Information Processing and Management 50 (2014) 554–567

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Automatic thematic classification of election manifestos

Suzan Verberne [a,*], Eva D'hondt [b], Antal van den Bosch [b], Maarten Marx [c]

[a] Centre for Language Studies and Institute for Computing and Information Sciences, Radboud University, Nijmegen, Netherlands
[b] Centre for Language Studies, Radboud University, Nijmegen, Netherlands
[c] Informatics Institute, University of Amsterdam, Netherlands

ARTICLE INFO

Article history:
Received 27 March 2013
Received in revised form 30 August 2013
Accepted 26 February 2014
Available online 13 April 2014

Keywords:
Text classification
Political data
Expert evaluation

ABSTRACT

We digitized three years of Dutch election manifestos annotated by the Dutch political scientist Isaac Lipschits. We used these data to train a classifier that can automatically label new, unseen election manifestos with themes. Having the manifestos in a uniform XML format with all paragraphs annotated with their themes has advantages for both electronic publishing of the data and diachronic comparative data analysis. The data that we created will be disclosed to the public through a search interface. This means that it will be possible to query the data and filter them on themes and parties. We optimized the Lipschits classifier on the task of classifying election manifestos using models trained on earlier years. We built a classifier that is suited for classifying election manifestos from 2002 onwards using the data from the 1980s and 1990s. We evaluated the results by having a domain expert manually assess a sample of the classified data. We found that our automatic classifier obtains the same precision as a human classifier on unseen data. Its recall could be improved by extending the set of themes with newly emerged themes. Thus when using old political texts to classify new texts, work is needed to link and expand the set of themes to newer topics.

1. Introduction

Isaac Lipschits (1930–2008) was a Dutch historian and political scientist. One of his works is an annotated collection of election manifestos (party programmes) for the Dutch elections between 1977–1998 (Lipschits, 1977). For each election year he compiled a book with the manifestos published by all parties that participated in that year's elections. This book was then made available for purchase nationally before the election. To facilitate voters' decision-making process when comparing parties on election issues, Lipschits manually labelled the manifestos with themes: he segmented the manifestos into coherent text fragments, gave them a unique identifier consisting of the party's acronym and a number, and added an index of themes in the back of the book referring to these identifiers.

In the Political Mashup project (Marx, 2009), Dutch political data from 1814 onwards is being digitized and indexed. The aim of the project is to bring together political information produced by political parties and information on the reception of political promises and actions, in the news as well as in user–generated content. The data are not only digitized and integrated but also disclosed to the public. This means that it will be possible to query the data and filter them on themes, persons and events. In traditional historical and political research, scientists who work with textual data have to open each

* Corresponding author. Tel.: +31 24 36 53431.
E-mail address: s.verberne@cs.ru.nl (S. Verberne).

http://dx.doi.org/10.1016/j.ipm.2014.02.006
0306-4573/

potentially relevant document or book separately, and search and browse through them using a static register. This requires a vast amount of manual analysis and time. The goal of Political Mashup is to automate part of this manual effort.

Election manifestos are traditionally considered to be a key source of information on the ideological stance of a political party on election issues (Budge, Klingemann, Volkens, Bara, & Tanenbaum, 2001). The annotated election manifestos by Isaac Lipschits are part of the Political Mashup data. The aims of the work presented in the current paper are: (1) to digitize the 1977–1998 Lipschits collections and (2) to build an automatic classifier for more recent, unclassified election manifestos. The starting points for our work are the Lipschits books, scanned as PDF files.

The challenges we face in digitizing the data and building the classifier are:

- the PDFs contain complicating OCR errors;
- the text fragments have been assigned multiple themes by Lipschits (more than 6 on average), which resulted in a large number of themes (more than 200) in total, for a relatively small corpus;
- there are inconsistencies in the labelling schemes over time (e.g. 'fishing industry' vs. 'fishing industry policy')[1];
- new themes have emerged over time (e.g. 'information technology') and the same themes may have changing content over time (concept drift);
- automatic text segmentation leads to shorter and much more segments than in the manually segmented training data.

We took the following approach: We first converted the scanned PDFs to XML data in which each text fragment has been annotated with the Lipschits themes (Section 3). We then used these data to train an automatic classifier, using the original segmentation and labelling by Lipschits (Section 4). We optimized the classifier on the task of classifying election manifestos using models trained on earlier years (Section 5). Based on these experiments, we built a classifier that is suited for classifying election manifestos from 2002 onwards using the data from the 1980s and 1990s. We then evaluated the results by having a domain expert manually assess a sample of the classified data (Section 6).

We found that our automatic classifier obtains the same precision as a human classifier on unseen data. Its recall could be improved by extending the set of themes with newly emerged themes. In addition, we found that although a smaller theme set could improve classification scores even more, a more fine-grained classification is preferred by domain experts.

2. Background and related work

Automatic text classification (or document categorization) in predefined categories on the basis of pre-categorized examples is a supervised machine learning task (Sebastiani, 2002). If a text in the collection has been assigned to more than one category, the task is called *multi-label* text classification. In multi-label text classification it is sensible for the classifier to rank the classes according to their estimated relevance to the text. Like any supervised machine learning task, text categorization requires a collection of (manually) labelled examples as training material. When training a classification model, each text in the collection is represented as a feature vector. The features are the terms in the collection, where *terms* can be usually read as *words*, although many attempts have been made to extend words with *n*-grams or phrases (Sebastiani, 2002). Considering each word in the corpus as an independent feature makes text categorization a classification problem in a high-dimensional feature space. The most widely used algorithms for text classification are Support Vector Machines (SVMs) and Naïve Bayes (NB), although NB is mostly considered as a baseline for more sophisticated techniques. Van Mun (1999) argues that in domains with large amounts of features it is better to use Balanced Winnow (Dagan, Karov, & Roth, 1997; Littlestone, 1988) because of its ability to discard irrelevant features. In Section 4, we describe the Winnow classifier that we used for training the Lipschits classifier in more detail.

2.1. Political text classification

Automatic classification of political texts using supervised learning techniques has been applied to legislative texts, parliamentary documents, manifestos, and even speeches of the Dutch Queen (Breeman et al., 2009; Hillard, Purpura, & Wilkerson, 2008; Louwerse, 2011; Purpura & Hillard, 2006). Developers of classifiers for topical classification of political texts are faced with two challenges: (1) the definition of political topics is often fine-grained, resulting in a large set of topics and (2) the set of topics and the content of these topics is not stable over time (Mourão et al., 2008). The research council of the European Union has trained a multi-label classifier on EU legislation labelled with the official EU EUROVOC thesaurus consisting of more than 7000 classes (Pouliquen, Steinberger, & Ignat, 2003). The classifier proposes a ranked list of classes for each input document. Their JEX system reaches an *R*-precision of 0.56 for English documents. On average, documents are labelled with six classes, so this means that roughly three from the top 6 proposed classes are correct. Both the classifier and the training data have recently been made available (Steinberger, Ebrahim, & Turchi, 2012). Hillard, Purpura, and Wilkerson (2007) evaluate the efficacy and accuracy of automatic text classification using a corpus of all federal public bills introduced in the U.S. since 1947. The bills have been labelled according to a hierarchical classification scheme comprising 20 major

[1] For the convenience of the reader, we translated all example themes from Dutch to English throughout the paper, except for places where the literal orthography is important.

policy topics and 226 subtopics. The authors are able to build a classifier that achieves levels of accuracy comparable to humans (measured as inter-rater agreement: Cohen's $\kappa = 0.8$ at the subtopic level), while reducing human labor effort by about 80%. An important difference with our work is that the topics in their classification scheme are mutually exclusive; thus, they train a classifier that assigns one topic per document, while the Lipschits data require a multi-label classification approach.

