The Future of Data Analysis in Smart Cities - Research Article

International Journal of Distributed Sensor Networks

International Journal of Distributed Sensor Networks
2018, Vol. 14(11)

DOI: 10.1177/1550147718811827
journals.sagepub.com/home/dsn
SAGE

Automatic political discourse analysis with multi-scale convolutional neural networks and contextual data

Aritz Bilbao-Jayo and Aitor Almeida

Abstract

In this article, the authors propose a new approach to automate the analysis of the political discourse of the citizens and public servants, to allow public administrations to better react to their needs and claims. The tool presented in this article can be applied to the analysis of the underlying political themes in any type of text, in order to better understand the reasons behind it. To do so, the authors have built a discourse classifier using multi-scale convolutional neural networks in seven different languages: Spanish, Finnish, Danish, English, German, French, and Italian. Each of the language-specific discourse classifiers has been trained with sentences extracted from annotated parties' election manifestos. The analysis proves that enhancing the multi-scale convolutional neural networks with context data improves the political analysis results.

Keywords

Supervised classification, convolutional neural networks, online political discourse, sentence classification

Date received: 4 January 2018; accepted: 6 October 2018

Handling Editor: Antonino Staiano

Introduction

A large part of citizen participation in politics occurs in the written media: newspapers, online forums, social media and so on. Furthermore, the rise of new communication technologies has allowed citizens to become participants in the construction of the political agenda, forcing political parties to use more direct means of communication than the mainstream press and media. For this reason, new ways to analyse text with political content in an automated way must be found.

For that purpose, the authors propose a multidisciplinary approach, to build a text categorization classifier using manually annotated sentences from political manifestos and use it to analyse the underlying discourse on the claims of both the citizens and the politicians. Therefore, the authors combine the political science knowledge from the social scientists involved in the annotation of the political manifestos with natural language processing, in order to be able to process large quantities of data and study how the political discourse evolves online and off-line.

Since analysing the political discourse is a complex and time-consuming task, the authors propose in this research work a new approach for automatically analysing it in texts using multi-scale CNNs with word embeddings and using two types of context data as extra features: the previous sentence in the manifesto and the political leaning of the person. These extra features drastically improve the performance of the classifier as it will be demonstrated in the evaluation section.

This article is organized as follows. Section 'Related work' offers an overview of previous related work on

DeustoTech, Fundación Deusto, Universidad de Deusto, Bilbao, Spain

Corresponding author:
Aritz Bilbao-Jayo, DeustoTech, Fundación Deusto, Universidad de Deusto, Avda. Universidades, 24 Bilbao 48007, Bizkaia, Spain.
Email: aritzbilbao@deusto.es

the use of political manifestos for automated text analysis, sentence classification and social network analysis. Section 'Automated analysis of the political discourse' describes the research framework: used training data, performed data pre-processing and used architecture. Section 'Evaluation' explains how the developed classifier has been evaluated and its results. Finally, section 'Conclusion and future work' draws some conclusions and proposes further work.

Related work

Even though historically there has not been relevant works concerning the automatic codification of political manifestos and the use of this codification schema for the analysis of other types of political texts besides political manifestos, recently there have been some authors who have worked in this field.

In 2016, Zirn et al.[1] presented an approach for automated classification of political manifestos. The authors trained and validated their approach using six US manifestos (Republican and Democrat manifestos from 2004, 2008 and 2012 elections). Instead of using the original codification schema of 56 categories, they only worked with the seven policy domains. Their approach consisted in combining two different classifiers: one including only information about the sentence and the second one a binary classifier which predicts if two adjacent sentences have the same code or not. Finally, Zirn et al. combine these two classifiers with information about the topic distribution in the corpus (rules representing the conditional probabilities of domain transitions). They reached the best performance combining the previously explained two classifiers and using a transition rule which indicates that consecutive sentences have the same domain label.

However, this approach cannot be applied to other types of political texts since Zirn et al. used the distribution of topics, sequences of topics and topic transitions in the manifestos as an extra feature. For example, the structure of the text to be analysed can be completely different to the structure of a manifesto where sections or subsections with similar topics are discussed together and therefore topic transitions are easier to predict. In the presented approach, this is solved using the previous sentence or the political leaning of the person who has written the text are used instead of features such as topic transitions or distribution of topics.

Nanni et al.[2] used annotated political manifestos and speeches in order to analyse the speeches from the 2008, 2012 and 2016 US presidential campaigns in the seven main domains defined by the manifestos project. The main difference between Nanni et al.'s work and this research is that first, only annotated manifestos have been used as training data (while Nanni et al. used annotated speeches too) to later apply this knowledge to another texts, and second, our research work uses more than the seven main domains defined by the manifestos project.

In 2017, Glavaš et al.[3] proposed an approach for cross-lingual topical coding of sentences from electoral manifestos using as training data, manually coded manifestos with a total of 77,500 sentences in four languages: English, French, German and Italian. Using convolutional neural networks (CNNs) with word embeddings and inducing a joint multilingual embedding space, Glavas et al. obtained better results than monolingual classifiers in English, French and Italian. However, they achieved worse results with their multilingual classifier than a monolingual classifier in German. According to Glavas et al., this happened because they had two decades of German political manifestos covering a wider span of political issues with a high language variation.

