allocation of local sport clubs. This is in contrast to the parliament, because there broader themes such as criminal law and measure for social security are examined.
All of the parliament data is annoted with any number of labels denoting their content. These labels have 2 levels of detail; one broad category such as healthcare or law and one detailed category such as healthcare for the elderly or crim-

inal law. For this research only the 17 broad labels are used, the detailed 118 labels are dropped. The municipality set was not labeled, but for this research manually a part was annotated using the framework of the TaxonomieBeleidsagenda. This taxonomy has also been used to annotate the parliament data. The labeled municipality dataset is named MUN_LABELED.

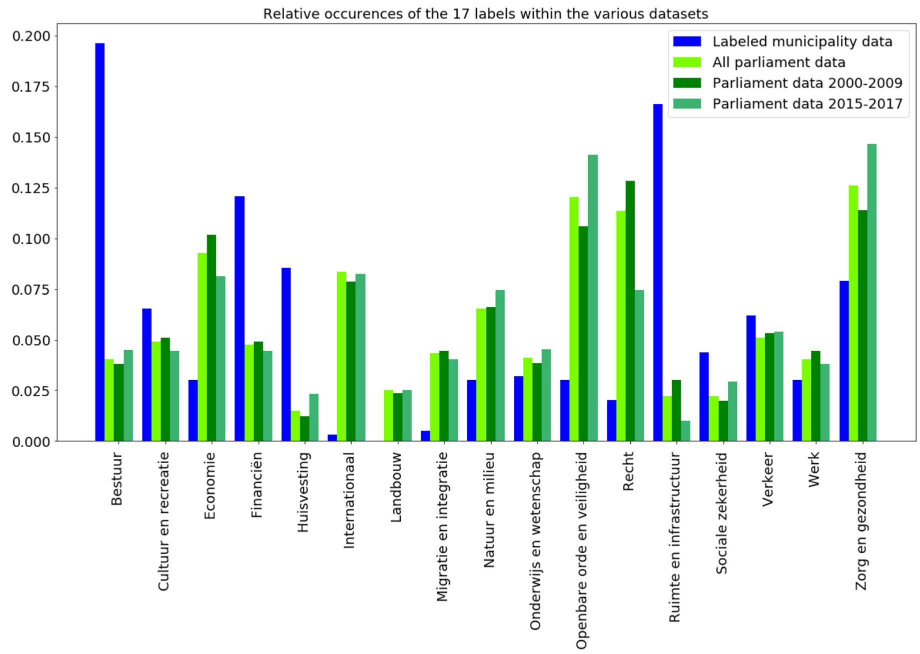

Figure 1: Relative occurences of labels within various datasets

Within Table 1 the datasets are summarized, as it shows some characteristics of these sets. Note that for one experiment the dataset with questions of parliaments has been split in two parts, based on when these questions have been asked. The first set consists of all questions from 2001-2009 (PAR_EARLY) and the second set composed of the questions of 2015-2017 (PAR_LATE). Figure 1 shows the relative occurences of labels within the datasets.

Table 1: Summary of the datasets used within this research

Name Dataset	PAR_ALL	PAR_EARLY	PAR_LATE	MUN_ALL	MUN_LABELED
Amount samples	52397	28063	8202	20886	439
Average amount words	688.3	648.2	834	119.4	136.8
Standard deviation words	429.3	382	554.0	53.2	106.8
Average amount labels	1.62	1.85	1.32	-	1.36
Standard deviation labels	0.76	0.83	0.52	-	0.89

3.2 Experiments

The main goal of this research is to correctly classify MUN_ALL data, but it also examines the effects of the domain shift and domain adaptivity algorithms.

The algorithms and their parameters are explained in the next section, as this subsection explains how the research questions are answered.
Firstly, it is checked how well the individual algorithms perform on merely the parliament data without any difference between the source and target domain. This means the PAR_ALL dataset is used and shuffled. Also, around 70 percent of the data is used for training, 15 percent for selecting the best hyper-parameters and 15 percent for the final evaluation. Just like with all the following experiments, the performance of algorithms is based on the micro-average of the F1 score, which balances recall and precision.
Secondly, the models are trained on the PAR_ALL data as source domain and tested on the MUN_LABELED data. This experiment is done twice per algorithm, as it is executed once with and once without importance sampling. For the importance sampling a logistic regression classifier is used which uses 10,000 samples from PAR_ALL and all of MUN_ALL. This classifier is then employed to weigh the remainder of the training data.
Thirdly, an experiment very similar to the second experiment is carried out. However, the data is different, because now the source domain is all PAR_EARLY and the target domain is the parliament data from PAR_LATE. This experiment is executed to examine how important the domain shift is.