A related task to political theme classification is political opinion classification. In this task the goal is to correctly sort political texts depending on whether they support or oppose a given political issue under discussion (Awadallah, Ramanath, & Weikum, 2010; Yu, Kaufmann, & Diermeier, 2008). Political opinion classification is related to sentiment classification but political texts tend to contain few sentiment-bearing words. Hirst, Riabinin, and Graham (2010) aim to classify Canadian legislative speech by party. They initially succeed in this task but an analysis of the discriminative terms show that their classifier not distinguishes between the ideologies of the parties but the roles of the party (government or opposition). More recently, Diermeier, Godbout, Yu, and Kaufmann (2012) succeeded in predicting a senator's ideological position with 92% accurracy based on terms associated with conservative and liberal ideologies. They conclude their paper with the wish to be able to investigate the content shift of ideologies over time. Yu et al. (2008) use U.S. senate speeches as documents and the speakers' party affiliation as classification labels. They formulate a number of recommendations for text classification in the political domain: First, due to the subjectivity of the class definitions, human classifiers may not agree on the correct labelling. Second, manual labelling is sensitive to errors. Third, the data may not be stable over time: in political agendas, the relevant issues change over time and new themes can emerge in new data. Fourth, the data is not independent: in the case of political speeches, speakers respond to each other, adopting themes and vocabulary from each other. The fifth problem is the sample size bias in the training data, especially in the case of small data sets. In the development of the Lipschits classifier, we face the first three challenges.

Closely related to thematic text classification is topic identification in political texts. Quinn, Monroe, Colaresi, Crespin, and Radev (2010) develop a statistical learning model that uses word choices to infer topical categories covered in a set of speeches and to identify the topic of specific speeches. Their method is unsupervised: they do not use a list of predefined topics, and they do not need a set of human-labelled examples. They use the topic model to examine the agenda in the U.S. Senate from 1997 to 2004. From the topics that are identified by their algorithm, they manually infer category labels and they create a hierarchical topic scheme.

2.2. Dutch political data

Some work has focused on Dutch political data: Jijkoun, Marx, De Rijke, and van Waveren (2007) describe an electoral search engine aimed at helping the general public to make an informed decision whom to vote for in the 2006 elections. They indexed news articles, blogs and election manifestos, and built a search engine in which the users can search by theme or free text. In this work, 179 themes were predefined by the *Instituut voor Publiek en Politiek* (IPP).[2] Of these, 175 were used as thematic queries by users of the website. In total, 28 thousand thematic searches were performed compared to 117 thousand free text searches. Although the search results are not evaluated in the paper, these statistics show the potential value of thematic labelling of political texts for application in a search system.

Gielissen and Marx (2009) describe the development of PoliDocs.nl, a Web Information System for the disclosure of Dutch parliamentary publications. They transformed long PDF documents containing the meeting notes of one day into a uniform XML format and make them available in a new Web Information System. The authors present a list of 15 requirements for the Information System that were formulated by end-users through a collective Dutch weblog. Most requirements are technological in nature. Only one is related to content: "Users should be able to display the results of a keyword search on a timeline where parliamentary papers that belong together are grouped". This requirement indicates that for end users of a search system for political data, time aspects are important, and grouping of documents (by party or theme) is necessary. Gielissen and Marx (2009) developed a faceted search engine for PoliDocs. In faceted search, the user can filter for categories in addition to search with keywords to find documents. In PoliDocs, it is possible to filter documents on the basis of personal names, year and political party.

Kaptein and Marx (2010) use transcripts of meetings of the Dutch parliament for experiments with *focused retrieval* and *result aggregation*. They evaluate the above-described PoliDocs.nl interface. They find that users performed thematic search tasks up to eight times faster using focused retrieval and faceted search compared to standard full document retrieval. They stress the importance of time- and party-specific information when searching for themes in political data: for users, "it is interesting to see how that theme developed over time and which political parties claimed that theme" (p. 425).

2.3. Temporal change in text classification

As we have mentioned above, an important aspect of political data is temporal change. Forman (2006) discusses the problem of topic drift in classifying news data. The authors learn a new classifier for each day, and they augment the bag-of-words feature vector with additional binary features that are generated by the predictions previous daily classifiers would

[2] Current name: Prodemos http://www.prodemos.nl/.

have made for today's cases. Mourão et al. (2008) show evidence that time should be taken into consideration in classification techniques and algorithms. According to Rocha, Mourão, Pereira, Gonçalves, and Meira (2008), the performance of classifiers is affected by three different temporal effects: class distribution, term distribution and class similarity. They try to find the selection of training data that optimizes the trade-off between the sampling effect (more training data gives better classification results) and the time effects (training data from mismatching time moments give worse classification results). They evaluate their approach on scientific articles. The trade-off between training set size and temporal match is also relevant for our data, but the difference with scientific literature is that the election manifestos originate from a few moments in history (election years: 1977, 1981, 1986, etc.), while scientific literature and news can be dated and timed at finer granularity levels. In addition, there is much more scientific literature available to train a classifier than there are annotated political texts.

3. Converting the PDF books to enriched publications

Each Lipschits book contains all election manifestos of the policitical parties that officially participated in that year's election. The body of data are the manifesto texts, which have been manually segmented by Lipschits. Each segment has been labelled with a number. The numbering starts at 1 for each party. We call these number labels 'topic numbers'. At the back of the book is a register of alphabetically sorted themes. For each theme, Lipschits has listed the associated parts of the election manifestos per party (e.g. CD, CDA, D66, etc.) by topic number. See Fig. 1a for an example of the register.

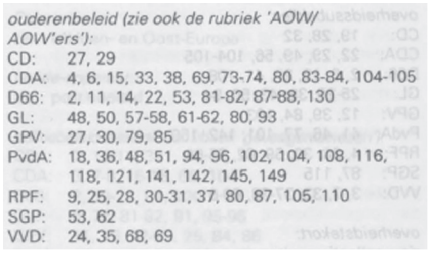
The PDFs were converted to XML using `pdftohtml` with the `-xml` option. Both the body of the election manifestos and the register needed a substantial amount of clean-up before the data could be published digitally, or used for classification. We used a Perl script for textual clean-up; the process is described in the next two subsections.

3.1. Body of data: numbered paragraphs

In the election manifesto texts, the start of each new topic is labelled with a topic number. In the source PDF, topic numbers are preceded by a bullet, so the task of passage segmentation can be defined as: finding a bullet followed by a number, and then saving all text between this number and the following bullet as content belonging to the topic number. However, we had to take into account a number of problems. First, during the scanning process the edge of the right page sometimes became part of the scan of the left page, so for a number of pages additional noisy text fragments (sometimes containing a bullet) were part of the XML data. Another scanning problem was that a few pages were mistakenly not scanned, as a result of which some topic numbers were missing.

In addition, we came across numerous OCR problems. We included a list of frequently recurring OCR corrections (such as 'WO' for the party name 'VVD') in our clean-up script. Some bullets were not recognized, or a bullet was recognized where there was none. Moreover, numbers behind bullets were not always recognized (or recognized as letters e.g. 'l' for '1'). This was further complicated by inconsistencies in text formatting between party programmes: sometimes bullets were used for lists within a topic (no new topic starts at the bullet); in other cases numbers were used for lists within a topic (no new topic starts at the number). We solved these problems by writing a set of regular expression patterns that try to find and reconstruct all topic numbers for each party, using the presence of bullets and expectations on the following number based on the preceding number.

In order to assess the quality of the passage segmentation, we counted (1) the number of automatically identified passages and (2) the number of automatically identified text passages that start with a number (indicating correct segmentation). The results are in Table 1.