Regarding other approaches of automated analysis of political text, some researchers have detected the polarity of raw text using natural language processing techniques. Iyyer et al.[4] designed a recursive neural network in order to identify the political polarity of a sentence. Authors used two datasets to evaluate their model: an existing one and another one annotated by them by means of crowdsourcing. Authors were able to identify most conservative or liberal *n*-grams and detect bias more accurately. Similar works have been conducted on Twitter. Rao et al.[5] used word embeddings and long short-term memory (LSTM) recurrent neural networks (RNNs) in order to classify twitter messages as democratic or republicans. Author established the ground truth using Twitter Lists, where users are categorized in different groups (democratic or republican) by other users.

However, some research works[6] have criticized the approach of raw machine learning classification techniques whose objective is to correctly classify the highest percentage of individual documents. According to Hopkings et al., those classification techniques can lead to highly biased category distributions which may derive in inaccurate analysis. Political discourse analyses are not interested in whether a sentence is correctly classified or not, and its goal is to globally analyse how the discourse evolves, which are the main topics of interests and so on. In other words, political discourse analyses are interested in the aggregate proportions of the topics. To solve this problem, authors proposed to use the misclassification probabilities in order to correct the predicted category proportions.

In regard to the natural language processing techniques used in the area, most of them have already been mentioned above: different size *n*-grams, lexicons, web-derived polarity lexicons,[7] word embeddings and LSTMs or raw RNNs. However, the model is based on

the work made by Kalchbrenner et al.[8] and Yoon Kim,[9] where Kim presented a model for sentence classification tasks with CNNs trained on top of pre-trained word embeddings. The model improved state-of-the-art benchmarks on four out of seven tasks, sentimental analysis and question classification among them. Furthermore, different researchers have based their work on this model, building new models with different additions depending on the task[10,11] or Zhang and Wallace[12] giving some guidelines about how those hyperparameters should be chosen in this kind of models.

Automated analysis of the political discourse

Training data

The main reason why the authors use political manifestos as training data is that political scientists have been manually annotating political parties' manifestos for years in order to apply content analysis methods and perform political analyses. Some examples of the performed political analyses are the comparison between different types of manifestos (national and European level manifestos by Wüst and Volkens[13]), analysing how much parties emphasize certain topics and which are their positions in some concrete topics depending on the elections context,[14] how these positions have evolved over the years[15] or to estimate policy positions for political parties on left-right scales using measures such as RILE scale[16] or other alternatives.[17]

The precursors of this methodology were the Manifesto Project, formerly known as the manifesto research group (MRG) and comparative manifestos project (CMP).[18] In 2001, they created the Manifesto Coding Handbook[19] which has evolved over the years. The handbook provides instructions to the annotators about how political parties' manifestos should be coded for later content analysis and a category scheme that indicates the set of codes available for codification. Nowadays, the category scheme for manifestos annotation consists in 56 categories grouped into seven major policy areas[20] (see Table 1): *external relations, freedom and democracy, political system, economy, welfare and quality of life* and *social groups*.

Due to the high number of available categories for annotation, manifestos annotation is not an easy task even for trained political scientists as Mikhaylov et al.[21] demonstrated, where after examining several annotators' intercoder reliability in two manifestos they concluded that the coding process is highly prone to misclassification. In order to annotate these types of political texts, annotators need a previous training to ensure maximum reliability and comparability of data which have been annotated by different people. To do so, annotators have to learn a set of coding rules and when to use each of the codes of the coding scheme. Moreover, these annotators must be under the supervision of an expert annotator throughout the whole training process in order to clarify any doubts that may arise.

Political discourse classifier

In order to address the text classification task and demonstrate that context information such as the previous sentence or the political leaning of the person who has written the sentence can improve the performance of an automated political discourse analyser, the authors have built a classifier using CNNs with Word2Vec word embeddings. Recently, CNNs have achieved excellent results in several text classification tasks.[9,22,23] This, combined with the fact that this type of classifier allows the use and fine-tuning of word embeddings, allowing us to extract knowledge from non-annotated texts, is the reason why this approach has been adopted.

Semantic embeddings for sentence representation. The inputs of the model are the sentences which are fed to the neural network as sequences of words. These sequences have a maximum length of 60 words. The maximum length has been decided after an analysis of the corpus sentences' length and detecting that most of the sentences have 60 or less words. Therefore, the CNNs used for this task have an input size of 60.

However, the words are not inserted as raw text to the CNN. The words are inserted as word vectors, a multidimensional representation of each word. Those word vectors have been generated using the Word2Vec[24] unsupervised learning algorithm, which produces a large vector space having non-annotated raw text as input. Using Word2Vec, each word of the corpus is positioned in a multidimensional vector space taking into account its context (its surrounding words). Word's position in the N-dimensional vector space (being N the number of dimensions of the defined vector space) is used as its representation (word vector).

For example, given a sentence $S = [w_1, w_2, w_3, \ldots, w_n]$ (n is the number of words in the sentence), the context of the word w_i would be $Context_k(w_i) = [w_{i-k}, \ldots, w_{i-1}, w_{i+1}, \ldots, w_{i+k}]$ where $2k$ is the window size for the context. Then, the log-likelihood is maximized in order to compute the word vector of each word

$$J_{ML} = \log p(w_i | Context_k(w_i))$$

The authors have chosen 300 as word vectors' size (number of dimensions of the multidimensional space where the words are positioned) to take advantage

Table 1. Categories in seven policy domains.[20]