3.3 Algorithms

Within this research newer CNN-classifiers are compared to traditional classifiers such as Support Vector Machines (SVM), Logistic Regression (LR), Random Forest (RF) and Multinominal Naive Bayes (NB). For all these implementations the input is a bag-of-words representation of the various documents. The implementation of Scikit-Learn is employed for these algorithms and the optimal hyper-parameters are chosen based on a grid-search. Moreover, when needed, the one-versus-all classifier is used to make the classifiers multi-class. To ensure multi-label outcomes multiple thresholds for prediction are experimented with.
Instead of bag-of-words CNN uses word embeddings as input. This research experiments with two embeddings, namely pre-trained embeddings retrieved trained on a variety of Dutch resources (Tulkens, Emmery, & Daelemans, 2016) and self created embeddings using Word2Vec-implementations from Gensim trained on PAR_ALL and MUN_ALL.
One of the employed CNN within this research is similar to earlier architectures (Kim, 2014). This means that an embedding layer is used to transform sentences to a multi-dimensional space using one of the two Word2Vec-embeddings. Both embedding spaces have been tested with static and non-static initializations which indicates whether the embeddings can alter during training.
Thereafter three convolutional layers and three max-pooling layers are alternated between. For the convolutional layers multiple filter sizes have been tested and in addition also multiple filter sizes in one layer are experimented with. Then the multidimensional is flattened and a dropout layer is used to prevent overfitting within the network. In some architectures also L1-regularization has been used, however, later research demonstrated the minimal effect of this regularization (Zhang & Wallace, 2015).
The last two layers are fully connected layers in order to gain the final prediction. Since the classification task is multilabel, binary crossentropy is used as

loss function in combination with the sigmoid function as activation within the final dense layer (Nam et al., 2014) (Sterbak, 2017). The output of the sigmoid function is a number between 0 and 1 per label, and multiple thresholds are tested in order to determine when to predict a certain label. All the parameters of this standard version of CNN are listed in Table 2

Table 2: Parameter and values within standard CNN

Parameter	**Values**
Word2Vec Model	Pre-trained, trained on corpus
Word2Vec Initialization	Dynamic, Static
Amount of filters	1,3
Filter sizes with one filter	3,5,7,11
Filter sizes with three filters	[2,3,4],[3,4,5],[7,8,9]
Threshold of prediction	0.3,0.4,0.5,0.6,0.7

An addition of this research is a slight variation to Kim's architecture which splits the input into smaller blocks of length 200 instead of choosing a pre-defined length of the input sentences. Training is done on smaller blocks of input and when predicting sentences each of the individual blocks of the sentence are classified individually. These predictions are aggregated into one prediction per sentence, either by summing or using the max value of the predictions per block. Then once again predictions above a certain threshold are predicted to belong to that label, and those below that threshold are considered not to have that label. The parameters for this version are similar to those of the Kim's standard implementation, but all the extra parameters are listed in Table 3.

Table 3: Extra parametera and values within CNN-split

Parameter	**Values**
Input size	200
Aggregation	sum, max

4 Evaluation

Met een subsectie voor elke deelvraag.

In hoeverre is je vraag beantwoord?

Een mooie graphic/visualisatie is hier heel gewenst.

Hou het kort maar krachtig.

5 Conclusions

Hierin beantwoord je jouw hoofdvraag op basis van het eerder vergaarde bewijs.

Table 4: Performances of algorithms on central government data

Model	**Accuracy**	**Micro-F1**	**Micro-Recall**	**Micro-Precision**
Random Forest	0.27	0.45	0.30	0.92
SVM	0.55	0.79	0.72	0.84
Logistic Regression	0.54	0.77	0.72	0.84
Multinominal Naive Bayes	0.06	0.09	0.05	0.98
CNN	0.40	0.71	0.71	0.70

6 Acknowledgements

I would like to thank everyone from Open State foundation for their support and friendliness during my internship. I have always gone to OSF with great pleasure due to the good political discussions. Moreover, I have greatly appreciated the help of Maarten Marx and Tom Kunzler, as they constantly proposed new ideas and critically looked at mine. Lastly, thanks to my parents Diederik and Moniek, siblings Max and Iris, girlfriend Amber and friends Joppe, Simon, Ethan and Josse for keeping me motivated and supporting me during the entirety of my studies.

References

Aggarwal, C. C., & Zhai, C. (2012). A survey of text classification algorithms. In *Mining text data* (pp. 163–222). Springer.

Denil, M., Demiraj, A., Kalchbrenner, N., Blunsom, P., & de Freitas, N. (2014). Modelling, visualising and summarising documents with a single convolutional neural network. *arXiv preprint arXiv:1406.3830*.

Ganin, Y., & Lempitsky, V. (2014). Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*.

Joachims, T. (2001). A statistical learning learning model of text classification for support vector machines. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (pp. 128–136).

Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning* (pp. 1188–1196).

Li, Y. H., & Jain, A. K. (1998). Classification of text documents. *The Computer Journal*, *41*(8), 537–546.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., & Jin, Z. (2016). How transferable are neural networks in nlp applications? *arXiv preprint arXiv:1603.06111*.

Nam, J., Kim, J., Mencía, E. L., Gurevych, I., & Fürnkranz, J. (2014). Large-scale multi-label text classification—revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 437–452).

Nguyen, T. H., & Grishman, R. (2015). Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 2: Short papers)* (Vol. 2, pp. 365–371).

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

Sterbak, T. (2017). *Guide to multi-class multi-label classification with neural networks in python.* `https://www.depends-on-the-definition.com/guide-to-multi-label-classification-with-neural-networks/`. (Accessed: 2018-05-23)

Tulkens, S., Emmery, C., & Daelemans, W. (2016, may). Evaluating unsupervised dutch word embeddings as a linguistic resource. In N. C. C. Chair) et al. (Eds.), *Proceedings of the tenth international conference on language resources and evaluation (lrec 2016).* Paris, France: European Language Resources Association (ELRA).

Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923*.

Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.