(a) Original pdf

```
ouderenbeleid (zie ook de rubriek 'AOW/
AOW'ers'):
CD: 27,29
CDA: 4,6,15,33,38,69, 73-74,80, 83-84,104-105
D66: 2, 11, 14, 22, 53, 81-82, 87-88, 130
GL: 48, 50, 57-58, 61-62,80,93
GPV: 27, 30, 79, 85
PvdA: 18, 36, 48,51 , 94,96,102,104,108, 116,
118,121,141 , 142,145, 149
RPF:
SGP:
WO:
9, 25, 28, 30-31,37, 80, 87, 105, 110
53, 62
24,35, 68,69
```

(b) OCR output

Fig. 1. Example of incorrect column splits between party names and number sequences in the conversion from pdf to text through OCR.

Table 1
Counts of the numbers of passages in the digitized manifestos as an indication for the conversion quality.

year	Total # of passages	# Of identified passages	% Starting with number
1986	803	797	96
1994	951	945	97
1998	839	833	97

3.2. Register: per theme the listed paragraph numbers

When processing the theme register in the back of the book, we dealt with two challenges. First, we had to decide which subsequent lines should be concatenated and which not. For example, in Fig. 1, the second line should be concatenated to the first line and the line starting with "118, 121" should be concatenated to the line before it. An additional problem was that the OCR had sometimes incorrectly recognized column splits between party names and number sequences. For example, in Fig. 1b, the three last lines should be concatenated to the three lines preceding them. This even gets more complicating when trailing number lines (such as the line starting with "118, 121") are intertwined with incorrect column splits.

Manual cleanup was not an option because the total number of index lines per year was around 5000. We solved the decision problem of concatenating subsequent lines using a set of heuristics that takes into account punctuation, a set of likely topic names (extracted from the register itself), the set of party names, and a regular expression that matched sequences of numbers with possible OCR errors.[3] We solved incorrect column splits using two parallel stacks of party names and number sequences, pairing them up while keeping track of incomplete pairs. We checked the soundness of the converted theme registers by assuring that there are no remaining party names that are not followed by a number sequence and no number sequences that are not preceded by a party name.

3.3. Results of digitizing the election manifestos

Using the procedure described above, we were able to convert all texts from the election manifestos of 1998, 1994 and 1986 to annotated XML files of acceptable quality. The result are three XML files in which each text fragment has been annotated with the themes from the register. For the 1981 manifestos, the quality of the scans was too poor. The 1977 and 1989 manifestos use another encoding of topic numbers to text fragments, leading to new and more severe OCR problems. Therefore, we did not convert these years to annotated XML files yet.

Having the manifestos in a uniform XML format with all paragraphs annotated with their themes has advantages for both electronic publishing of the data and diachronic comparative data analysis. We briefly discuss these two aspects here. Note that both training data and automatically classified manifestos are stored in the same XML format. With a simple XQuery we can create the input data (in e.g. SPSS or CSV format) for performing diachronic comparative saliency analysis of the parties Laver (2001): i.e., how much attention gives each party to each theme (election topic) and how does that change over the years?

Using XSLT and CSS stylesheets we can publish the manifestos as hyper documents with a number of additional features not present in printed or scanned format. First, next to creating an inverted (back of the book) theme-index with hyperlinks to the paragraphs, we can create such an inverted theme-index for each manisfesto, as an alternative to a table of contents for that manifesto. This index also provides a good impression on the saliency of the themes of that party in that election year. An attractive presentation of this hyperlinked index can be given as a dispersion matrix (Bird, Klein, & Loper, 2009), with the themes ordered by their dispersion value. An example is presented in Fig. 2 for the manifesto of the 'pensioners party' in 1998. We see that dispersion is a good indicator of saliency of a theme: half of the themes in the top 20 directly relate to themes relevant for elderly people. Second, we can 'tag' each paragraph with its themes, making it quickly clear what the paragraph is about. The tags are hyperlinked to the inverted index of all manifestos of all parties, or just to the inverted index of the specific party. Third, we may allow users to create 'theme-views' of a manifesto: only show the paragraphs labelled by that theme. This makes it easy to compare the standpoints of different parties on a specific theme, or diachronically compare one party on one theme.

We have made the following data available for further research: the transformed manifestos in XML format for the years 1986, 1994 and 1998; the automatically labelled manifestos in XML for the years 2006, 2010 and 2012; the RelaxNG schema which validates these XML files; the XSLT stylesheet which transforms the XML into hyperlinked HTML documents as described above; the HTML output for all XML files; the XQuery creating a saliency matrix in CSV, and the output for the year 1998. The data is available at http://data.politicalmashup.nl/lipschits.

4. Automatic classification

We aim at developing a Lipschits classifier that can assign themes to unseen Dutch election manifestos that were written after Lipschits' work, i.e. from 2002 onwards. Since we do not have manually labelled data from these recent years, we have

[3] `[ISBlOaO-9,. ~-]*[O-9][ISBlOaO-9,. ~-]*.`

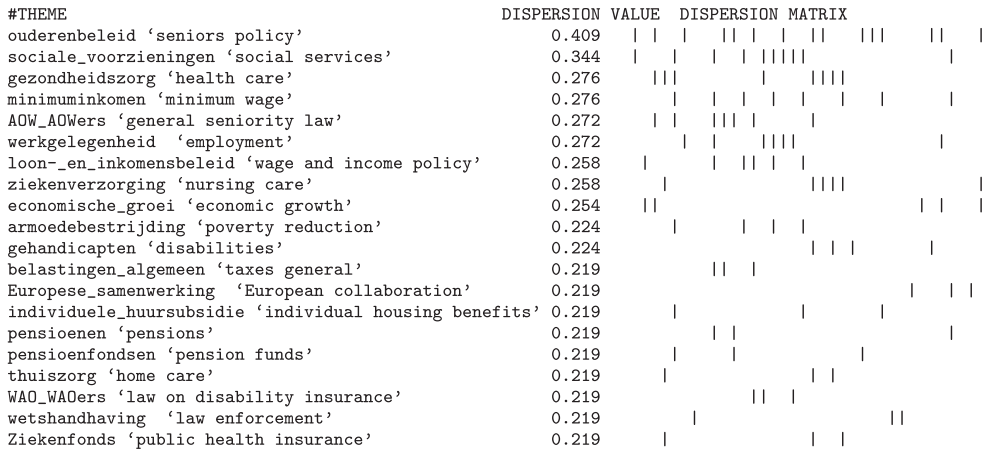


Fig. 2. Dispersion matrix of the 1998 manifesto of the 'pensioners party' (Algemeen Ouderen Verbond).

Table 2
Statistics on the classification data.

Year	# Words	Vocabulary size	# Texts	# Themes	# Words/text (std. dev.)	# Themes/text (min–max, std. dev.)
1986	290,942	32,836	797	214	373 (349)	10.5 (1–68, 8.3)
1994	269,270	32,499	951	210	284 (186)	6.9 (1–39, 4.3)
1998	244,697	31,162	826	218	302 (189)	8.3 (1–29, 4.7)

to rely on the older data from the eighties and nineties for training and optimization of the classifier. Therefore, our initial experiments are directed at obtaining the best classification results for the 1998 manifestos, using the data from older years in the training phase. In Section 4.1, we summarize the statistics of our classification data. In Section 4.2, we specify the two classifiers that we use, Support Vector Machines and Balanced Winnow. In Section 4.3, we optimize their parameters, and in Section 4.5, we compare their performance on the classification task.

4.1. Classification data

As described in the previous section, we transformed the digitized Lipschits books to a collection of short texts with associated themes. See Table 2 for the statistics per year. The table shows that although the texts are only a few hundred words long (317 on average, with a standard deviation of 252), Lipschits assigned more than six themes per text on average.