Domain 1: External Relations
101 Foreign Special Relationships: Positive
102 Foreign Special Relationships: Negative
103 Anti-Imperialism: Positive
104 Military: Positive
105 Military: Negative
106 Peace: Positive
107 Internationalism: Positive
108 European Integration: Positive
109 Internationalism: Negative
110 European Integration: Negative
Domain 2: Freedom and Democracy
201 Freedom and Human Rights: Positive
202 Democracy
203 Constitutionalism: Positive
204 Constitutionalism: Negative
Domain 3: Political System
301 Decentralization: Positive
302 Centralization: Positive
303 Governmental and Administrative Efficiency: Positive
304 Political Corruption: Negative
305 Political Authority: Positive
Domain 4: Economy
401 Free-Market Economy: Positive
402 Incentives: Positive
403 Market Regulation: Positive
404 Economic Planning: Positive
405 Corporatism: Positive
406 Protectionism: Positive
407 Protectionism: Negative
408 Economic Goals
409 Keynesian Demand Management: Positive
410 Economic Growth
411 Technology and Infrastructure: Positive
412 Controlled Economy: Positive
413 Nationalization: Positive
414 Economic Orthodoxy: Positive
415 Marxist Analysis: Positive
416 Anti-Growth Economy: Positive
Domain 5: Welfare and Quality of Life
501 Environmental Protection: Positive
502 Culture: Positive
503 Equality: Positive
504 Welfare State Expansion
505 Welfare State Limitation
506 Education Expansion
507 Education Limitation
Domain 6: Fabric of Society
601 National Way of Life: Positive
602 National Way of Life: Negative
603 Traditional Morality: Positive
604 Traditional Morality: Negative
605 Law and Order
606 Civic Mindedness: Positive
607 Multiculturalism: Positive
608 Multiculturalism: Negative
Domain 7: Social Groups
701 Labour Groups: Positive
702 Labour Groups: Negative
703 Agriculture and Farmers
704 Middle Class and Professional Groups: Positive
705 Minority Groups: Positive
706 Non-Economic Demographic Groups: Positive
000 No meaningful category applies

of already pre-trained Word2Vec models in several languages published by Kyubyong Park.[25] However, for the Spanish Word2Vec model, the authors have taken advantage of a 300 Word2Vec model created with 3 billion words.[26] In the case of the English Word2vec, the authors have used a Word2vec model pretrained with Google News corpus (3 billion running words).

Once all the word vectors have been computed, the following is performed. First of all, a dictionary D where words are mapped to indexes $(1, \ldots, |D|)$ is created, being $|D|$ the number of unique words in the corpus and saving the 0 index for padding purposes. Therefore, the input sequences of words are transformed into a sequence of 60 indexes, padding with 0s those phrases which have a length of less than 60 words, since CNNs do no admit different sizes for the input data once the input size has been set. Then, these indexes are transformed into their corresponding word vector using an embedding layer or matrix. This embedding matrix acts as a dictionary: having the word index, the embedding matrix returns the corresponding word vector which has been previously computed. The embedding matrix is generated concatenating all the vector representations of all the existing words in D, creating a matrix $W \in \mathbb{R}^{|D| \times d}$, where d represents the vector size of the word embeddings which is 300 in this research.

Therefore, the embedding matrix works as a dictionary whose input is the word index and its output is the vector representation as it can be seen in Figure 1. The embedding matrix can be both static and non-static. On one hand, the static approach treats all the word vectors as static values which cannot change through the training process and therefore all those weights per word defined by Word2Vec remain constant through all the training. On the other hand, a non-static embedding matrix changes as the training process evolves since the word vectors are interpreted as new parameters for the

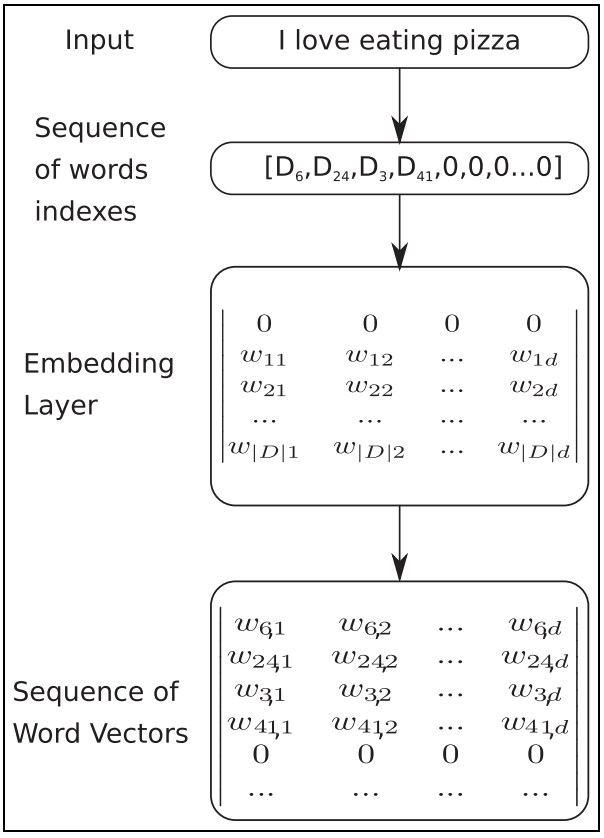

Figure 1. Raw text transformation into a matrix of word vectors.

model and they are fine-tuned during the training. The authors have opted for the non-static word-embedding since it improves model's performance.[9]

Convolutional model for political discourse classification. Once the phrase has been transformed from a sequence of words to a sequence of word indexes and finally to a sequence of word vectors (see Figure 1), the phrase can finally be fed into the CNN, since the sequence of word vectors are in fact a matrix which dimensions are $60 \times d$ where convolution operations can be performed.