The large number of themes assigned per text segment seems to suggest that Lipschits' segmentation led to segments that are not coherent and may be too long. We looked into the possibility of splitting the texts in smaller segments, and classifying them separately, but we found that even within one sentence often more than one Lipschits theme occurs (see for example the text in Fig. 3). Furthermore, Lipschits intended his labeling to apply to entire paragraphs. The large number of themes assigned by Lipschits is not so much a result of long text segments (they are only 317 words on average) but of his very detailed classification system, and the political domain, where themes tend to be interrelated.

4.2. Classification method

The classification task we consider is a multi-label classification task, with each Lipschits theme being a class. We used the Linguistic Classification System LCS (Koster, Seutter, & Beney, 2003) for our experiments. The LCS is a generic system for the classification and routing of full-text documents.[4] The LCS trains a model (class profile) for each class in a one-versus-all setting (Van Mun, 1999; Koster & Seutter, 2003; Koster & Beney, 2007). In the test phase, the classifier runs a test document by each of the class profiles and assigns a score to each class, yielding a ranked list of classes for the document. The LCS performs text pre-processing in the form of tokenization and feature selection. For feature selection, we set the minimum document frequency to 2 and the minimum corpus frequency to 3. The LCS has one classifier built in: Balanced Winnow (Littlestone, 1988; Dagan et al., 1997), and it has an interface to the linear SVM^{light} implementation by Joachims and Svmlight (1999).

The Balanced Winnow algorithm learns two weights for each feature t and class c, $W^-_{t,c}$ and $W^+_{t,c}$. The difference between W^+ and W^- is the effective weight of the feature. In the training phase, Balanced Winnow goes through the set of training examples in multiple iterations. The score of a document d for a class c is computed as

[4] See http://www.phasar.cs.ru.nl/LCS/.

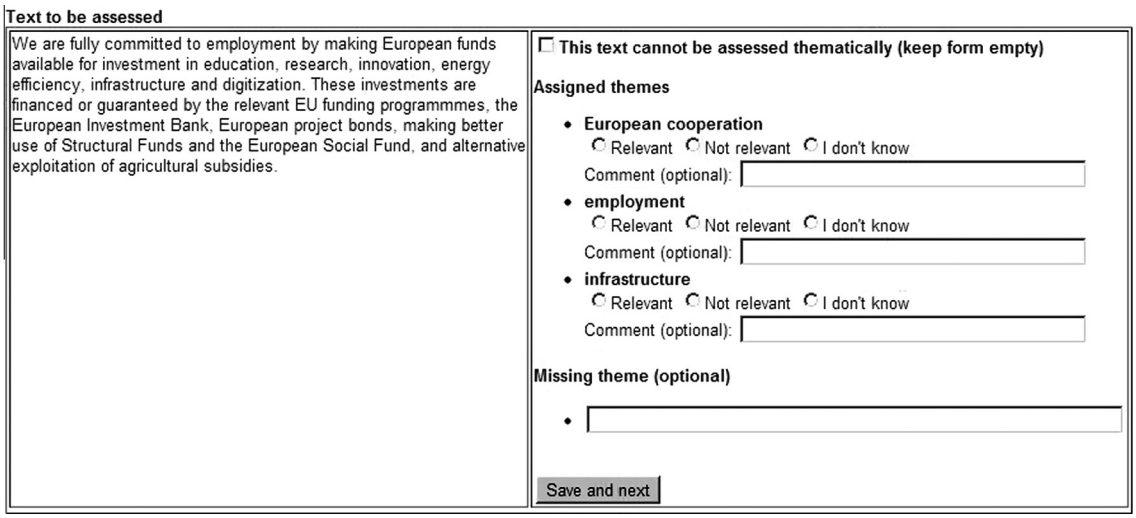


Fig. 3. The assessment interface for the evaluation of the classifier output by an expert (translated to English for the reader's convenience). The text fragment is shown on the left side. On the right side are from top to bottom: a checkbox for "this text cannot be assessed thematically"; the themes assigned by the classifier (in this case without theme merging) with radiobuttons for 'relevant', 'not relevant' and 'I don't know' and the option for leaving a comment; and a textbox for missing themes. When typing a string here, matching themes from the set of (merged) themes are shown.

$$\text{SCORE}(c, d) = \sum_{t \in d} \left(W^{+}_{t,c} - W^{-}_{t,c} \right) \cdot s(t, d) \tag{1}$$

where $s(t, d)$ is the strength of the term t in d (e.g. its tf-idf weight).

In the original Winnow implementation (Littlestone, 1988), a document d is considered to be part of the class c if $\text{SCORE}(c, d) > \theta$, with $\theta = 1$. In Dagan et al. (1997)'s version of Balanced Winnow, the threshold is defined by two parameters: an upper bound θ^{+} and a lower bound θ^{-}. During training, the algorithm predicts 0 when $\text{SCORE}(c, d) < \theta-$ and it predicts 1 when $\text{SCORE}(c, d) > \theta^{+}$. All positive examples with a score below θ^{-}, all negative examples with score above θ^{+} and all examples with scores in the range $[\theta^{-}, \theta^{+}]$ are considered as misclassified. When a training document is misclassified, the feature weights are updated using two parameters: the promotion parameter α ($\alpha > 1$) and the demotion parameter β ($0 < \beta < 1$). When a training document belonging to class c scores below θ for c, the positive weight $W^{+}_{t,c}$ of the active feature is multiplied by α (it is promoted) and the negative weight $W^{-}_{t,c}$ is multiplied by β (it is demoted). When a training document does *not* belong to class c but scores above θ, the positive weight of the active feature is demoted and the negative weight is promoted.

In SVM, a hyperplane is constructed that separates the positive from the negative training examples in the feature space. The best hyperplane is the one that has the largest distance to the nearest training examples in both classes. The larger the margin of the hyperplane (the wider the gap between the two classes), the lower the generalization error of the classifier. In realistic data, there is no hyperplane that can completely separate the positive from the negative training examples. Cortes and Vapnik (1995) introduced the soft margin approach, in which a hyperplane is constructed that separates the training set with a minimal number of errors. This approach requires a training parameter c that controls the trade-off between the width of the margin and the number of classification errors made during training.

In addition to the Winnow and SVM parameters, the LCS has three parameters that govern the selection of classes in the case of multi-label classification. For each text in the test set, the system returns a ranked list of all classes with the classification scores assigned to them. In order to decide which of the (top-ranked) classes should be assigned to the text, the set of selected classes is determined per text using three parameters: the maximum number of returned classes (*maxranks*), the minimum number of returned classes (*minranks*), and a *threshold* on the classification score.

4.3. Optimizing the classification parameters

SVM's linear kernal has one parameter c, the tradeoff between training error and margin. For Winnow, we optimized the values for the threshold parameters θ^{+} and θ^{-} and its weight update parameters α and β. We also tuned the class selection parameters *minranks*, *maxranks* and *threshold* in the LCS.[5]

[5] We did not optimize the number of training iterations but kept it at 10 in all experiments.

Table 3
Results for the automatic classification of the 826 texts fragments from the 1998 election manifestos using all data from 1986 and 1994 data with Support Vector Machines (linear kernel) and Balanced Winnow, both with optimized values for the parameters.

Classifier	Precision (%)	Recall (%)	*F*1 (%)	MAP
SVM 1 (threshold=0.6, $maxranks = 14$)	49.3	44.7	46.9	0.778
SVM 2 (threshold=1, $maxranks = 14$)	71.9	25.4	37.5	0.823
Winnow 1 (threshold=1, $maxranks = 10$)	70.5	35.2	47.0	0.841
Winnow 2 (threshold=1, $maxranks = 14$)	68.4	37.4	48.4	0.836

For tuning the parameters, we randomly partitioned the 1998 data into 10 equally sized parts and took one of the parts as test set and the rest as training set. We chose F-score with $\beta = 1$ (F1 for short, the harmonic mean of precision and recall) as optimization criterion. For the SVM parameter c, we followed the suggestion by Hsu, Chang, and Lin (2003) to test exponentially growing values of c, from 2^{-13} to 2^{13}. We obtained the optimal F1-score for $c = 2$.

For the Winnow parameters α, β, θ^+ and θ^-, we used a range of parameter values comparable to Koster and Beney (2007)[6] and we performed a full grid search. The resulting F1 landscape exhibits a lot of variation, with optimal values in divergent areas. The best scores appear to lie in the area of high α and θ^+ values. We chose $\alpha = 1.08;\ \beta = 0.84;\ \theta^+ = 5;\ \theta^- = -0.25$ as optimal parameter combination.

With the optimal choice for the SVM and Winnow parameters we tuned the class selection parameters. We set minranks to 1 so that each text gets assigned at least one theme. In the case of Winnow, we used the natural threshold of 1 (like in Koster & Beney (2007)). For SVM, we searched the optimal threshold by varying its value from -2 to 2 in steps of 0.2. For maxranks, we searched over the values from 6 to 20 in steps of 2. For SVM, we observe large differences in precision scores when varying threshold and maxranks. Taking precision into account besides F1, the optimal parameter combination is a maxranks value of 14 and either a threshold of 0.6 (with a good F1 score but low precision) or a threshold of 1 (with lower F1 but much higher precision). For Winnow, the differences in F1 for the best scoring values of maxranks are small; maxranks values of 10, 12 and 14 all give reasonable results for precision and F1.

4.4. Evaluation measures

As evaluation measures we use precision, recall and F1.

$$\text{Precision} = \frac{\#\text{ of themes assigned to text by classifier and expert}}{\#\text{ of themes assigned to text by classifier}} \tag{2}$$

$$\text{Recal} = \frac{\#\text{ of themes assigned to text by classifier and expert}}{\#\text{ of themes assigned to text by expert}} \tag{3}$$

$$\text{F}_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4}$$

We will also report Mean Average Precision (MAP), which gives the quality of the generated ranking of the themes per text:

$$\text{Average Precision} = \frac{\sum_{k=1}^{n} (P(k) \times rel(k))}{n_c}, \tag{5}$$

where $P(k)$ is the precision at rank k, n is the total number of assigned themes, n_c is the number of correctly assigned themes[7] and $rel(k)$ is a function that equals 1 if the theme at rank k is a correctly assigned theme, and zero if it is a false hit. MAP is the mean of this score over all texts in the test set.

4.5. Comparing Winnow and SVM

We trained both SVM and Balanced Winnow with parameter settings as described in Section 4.3 on the data from 1986 and 1994 and evaluated the trained models on the data from 1998. The results are in Table 3. For SVM, we show the results for two threshold values because the threshold that gave an optimal F1 score led to a very low precision. For Winnow, we show the results for two values of maxranks that lead to a slightly different balance between precision and F1. A paired t-test on F-scores for individual texts ($n = 826$) obtained by Winnow 1 and SVM 2 showed that Winnow gives significantly better results than SVM ($P < 0.001$). The other SVM setting (SVM 1) gives an average F1-score that is very close to the F1-score obtained by Winnow (46.9% compared to 47.0%), but against at the expense of a much lower precision (49.3% compared

[6] $\alpha \in \{1.01, 1.02, 1.04, 1.08, 1.16, 1.32, 1.64\}; \beta \in \{0.99, 0.98, 0.96, 0.92, 0.84, 0.68, 0.32\}; \theta^+ \in \{1, 1.2, 1.5, 2, 2.5, 3, 5\}; \theta^- \in \{-1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75, 1\}$.

[7] Note that in the original definition of Average Precision (Zhu, 2004), n_c is the total number of relevant themes, but this assumes that the output is a full ranking of all classes. In that case, all relevant classes are present in the ranking. Since we do not evaluate the full ranking but a selection of classes (maximum 10 per document), we cannot evaluate the full ranking and thus define n_c as the number of correctly assigned themes.

to 70.5%). Considering the results in Table 3, we decided to continue our experiments with Balanced Winnow as classifier. Because the differences between maxranks = 10 and maxranks = 14 are small, we opt for the somewhat higher precision and MAP and set maxranks to 10.[8]

5. Classification experiments

In all classification experiments, we used the texts from the 1998 manifestos as test data, and the 1986 and 1994 manifestos as training data. In all experiments, we used the optimized Winnow parameters that were found by tuning on a (held-out) subset of the 1998 data (Section 4.3).

5.1. Selection of training data

We first evaluated the use of the years 1986 and 1994 as training data, both separately and in combination. As a comparison, we performed 10-fold cross validation experiments on the 1998 data.[9] To give an indication of the complexity of the task, we also calculated the results for a weighted random guessing baseline (D'hondt, Verberne, Koster, & Boves, 2013). This was generated using a script that takes into account the theme distributions and theme frequency distributions in the training data. First, it processes the training set and fills two arrays:

- $X = (c_1, c_2, \ldots c_N)$, where N is the number of training examples and c_i is the number of themes for training example i.
- $Y = (t_1, \ldots t_1, t_2 \ldots t_2, \ldots t_M \ldots t_M)$, where M is the total number of theme occurrences in the training set and in which each theme t_j is listed as often as it occurs in the training set.

For each example e_k in the test set, the script randomly takes a number x from X, and it randomly takes x times a theme t_j from Y, while ensuring that there is no t_j occurs more than once for e_k. Then it creates the random labelling of the test set by assigning the labels t_j to the test example.

The results are in Table 4. Because not all themes from the 1998 texts are present in the 1986 or 1994 data, a classifier trained on the earlier years and evaluated on 1998 data can never obtain 100% recall. We calculated the maximum recall for each training set as follows. Let T_{test} be the set of themes in the test data, $T_{train \cap test}$ the set of themes that occur in both the train and the test data, and count(t) the number of occurrences of theme t in either of the sets. Then the maximal obtainable recall is:

$$\text{Recall}_{\max} = \frac{\sum_{t \in (T_{\text{train}} \cap T_{\text{test}})} \text{count}(t)}{\sum_{t \in T_{test}} \text{count}(t)} \tag{6}$$

Table 4 shows that training on the data from 1994 gives much better results than training on the data from 1986. Adding the 1986 data to the 1994 data results in a slightly lower recall but an improvement in MAP. The lower recall is probably due to an increase of the number of themes in the training data, making the classification problem more complex. Because of changes in the theme set, adding a new year to the training data tends to add more confusion. For example, having both the themes 'fishing industry' from 1986 and 'fishing industry policy' from 1994, which cover the same topics but in different years, yields a more complex classification problem than having only one of them.

Table 4 also shows that the experiments with the (1986 and) 1994 training data do not reach the level of the cross validation experiment. This means that the data from earlier years are only partly representative for the 1998 data. We investigated the effect of non-overlapping themes between the years: 42 themes from the 1998 data do not occur in the training data (e.g. 'bed capacity', 'city transport', 'France') and 143 themes in the training data (either from 1986 or 1994 or both) do not occur in the 1998 data (e.g. 'Aids', 'property tax').

Of course, better results could be obtained if we would use the theme set from 1998 to train the classifier on. However, this is not realistic since in the future application of the classifier it should be able to classify texts from years for which the theme set is unknown. Since the theme set of the test year will be unknown, we not only aim at selecting the best training data but also the best theme set from the training data. We did an experiment in which we used the data from 1986 and 1994 to train on, but changed the theme set: all themes from 1986 + 1994 or only the 1994 themes. The results of this experiment are in Table 5. We also included the results for using the 1998 theme set (third row) as reference.