The CNNs are a specific type of neural networks with neurons, weights and biases where convolution operations are performed and had been traditionally used for recognizing visual patterns directly from images (pixels).[27] However, as previously has been explained, in recent years, CNNs have also been used for text classification. In brief, convolution operations consist in moving different windows (filters made of neurons) with different sizes (filter sizes) analysing different regions in the matrix (an image or a list of word vectors) to extract different features. The proposed model performs convolution operations with three different filter sizes, batch normalization[28] and ReLU as the activation function. Batch normalization acts as an extra regularizer and increases the performance of the model.

The defined filter sizes are $2 \times d$, $3 \times d$ and $4 \times d$. These filter sizes can be compared to a selection of *n*-grams: bigrams, trigrams and fourgrams, respectively. As it can be seen from Figure 1, each row in the matrix represents a word and therefore a filter size of $2 \times d$ will take the whole width of all the possible bigrams of the sentence, filter size of $3 \times d$ all the possible trigrams and filter size of $4 \times d$ all the possible fourgrams. This is how a single filter would work, however, as it is stated in Zhang and Wallace,[12] multiple filters should be used in order to learn complementary features. The model has 100 filters per different filter size. Once a filter has been applied, a feature map is generated. Therefore, a different feature map is generated per applied filter.

After the convolutional layer, there is a pooling layer whose objective is to reduce the dimensionality of the incoming data. There are different pooling strategies: average pooling, max-pooling, 1-max-pooling and so on. The authors have opted for the 1-max-pooling[29] strategy since it has been proved in Zhang and Wallace[12] that is the best approach for natural language processing tasks. It captures the most important feature (the highest value) from each of the feature maps. Therefore, the output of the pooling is a feature per filter which are later concatenated into a feature vector.

Next, a dropout[30] rate of 0.5 is applied as regularization in order to prevent the network from over-fitting, followed by a fully connected layer with ReLU as the activation function and batch normalization. Then, a 0.5 dropout is applied.[12] Finally, the softmax function computes the probability distribution over the labels.

To sum up, the flowchart of the model is the following (see Figure 2):

1. The phrase and the previous phrase are inserted as a list of words.
2. The embedding matrix replaces each word with its corresponding word vector, generating a sequence of word vectors from a sequence of words.
3. The phrase and the previous phrase are fed into two different structures of CNNs with 100 filters and filter sizes of $2 \times d$, $3 \times d$ and $4 \times d$.
4. The 1-max-pooling reduces the dimensionality of the feature maps generated by each group of filters.
5. Once their dimensionality has been reduced, the feature maps generated from the phrase and the previous phrase are concatenated.
6. If the political party to which the text belongs to is used, its one hot codification is concatenated with the feature extracted from the CNNs.

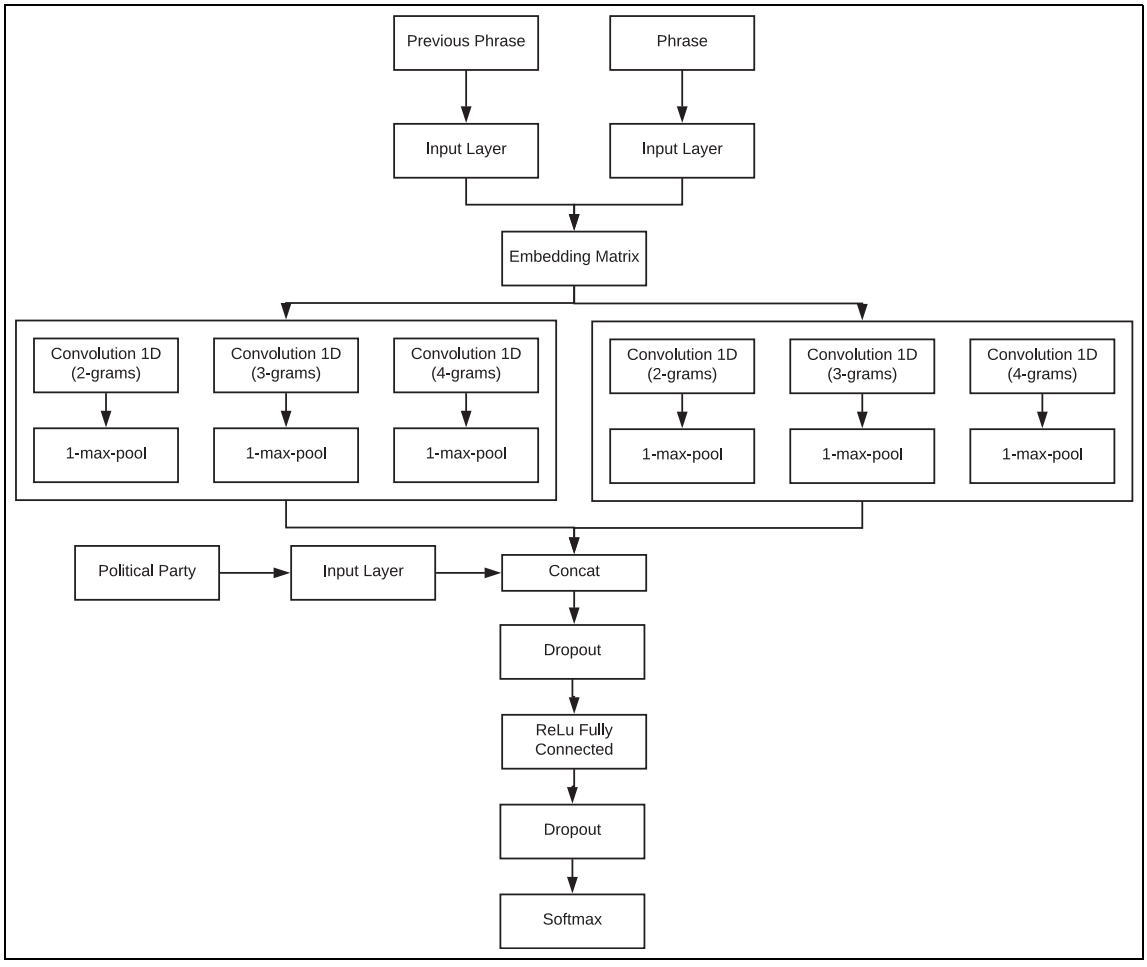

Figure 2. Multi-scale CNN architecture for political discourse analysis.

7. A dropout rate of 0.5 is applied.
8. Then, to classify the phrases to the objective political topics, a fully connected layer with ReLu as activation function is used.
9. A dropout rate of 0.5 is applied.
10. The layer with softmax computes the probability distribution over the labels.

The categorical cross-entropy loss has been used as training objective function since it supports multiclass classifications. Regarding the optimizer, the optimization has been performed using Adam[31] with the parameters of the original manuscript.

Contextual data as new inputs. Regarding how the authors have added the previous phrase to the model as a new input in order to improve the performance of the model as it will be demonstrated in the next section, two different approaches have been tested:

- As a second channel in the convolutional layers: when convolution operations are applied to text only one channel is used, the channel where the sentence to be classified is inserted. However, in this approach, a second channel is used to insert the previous sentence. Therefore, the convolution operations are applied to two channels.
- Replicating for the previous phrase, the same convolution-pooling process is used in the actual phrase as it can be seen in Figure 2.

With regard to the political leaning, it is represented with a one-hot encoding scheme. Therefore, the size of the one-hot encoding will vary depending on the number of political parties whose manifestos has been used to train the model. Then, the authors have concatenated a one-hot-encoding representation of the political party which the phrase belongs to, to the feature maps obtained after the convolutions as it can be seen in Figure 2.

Table 2. Datasets' statistics.[32]

Language	# Manifestos	# Sentences
Spanish	45	78,221
Finnish	14	7872
Danish	36	7559
English	115	86,500
German	78	95,833
French	21	8301
Italian	15	4151

Evaluation

Dataset

The experimentation performed in this research work has been done using Manifesto Project's public corpus of annotated political manifestos.[32] In particular, political manifestos written in Spanish, Finnish, Danish, English, German, French and Italian have been used (see datasets' statistics in Table 2).

Results and discussion

The proposed approach has been evaluated using seven different datasets (one per language) and two types of codifications for the political discourse: domains and subdomains:

- Domains: the seven major policy areas in the Manifestos' project codification schema.
- Subdomains: the 56 categories defined by the Manifestos' project codification schema

As it can be seen from Table 3, the distribution of samples over the seven domains is imbalanced in the seven languages. When domains like External Relations and Freedom and Democracy usually have around 8% and 7% of the samples, respectively, Economy and Welfare and Quality of Life have more than 60% of the samples. With regard to the subdomains, the imbalanceedness is even worse since the already imbalanced domain samples are split in 56 different subdomains where some classes have very few samples. For instance, in the Spanish dataset the category 504 (Welfare State Expansion) has 9.05% of the samples, while 502 (Culture) and 204 (Constitutionalism: Negative) 3.9% and 0.25%, respectively.

Due to the imbalance, the results have been presented using three different measures: the accuracy rate, the F-Measure and G-Mean. Moreover, in order to statistically evaluate the improvement given by the auxiliary information, the recommendations provided by Demšar[33] have been followed. The author proposes the use of non-parametric statistical tests to check whether there are differences among different algorithms or not. Specifically, Demšar concludes that Friedman test[34] with the corresponding post hoc tests is the most suitable approach when comparing more than two classifiers over different datasets.

In order to evaluate the proposed approach, the dataset has been divided into two different subsets: training and validation sets (85%), and test set (15%). The training and validation set has been used in order to create models with fivefold cross-validation to later test their performance with the same test set. The reason why the authors have split the dataset in two subsets and then apply cross-validation to one of them is because early stopping[35] has been used in order to stop model's training when it started to over-fit. Early stopping compares the training accuracy with the validation accuracy and after some epochs without any improvements in the validation accuracy it stops the training. However, the model may have over-fitted with respect to the validation set; therefore, a third set, the test set, is needed in order to measure the real performance of the model.

Furthermore, since the dataset is imbalanced, the authors have applied stratification in order to preserve the same percentage of samples for each class. Using this approach the authors are able to evaluate how each class is classified since it ensures that in each of the subsets there will be a representation of each class.

Tables 4 and 5 show the results of domain and subdomain datasets, respectively, for each of the used languages. Five different classifier configurations are tested with each of the datasets:

Table 3. Domain codes' distribution per dataset.

	1	2	3	4	5	6	7
Spanish	6.78%	6.88%	15.08%	28.59%	27.42%	5.88%	9.337%
Finnish	8.24%	4.51%	6.88%	22.35%	32.41%	13.68%	11.905%
Danish	7.01%	5.15%	6.47%	20.23%	37.5%	14.19%	9.42%
English	6.50%	4.42%	10.64%	25.45%	31.77%	11.20%	9.99%
German	9.65%	8.41%	8.94%	22.85%	28.75%	11.03%	10.34%
French	10.45%	6.99%	6.80%	22.23%	32.55%	11.38%	9.56%
Italian	7.32%	7.24%	17%	24.20%	26.69%	11.52%	5.99%

Table 4. Domain results for each one of the experiment configuration and datasets.