The first row of Table 5 is the same as the fourth row in Table 4. The second row of Table 5 shows that by keeping only the themes from the most recent year in the training data we obtain a higher precision. The higher recall in the third row of Table 5 shows that non-overlapping themes are indeed a problem for classifying across years. We continue our experiments with the data from both 1986 and 1994 as training data, but using only the theme set from 1994.

[8] Note that this means that recall can never be 100% since there are text segments with more than 10 themes. However, a higher value for maxranks will penalize the precision too much.

[9] Cross validation is not possible for the data from 2002 onwards because we do not have labelled data for those years.

Table 4
Results for the automatic classification of the 826 texts fragments from the 1998 election manifestos using different sets of training data and a standard bag-of-words representation. 'P' stands for precision and 'R' for recall.

Training data	# Training texts	# Themes	*P* (%)	*R* (%)	*F*1 (%)	MAP	Max recall (%)
random guessing baseline		320	5.5	4.4	4.9	0.118	100.0
1986 data	797	214	34.9	29.1	31.7	0.546	58.5
1994 data	951	211	68.8	39.2	49.9	0.798	79.5
1986 and 1994 data	1748	320	68.8	37.3	48.4	0.835	83.1
cross val. on 1998 data	743	218	80.7	50.3	61.9	0.910	100.0

Table 5
Results for the automatic classification of the 826 texts fragments from the 1998 election manifestos, trained on the data from 1986 and 1994.The first row is identical to the fourth row in Table 4. In the second row are the results from classification in which only themes that occur in the most recent training year (1994) are kept. In the third row, only themes that occur in the test data (1998) are kept in the training data. 'P' stands for precision and 'R' for recall.

theme set	# training texts	# themes	*P* (%)	*R* (%)	*F*1 (%)	MAP	Max recall (%)
1986 + 1994	1748	320	68.8	37.3	48.4	0.835	83.1
1994	1718	211	72.2	37.1	49.0	0.841	79.5
1998	1715	218	72.6	50.6	59.6	0.855	100.0

Table 6
Results for the automatic classification of the 826 texts fragments from the 1998 election manifestos using the text examples from the 1986 and 1994 data (themes from 1994) with different text representations. The asterisk denotes that the *F*-scores for the individual texts are significantly different ($P < 0.01$) from to the unigram baseline. '*P*' stands for precision and '*R*' for recall. Boldface indicates the highest scoring setting per evaluation measure.

Text representation	*P* (%)	*R* (%)	*F*1 (%)	MAP
Unigrams	72.2	**37.1**	**49.0**	0.841
Unigrams lemmatized	**73.3**	36.8	**49.0**	**0.850**
Unigrams lemmatized without stopwords	72.9	36.0	48.2	0.839
Unigrams + bigrams	68.8	36.4	47.6*	0.834
Unigrams + bigrams lemmatized	73.1	36.1	48.3	0.845
Unigrams + bigrams lemmatized without stopwords	72.4	36.0	48.1	0.837

5.2. Variants of the bag of words representation

There is evidence that text classification can be improved by adding word bigrams (Braga, Monard, & Matsubara, 2009; Bekkerman & Allan, 2004), also for Dutch (Gaustad & Bouma, 2002, p. 11), although the success depends on the domain and corpus size (Sebastiani, 2002). We extracted all within-sentence bigrams from the texts in our corpus and added them to the bag of unigrams for each text. We also lemmatized the texts using the output of the Dutch morpho-syntactic analyzer Frog (Van den Bosch, Busser, Canisius, & Daelemans, 2007) and experimented with lemmatized unigrams and bigrams. Finally, we experimented with removing stopwords from the feature set. The results of classification experiments with the variants of the data are in Table 6.

A paired *t*-test on F-scores for individual texts ($n = 826$) showed that the only settings that gives significantly different results from the unigram baseline is *unigrams + bigrams* ($P < 0.0001$) – yielding a lower average F-score than the baseline. Since none of the alternative text representations gives results above the unigram baseline, we decide to continue our experiments with unigrams only, wordforms rather than lemmas and without stopword removal.

5.3. Merge themes

In the experiments presented thus far we have trained on a relatively large number of themes in a relatively small number of texts: 210 themes[10] occurring in 1718 texts (1986 + 1994 data). We investigated whether theme merging can be profitable for classification performance, because we saw that Lipschits chose to formulate distinct themes for very similar content (e.g. 'Aruba' and 'Netherlands Antilles'). For the purpose of reducing the number of themes, we clustered similar themes together. For each theme in the training data, we created a subcorpus by concatenating all texts with this theme. Then we created a term vector with relative term frequencies for all (lemmatized) terms occurring in the theme subcorpus. For each pair of themes, we calculated the cosine similarity. For example, the similarity between 'teachers' and 'school' is 0.98 while the similarity between 'teachers' and 'economic policies' is 0.74, and the similarity between 'teachers' and 'weapons' is 0.26. We then clustered the themes using an agglomerative hierarchical clustering approach (Day & Edelsbrunner, 1984):

[10] This would have been 320 if we would have kept the 1986 classes.

Table 7
Statistics for the similarity-based merging of themes in the 1986 + 1994 training data. For each similarity threshold, the resulting number of themes and the average number of texts per theme are shown, together with the classification performance ($F1$) for the 1998 data, the entropy $H(T')$ of the theme distribution and the average proportion of themes on a text (A). The last column shows the product of H and A, as a heuristic for the optimal similarity threshold. We do not show all 0.05 steps of the merging process. Boldface indicates the highest scoring setting per evaluation measure.

Threshold	# Themes	Avg. # Texts per theme	$F1$ (%)	$H(T')$	$A(T)$	$H * A$
No merging	210	52	48.9	3.16	0.03	0.095
0.95	183	50	49.6	3.09	0.03	0.093
0.9	153	52	50.8	3.00	0.03	0.090
0.85	120	58	52.1	2.88	0.03	**0.086**
0.8	96	68	54.2	2.77	0.04	0.111
...						
0.4	18	189	56.6	1.24	0.11	0.136
0.35	15	209	58.3	1.07	0.12	0.128
0.3	8	356	**58.7**	0.78	0.21	0.164
0.25	7	407	**58.7**	0.77	0.24	0.185
0.2	4	536	55.9	**0.31**	0.31	0.096

1. The clustering process is initiated by making each theme its own cluster.
2. Iteratively, the two clusters are selected that are the most similar following a complete-linkage strategy: The similarity between cluster A and cluster B $sim(A, B)$ is equal to the lowest similarity $sim(a, b)$, where a is a theme in cluster A and b is a theme in cluster B.
3. Stop when there are no cluster pairs with a similarity above a given threshold. As thresholds we tested a cosine similarity range of $(1.0 \ldots 0.2)$ in steps of 0.05.

For each of the threshold values, we replaced the original themes by the merged themes in the example data (both the training and the test set). Then we performed a number of analyses on the resulting data in order to decide on the optimal similarity threshold for merging: (1) we evaluated the classification performance (in terms of $F1$) with the merged data when classifying the 1998 data; (2) we calculated the entropy of the theme distribution:

$$H(T') = -\sum_{t' \in T'} p(t') * \log(p(t')) \tag{7}$$

in which T' is the set of theme combinations that occur in the data (t' is one unique combination of themes, occurring for at least one text); and (3) we counted the proportion of themes on average attached to a text:

$$\mathrm{A(T)} = \frac{\sum_d t_d / t}{D} \tag{8}$$

in which D is the number of texts, t_d is the number of themes on a text and t is the total number of themes in the set.