	Spanish	Finnish	Danish	English	German	French	Italian
E1	Acc: 65.79%	Acc: 47.03%	Acc: 52.2%	Acc: 64.29%	Acc: 58.01%	Acc: 57.54%	Acc: 53.04%
	F1: 61.11	F1: 41.61	F1: 44.49	F1: 60.04	F1: 55.78	F1: 52.71	F1: 48.62
	G-M: 75.2	G-M: 61	G-M: 63.42	G-M: 75.12	G-M: 71.63	G-M: 69.73	G-M: 66.66
E2	Acc: 66.09%	Acc: 47.91%	Acc: 54.47%	Acc: 64.63%	Acc: 58.41%	Acc: 58.39%	Acc: 54.17%
	F1:62.12	F1:43.38	F1:46.7	F1:60.17	F1:55.89	F1:53.77	F1:48.82
	G-M: 76.32	G-M: 62.53	G-M: 64.49	G-M: 75.09	G-M: 71.69	G-M: 70.47	G-M: 66.92
E3	Acc: 71.08%	Acc: **57.42%**	Acc: **58.01%**	Acc: 67.85%	Acc: 64.43%	Acc: 61.96%	Acc: 59.97%
	F1: 67.13	**F1: 52.58**	**F1: 50.27**	F1: 64.17	F1: 62.44	F1: 57.77	F1: 55.45
	G-M: 79.67	**G-M: 69.61**	**G-M: 67.83**	**G-M: 78.43**	G-M: 76.48	G-M: 73.2	G-M: 72.13
E4	Acc: 72.18%	Acc: 56.68%	Acc: 57.38%	Acc: 68.61%	**Acc: 65.7%**	Acc: 61.94%	**Acc: 60.74%**
	F1:68.42	F1:51.41	F1:49.73	F1:64.93	F1:64.03	F1:57.71	F1:57.07
	G-M: 80.41	G-M: 68.73	G-M: 67.64	G-M: 78.33	**G-M: 77.54**	G-M: 73.27	**G-M: 72.84**
E5	**Acc: 72.44%**	Acc: 57.04%	Acc: 57.99%	**Acc: 69.02%**	Acc: 65.48%	**Acc: 62.6%**	**Acc: 61.06%**
	F1:68.82	F1:51.84	F1:50.04	**F1:65.32**	**F1:63.43**	**F1:58.74**	F1:56.43
	G-M: 80.79	G-M: 69.02	G-M: 67.47	G-M: 78.41	G-M: 77.24	**G-M: 73.88**	G-M: 72.55

The accuracy (acc), f-measure (F1) and G-Mean (G-M) of each experiment is shown.
The best scores are highlighted in bold.

Table 5. Subdomain results for each one of the experiment configuration and datasets.

	Spanish	Finnish	Danish	English	German	French	Italian
E1	Acc: 54.16%	Acc: 30.96%	Acc: 37.28%	Acc: 50.65%	Acc: 42.71%	Acc: 46.12%	Acc: 37.46%
	F1: 38.33	F1: 18.1	F1: 22.33	F1: 35.32	F1: 26.66	F1: 29.85	F1: 25
	G-M: 59.24	G-M: 38.64	G-M: 42.07	G-M: 58.51	G-M: 51.02	G-M: 50.06	G-M: 42.89
E2	Acc: 55.96%	Acc: 33.07%	Acc: 38.54%	Acc: 52.18%	Acc: 43.45%	Acc: 47.27%	Acc: 43.21%
	F1: 41.69	F1: 20.65	F1: 24.86	F1: 38.89	F1: 28.25	F1: 31.94	F1: 31.3
	G-M: 61.64	G-M: 41.25	G-M: 43.52	G-M: 61.24	G-M: 52.15	G-M: 51.77	G-M: 47.28
E3	Acc: 60.58%	Acc: 35.56%	Acc: 42.15%	Acc: 55.35%	Acc: 48.77%	Acc: 50.26%	Acc: 45.83%
	F1: 44.31	F1: 19.99	F1: 24.41	F1: 38.74	F1: 32.27	F1: 29.33	F1: 30.17
	G-M: 63.04	G-M: 40.57	G-M: 42.97	G-M: 60.89	G-M: 55.82	G-M: 50.32	G-M: 46.92
E4	Acc: 61.2%	Acc: 36.49%	Acc: 40.56%	Acc: 55.7%	Acc: 50.69%	Acc: 52.02%	Acc: 45.36%
	F1: 45.04	F1: 22.67	F1: 24.6	F1: 40.73	F1: 34.21	F1: 33.8	F1: 30.32
	G-M: 64.27	G-M: 43.33	G-M: 43.3	G-M: 63.2	G-M: 57.87	G-M: 53.51	G-M: 47.21
E5	**Acc: 62.31%**	**Acc: 39.03%**	**Acc: 41.15%**	**Acc: 56.85%**	**Acc: 50.84%**	**Acc: 53.72%**	**Acc: 49.66%**
	F1: 47.7	**F1: 24.49**	**F1: 25.55**	**F1: 42.73**	**F1: 35.68**	**F1: 38.17**	**F1: 34.66**
	G-M: 66.14	**G-M: 44.86**	**G-M: 44.3**	**G-M: 64.56**	**G-M: 58.72**	**G-M: 57.04**	**G-M: 50.6**

The accuracy (acc), f-measure (F1) and G-Mean (G-M) of each experiment is shown.
The best scores are highlighted in bold.

- *E1*. Only the sentence to be classified with no additional context;
- *E2*. The sentence plus the political party which belongs to;
- *E3*. The sentence plus the previous sentence in an additional channel on the CNNs;
- *E4*. The sentence plus the previous sentence in another CNNs structure, concatenating the features extracted by both networks;
- *E5*. The sentence, the political party to which the sentence belongs to and the previous sentence in another CNN.

If we compare the accuracy, F1 and G-mean scores shown in Tables 4 and 5 without any statistical analysis, the following conclusions can be drawn:

- Adding the previous sentence (as a channel or in another CNNs structure) improves the performance of the classifier on domains and subdomains.
- Adding the political party to which the text belongs to significantly improves the performance of the classifier in some languages (more than 1 point in F1): Spanish, Finnish, Danish and French. This improvement is more remarkable when classifying the sentences on subdomains.
- The previous phrase and the political party are not always complementary features when classifying domains as it can be seen in Table 4. However, they are complementary when classifying subdomains (see Table 5).

Table 6. Differences between the average ranking of the tested algorithms computed with the Nemenyi test and F-measures of the classifiers.

	E1	E2	E3	E4	E5
E1	–	1.42	**1.85**	**2.71**	**3.64**
E2	1.42	–	0.42	1.28	**2.21**
E3	**1.85**	0.42	–	0.85	**1.78**
E4	**2.71**	1.28	0.85	–	0.92
E5	**3.64**	**2.21**	**1.78**	0.92	–

Table 7. Differences between the average ranking of the tested algorithms computed with the Nemenyi test and G-means of the classifiers.

	E1	E2	E3	E4	E5
E1	–	1.35	**2**	**2.78**	**3.5**
E2	1.35	–	0.64	1.42	**2.14**
E3	**2**	0.64	–	0.78	1.5
E4	**2.78**	1.42	0.78	–	0.71
E5	**3.5**	**2.14**	1.5	0.71	–

Another way to analyse the improvement resulting of each approach is to apply the Friedman test. However, when a statistical analysis of the results is made, the conclusions vary slightly. The Friedman test has been applied with two different metrics: F-measure and G-mean. On one hand, after applying the Friedman test with the F-measures, the resulting p-value is $1.537e-08$. On the other hand, when the Friedman test is applied to G-mean, a p-value of $3.208e-08$ is obtained.

Since the two p-values are smaller than 0.01, the null hypothesis (that all algorithms perform equally) can be rejected. Once the null hypothesis has been rejected, the corresponding post hoc tests can be performed in order to compare the different algorithms between them and analyse which are different.

In order to compare all the algorithms pairwise, Demar proposes the use of the Nemenyi test.[36] This post hoc test determines the critical difference (CD) for a significance level α. Next, if the difference between the average ranking of two algorithms is greater than the critical difference, then the null hypothesis that the algorithms perform equally is rejected.

The Nemenyi test has been performed with a significance of $\alpha = 0.05$ and two different metrics: F-measure and G-mean. In both cases, the resulting critical difference is 1.6768. Therefore, if any of the average ranking of two algorithms shown in Tables 6 and 7 is greater than the critical difference, then the null hypothesis is rejected and it can be affirmed that the two algorithms have a different behaviour.

After analysing the results shown in Tables 6 and 7, the following conclusions can be drawn:

- The comparisons where the null hypothesis has been rejected (in bold font) are quite similar in both tables. However, contrary to F-measure, using the G-Mean as metric does not reject the null hypothesis when comparing E3 and E5.
- In both cases, it is statistically validated that adding the previous phrase in an additional channel (E3) or another CNNs structure (E4) have a different behaviour than the baseline without any context data. Therefore, it can be affirmed that adding the previous phrase in an additional channel or as another CNNs structure improves the performance of the classifier, as it can be seen in Tables 4 and 5.
- Regarding the use of the political party which says the phrase, it has not been possible to statistically validate that there is an improvement when it is used. However, there is an improvement in performance in the majority of performed experiments, particularly when classifying subdomains and it is combined with the previous phrase as it can be seen in Table 5. One of the possible reasons for this could be that it has not been possible to perform the statistical tests with a greater number of datasets.

Conclusion and future work

In this research, the authors have introduced and validated a novel approach for automated analysis of the political discourse in texts, which helps to better understand its underlying themes. The proposed approach is based in multi-scale CNNs enhanced with context information (the political leaning of the speaker and the raw text of previous phrase, which to the best of our knowledge has not been used before). The proposed approach has been validated using two different types of codifications for the political discourse: domains (7 categories) and subdomains (56 categories).

The proposed system has been validated using the Friedman test with the corresponding post hoc tests (Nemenyi test). It has been statistically certified that adding the previous phrase as an additional channel or in another CNNs structure improves the performance of the classifier. However, the authors have not been able to statistically validate that the political party which the phrase belongs to improves the performance of the classifier, even though there is an improvement in performance in the majority of performed experiments, particularly when classifying subdomains and it is combined with the previous phrase as it can be seen in Table 5.

As future work, the authors plan to improve the current system by studying how training it with datasets in different languages can help to enhance its generalization capabilities using transfer learning techniques. Author's intuition is that the abstract representation of the sentences obtained after the convolutional layers will have similar structures for the same categories across different languages. It is also planned to analyse different attention mechanisms to identify the words that are more relevant in each sentence to improve the whole process.

Acknowledgements

We gratefully acknowledge the support of NVIDIA Corporation for the donation of Titan X used in this research.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: We gratefully acknowledge the support of the Basque Government's Department of Education for the predoctoral funding, the Ministry of Economy, Industry and Competitiveness of Spain under Grant No. CSO2015-64495-R (Electronic Regional Manifestos Project).