The rationale behind (2) and (3) is to find the optimal level of granularity: On the one hand we would want a classification problem with low entropy, making the classification problem easier, and on the other hand we do not want to move too far from the original fine grained Lipschits classification: if in the extreme case some texts in the set are labelled with all themes, then this labelling makes no sense. Therefore we used the relative number of themes on a text $A(T)$ as an additional criterion. Table 7 shows the calculated statistics.

Table 7 shows that the lowest entropy is obtained for a very low number of themes (4). Although this makes the classification problem easier (highest $F1 = 58.7\%$), it is no longer a sensible labelling in the view of the original Lipschits classification. If we multiply entropy with the relative number of themes on a text, then we find the lowest (optimal) $H * A$ at the similarity threshold of 0.85. At that point, the number of themes has been reduced from 210 to 120. Among these 120, there are 37 clusters of merged themes and 83 themes that were not merged. Examples of merged themes are:

- development aid, international economic cooperation, European cooperation, human rights, United Nations, international monetary collaboration, Middle and Eastern Europe,
- banks, insurance companies, pension funds,
- urban renewal, housing, spatial planning, house ownership, municipal administration and policy, provincial administration and policy, recreation, land policy,
- Aruba, Netherlands Antilles.

We see that conceptually similar themes are clustered for a similarity threshold of 0.85 on the 1994 theme set. After optimization of the 1994 theme set, we applied the theme merging to the 1998 theme set. We asked a domain expert, a political journalist, to give her opinion on three versions of the 1998 theme set:

1. the set of 101 themes after merging similar themes with a threshold of 0.85 (optimum for the 1994 data);
2. the set of 140 themes after merging similar themes with a threshold of 0.90 (a more conservative merging strategy);
3. the set of 218 original Lipschits themes, without merging.

Table 8
A comparison of the automatic classifier trained on the 1986 + 1994 data, automatically evaluated on the 826 labelled texts from the 1998 data, and the classification by Lipschits himself, manually evaluated by a human expert. 'P' stands for precision and 'R' for recall.

Year	# Texts	# Themes/text	*P* (%)	*R* (%)	*F*1 (%)	MAP
Automatic classifier	826	4.2	72.2	37.1	49.0	0.841
Lipschits' classification	50	7.7	71.5	89.0	79.3	0.843

Table 9
Statistics on the unseen data.

Year	# Words	Vocabulary size	# Texts	# Words/text
2006	235,949	35,547	4771	49.5
2010	756,254	35,547	21,329	35.5
2012	246,004	30,477	5880	41.8

According to our expert, it would be very difficult to judge the merged themes as relevant or irrelevant because some parts may be relevant and others may not. In addition, in the theme set resulting from the 0.85 threshold, some of the merged themes were noisy, such as 'sports policy, crime prevention, law enforcement, police policy, addiction'. The theme set resulting from the 0.90 threshold was less noisy but still the expert found the combinations of themes too broad to judge. For example, she judged the combined 'transport' theme ('car traffic, public transport, freight transport, airports and aviation, canal shipping, sea ports and sea shipping') as irrelevant when the text was about airports. Therefore, we decided not to merge the themes from the 1998 set but keep the original, fine-grained Lipschits theme set.

5.4. Comparing the automatic classifier to Lipschits himself

As a reference for the quality of the automatic classifier, we asked a domain expert to manually evaluate the labels assigned by Lipschits himself (for more details see Section 6). When presenting the domain expert with 50 randomly selected texts from the 1998 data, she thought they were automatically labelled. The quality of the manual classification can be seen as a reasonable upper bound for our classifier. The results are in Table 8.

When we compare the two rows in Table 8, we see that the Lipschits assigned more themes per text than our automatic classifier, leading to a higher recall and *F*1 score, but that precision is almost equal.

6. Labelling unseen election manifestos

6.1. Annotation and evaluation setup

We built an automatic Lipschits classifier that can assign themes to unseen election manifestos. We trained the classifier on all available data from 1986, 1994 and 1998, but only kept the themes from the most recent year: 1998. We used a standard bag-of-words representation without lemmatization and removal of stopwords. We configured the classification system to assign a minimum of 1 and a maximum of 10 labels to each text fragment, with a Winnow threshold of 1.

We obtained the unlabelled election manifestos from 2006, 2010 and 2012.[11] These manifestos were digitally born, published as PDF files, and automatically turned into HTML while trying to preserve the paragraph structure. The HTML files were manually checked on paragraph breaks and bad breaks were removed. The paragraphs in the unlabelled data are considerably shorter than the text fragments in the training data: around 40 words on average (in the older Lipschits data, multiple paragraphs were often brought together under one topic number). Statistics on the data from 2006 onwards are in Table 9.[12]

We automatically labelled the texts using our Lipschits classifier. Then we asked a domain expert, a political journalist, to manually evaluate a sample of the automatic annotations. The assessment interface is shown in Fig. 3.

We measured precision and recall by counting the number of assigned themes judged as relevant, the number of assigned themes judged as not relevant[13] and the number of themes that the expert typed in the field for missing themes:

$$\mathrm{P} = \frac{\#\text{assigned and relevant}}{(\#\text{assigned and relevant} + \#\text{assigned and wrong})} \tag{9}$$

and

$$\mathrm{R} = \frac{\#\text{assigned and relevant}}{(\#\text{assigned and relevant} + \#\text{missing})} \tag{10}$$

[11] The files for 2006, 2010 and 2012 came from verkiezingskijker.nl (Jijkoun et al., 2007), a manifesto search tool made by Google for the 2010 elections, and www.watzegtmijnpartij.nl, respectively.

[12] We do not know why the 2010 corpus is much bigger than the 2006 and 2012 corpora.

[13] The option 'don't know' was selected by the expert only in two instances. These were counted as irrelevant for the presented text.

Table 10
Results for the automatic classification of 193 text fragments from 2006 onwards into the 218 themes defined by Lipschits in 1998. The themes were manually evaluated by an expert. The 1998 results from Table 8 are included as reference. '*P*' stands for precision and '*R*' for recall.

Year	# Texts	# Themes/text	*P* (%)	*R* (%)	*F*1 (%)	MAP
2006 (Trained on 86 + 94 + 98)	39	1.7	74.2	45.8	56.6	0.683
2010 (Trained on 86 + 94 + 98)	122	1.9	77.4	55.0	64.3	0.714
2012 (Trained on 86 + 94 + 98)	32	2.0	84.6	56.1	67.5	0.766
1998 (Manually by Lipschits)	50	7.7	71.5	89.0	79.3	0.843

Table 11
Analysis of the themes that the classifier failed to assign (misses) to the texts from 2006, 2010 and 2012 according to our domain expert in the evaluation of the classifier.

	2006	2010	2012
Number of missing themes	58	142	43
Missed theme does exist in 1998 (exact match or synonym) (%)	41	44	37
Missed theme is related to a 1998 theme (%)	38	43	47
Missed theme is a completely new theme (%)	21	13	16

6.2. Evaluation

The expert assessed 200 text fragments picked randomly from the data. She judged seven of these as not assessable because they were too short, so 193 texts remain. She was expected to actively add themes if she thought a theme was missing. Although there was a list of themes available through auto-completion, the expert often chose to formulate new terms.[14] The results are in Table 10.