References

1. Zirn C, Glavaš G, Nanni F, et al. Classifying topics and detecting topic shifts in political manifestos, 2016, https://ub-madoc.bib.uni-mannheim.de/41552/1/classyman.pdf
2. Nanni F, Zirn C, Glavaš G, et al. TopFish: topic-based analysis of political position in us electoral campaigns. In: *Proceedings of the international conference on the advances in computational analysis of political text (PolText)*, Dubrovnik, 14–16 July 2016. Zagreb: University of Zagreb.
3. Glavaš G, Nanni F and Ponzetto SP. Cross-lingual classification of topics in political texts. In: *Proceedings of the second workshop on NLP and computational social science*, Vancouver, BC, Canada, 3 August 2017, pp.42–46. Stroudsburg, PA: ACL.
4. Iyyer M, Enns P, Boyd-Graber J, et al. Political ideology detection using recursive neural networks. In: *Proceedings of the Association for Computational Linguistics*, Baltimore, MD, 22–27 June 2014, pp.1113–1122. Stroudsburg, PA: ACL.
5. Rao A and Spasojevic N. Actionable and political text classification using word embeddings and LSTM. arXiv preprint arXiv:1607.02501, 2016.
6. Hopkins DJ and King G. A method of automated nonparametric content analysis for social science. *Am J Polit Sci* 2010; 54(1): 229–247.
7. Velikovich L, Blair-Goldensohn S, Hannan K, et al. The viability of web-derived polarity lexicons. In: *Human language technologies: the annual conference of the North American chapter of the Association for Computational Linguistics*, Los Angeles, CA, 2–4 June 2010, pp.777–785. Stroudsburg, PA: ACL.
8. Kalchbrenner N, Grefenstette E and Blunsom P. A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188, 2014.
9. Kim Y. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882, 2014.
10. Johnson R and Zhang T. Semi-supervised convolutional neural networks for text categorization via region embedding. In: *Proceedings of the 28th international conference on advances in neural information processing systems*, Montreal, QC, Canada, 7–12 December 2015, pp.919–927. Cambridge, MA: The MIT Press.
11. Zhang Y, Roller S and Wallace B. MGNC-CNN: a simple approach to exploiting multiple word embeddings for sentence classification. arXiv preprint arXiv:1603.00968, 2016.
12. Zhang Y and Wallace B. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820, 2015.
13. Wüst AM and Volkens A. Euromanifesto coding instructions, 2003, http://www.mzes.uni-mannheim.de/publications/wp/wp-64.pdf
14. Alonso s, Cabeza L and Gómez B. Disentangling peripheral parties' issue packages in subnational elections. *Comp Eur Polit* 2017; 15(2): 240–263.
15. Benoit K. Irish political parties and policy stances on European integration. *Irish Polit Stud* 2009; 24: 447–466.
16. Budge I. *The standard right-left scale*. London: University of Essex, 2013.
17. Lowe W, Benoit K, Mikhaylov S, et al. Scaling policy preferences from coded political texts. *Legis Stud Quart* 2011; 36: 123–155.
18. Budge I, Klingemann HD, Volkens A, et al. *Mapping policy preferences: estimates for parties, electors, and governments 1945–1998*, vol. 1. Oxford: Oxford University Press, 2001.
19. Volkens A. Manifesto coding instructions, 2002, https://www.poltext.org/sites/poltext.org/files/iii02-201.pdf
20. Gómez B, Alonso S and Cabeza L. Regional manifestos: coding manual, 2011, http://www.regionalmanifestosproject.com/ingles/download-dataset
21. Mikhaylov S, Laver M and Benoit KR. Coder reliability and misclassification in the human coding of party manifestos. *Polit Anal* 2012; 20(1): 78–91.
22. Poria S, Cambria E and Gelbukh A. Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. In: *Proceedings of the conference on empirical methods in natural language processing*, Lisbon, 17–21 September 2015, pp.2539–2544. Stroudsburg, PA: ACL.
23. Poria S, Cambria E and Gelbukh A. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowl-Based Syst* 2016; 108: 42–49.
24. Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

25. Park K. Pre-trained word vectors of 30 + languages, 2018, https://github.com/Kyubyong/wordvectors
26. Almeida A and Bilbao A. Spanish 3B words Word2Vec embeddings, 2018, https://doi.org/10.5281/zenodo.1155 474
27. LeCun Y, Bottou L, Bengio Y, et al. LeNet-5, convolutional neural networks, 2015, http://yann.lecun.com/exdb/lenet
28. Ioffe S and Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015.
29. Boureau YL, Ponce J and LeCun Y. A theoretical analysis of feature pooling in visual recognition. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, Haifa, 21–24 June 2010, pp.111–118. Madison, WI: Omnipress.
30. Srivastava N, Hinton GE, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res* 2014; 151: 1929–1958.
31. Kingma D and Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
32. Lehmann P, Matthieß T, Merz N, et al. *Manifesto corpus* (version: 2017-2). Berlin: WZB Berlin Social Science Center, 2018.
33. Demšar J. Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 2006; 7: 1–30.
34. Friedman M. A comparison of alternative tests of significance for the problem of m rankings. *Ann Math Statist* 1940; 11: 86–92.
35. Prechelt L. Early stopping-but when? In: G Montavon, G Orr and KR Müller (eds) *Neural networks: tricks of the trade*. Berlin: Springer, 1998, pp.55–69.
36. Nemenyi P. Distribution-free multiple comparisons. *Biometrics* 1962; 18: 263.