The data from 2006 seem more difficult to classify than the data from 2010 and 2012, seen from the lower Precision, Recall and *F*-score obtained for that year. However, this difference is not significant according to Welch's *t*-test.[15]

When we compare the results obtained with the automatic classification of the 2006–2012 data to the evaluation of the manual Lipschits labelling (the last row), we see that although recall is substantially lower, its precision is higher. This means that our automatic Lipschits classifier is able to emulate Lipschits' precision, but is not as complete as Lipschits himself. We performed Welch's *t*-test on *F*-scores in the 2006–2012 sample ($n = 193$) and the *F*-scores in the 1998 sample ($n = 50$). The *P*-value for the difference between the means is 0.038.

One of the likely reasons for the difference is the effect of the 'theme gap': for themes in the 2012 data that did not exist yet in 1998 we did not have any training data. In order to estimate this effect, we performed an error analysis: We manually went through the 243 false misses that the expert reported for the 2006–2012 data and categorized them in three groups:

1. The missed theme already existed in the 1998 data, possibly with different spelling or in the form of a synonym of the term formulated by the expert (e.g. 'care' vs. 'health care', and 'environment' vs. 'environmental policy');
2. The missed theme is related to another theme in the 1998 data, possibly more general or more specific (e.g. 'public transport' vs. 'railway transport', 'small and medium enterprises' vs. ' businesses', 'employment' vs. 'unemployment') but the missed theme is not new;
3. The missed theme is completely new compared to the 1998 theme set (e.g. 'nutrition', 'European regulations', 'taxi law', 'games', 'public broadcasting', 'social coherence').

We counted the themes in each category over the years; the results are in Table 11.

7. Conclusions

We digitized three years of Dutch election manifestos annotated by the Dutch political scientist Isaac Lipschits: 2574 short (~300-words) texts that Lipschits labelled. There are more than six political themes per text on average. We used these data to train a classifier that can automatically label new, unseen election manifestos with themes.

The general conclusions that we draw from this paper are (1) in the thematic classification of political texts, a high level of detail seems to be preferred by domain experts; (2) change of themes over the years affects recall of the learned classifier, but (3) precision is comparable to the precision obtained by a human expert labeller. Thus when using old political texts to classify new texts, work is needed to link and expand the set of themes to newer topics.

[14] The possibility of formulating new themes was a necessary addition because texts from more recent years may contain newly emerged concepts and political themes which are not covered in the themes specified by Lipschits.

[15] The *P*-value for the difference between the 2006 *F*1 mean and the 2012 *F*1 mean is 0.37 ($n_1 = 39, n_2 = 32$).

In future work, we will create a faceted search interface (using the themes as facets) for the digitized election manifestos, so that our work can be used by interested researchers, students and journalists. It may be interesting to investigate the trade-off between false hits (precision) and false misses (recall) in the context of faceted search. A second direction of future work is to investigate approaches to expand a political theme set with recently emerged themes. This could be done using a topic identification method (Quinn et al., 2010) on contemporary political texts.

Acknowledgements

This research was supported by the Netherlands Organization for Scientific Research (NWO) under Project No. 380-52-005 (PoliticalMashup). We thank Marjolein Bax for her annotation work, and Lars Buitinck for his help with the OCR'ed texts.

References

Awadallah, R., Ramanath, M., & Weikum, G. (2010). Language-model-based pro/con classification of political text. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval* (pp. 747–748).

Bekkerman, R., & Allan, J. (2004). Using bigrams in text categorization. 1003, Amherst: Department of Computer Science, University of Massachusetts.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python: Analyzing text with the natural language toolkit*. O'Reilly. <http://www.nltk.org/book>.

Braga, I., Monard, M., & Matsubara, E. (2009). Combining unigrams and bigrams in semi-supervised text classification. In *EPIA* (Vol. 9, pp. 489–500).

Breeman, G., Lowery, D., Poppelaars, C., Resodihardjo, S. L., Timmermans, A., & De Vries, J. (2009). Political attention in a coalition system: Analysing queen's speeches in the Netherlands 1945–2007. *Acta Politica, 44*(1), 1-1.

Budge, I., Klingemann, H. D., Volkens, A., Bara, J., & Tanenbaum, E. (2001). *Mapping policy preferences: Estimates for parties, electors, and governments 1945–1998* (Vol. 1). USA: Oxford University Press.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297.

Dagan, I., Karov, Y., & Roth, D. (1997). Mistake-driven learning in text categorization. In *Proceedings of the second conference on empirical methods in NLP* (pp. 55–63).

Day, W. H., & Edelsbrunner, H. (1984). Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification, 1*(1), 7–24.

Van den Bosch, A., Busser, B., Canisius, S., & Daelemans, W. (2007). An efficient memory-based morphosyntactic tagger and parser for dutch. In *Computational linguistics in the Netherlands: Selected papers from the seventeenth CLIN meeting* (pp. 99–114).

D'hondt, E., Verberne, S., Koster, C., & Boves, L. (2013). Text representations for patent classification. *Computational Linguistics, 39*(3), 755–775.

Diermeier, D., Godbout, J. F., Yu, B., & Kaufmann, S. (2012). Language and ideology in congress. *British Journal of Political Science, 42*(01), 31–55.

Forman, G. (2006). Tackling concept drift by temporal inductive transfer. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval. SIGIR '06* (pp. 252–259). New York, NY, USA: ACM.

Gaustad, T., & Bouma, G. (2002). Accurate stemming of dutch for text classification. *Language and Computers, 45*(1), 104–117.

Gielissen, T., & Marx, M. (2009). The design of poliDocs: A web information system for the disclosure of dutch parliamentary publications. In *Sixth international workshop on web information systems modeling (WISM 2009)*.

Hillard, D., Purpura, S., & Wilkerson, J. (2007). An active learning framework for classifying political text. In *Annual meeting of the midwest political science association*.

Hillard, D., Purpura, S., & Wilkerson, J. (2008). Computer-assisted topic classification for mixed-methods social science research. *Journal of Information Technology & Politics, 4*(4), 31–46.

Hirst, G., Riabinin, Y., & Graham, J. Party status as a confound in the automatic classification of political speech by ideology. In *Proceedings of JADT 2010*.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). Practical guide to support vector classification.

Jijkoun, V., Marx, M., De Rijke, M., & van Waveren, F. (2007). Electoral search using the verkiezingskijker: An experience report. In *Proceedings of the 16th international conference on World Wide Web* (pp. 1155–1156). ACM.

Joachims, T. (1999). Svmlight: Support vector machines. SVM-Light support vector machines (Vol.19(4)). University of Dortmund. <http://svmlightjoachimsorg/>.

Kaptein, R., & Marx, M. (2010). Focused retrieval and result aggregation with political data. *Information Retrieval, 13*(5), 412–433.

Koster, C., & Beney, J. (2007). On the importance of parameter tuning in text categorization. *Perspectives of Systems Informatics*, 270–283.

Koster, C., & Seutter, M. (2003). Taming wild phrases. *Advances in Information Retrieval*, 78-78.

Koster, C. H. A., Seutter, M., & Beney, J. (2003). Multi-classification of patent applications with winnow. In M. Broy & A. V. Zamulin (Eds.), *Ershov memorial conference. Lecture notes in computer science* (Vol. 2890, pp. 546–555). Springer.

Laver, M. (2001). *Estimating the policy positions of political actors* (Vol. 20). Psychology Press.

Lipschits, I. Verkiezingsprogrammas 1977: verkiezingen voor de Tweede Kamer der Staten-Generaal. 1977. Subsequent editions appeared for the elections in 81, 86, 89, 94, 98.

Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning, 2*(4), 285–318.

Louwerse, T. (2011). *Political parties and the democratic mandate. comparing collective mandate fulfilment in the United Kingdom and the Netherlands*. Ph.D. Thesis; LEI Universiteit Leiden.

Marx, M. (2009). Advanced information access to parliamentary debates. *Journal of Digital Information, 10*(6).

Mourão, F., Rocha, L., Araújo, R., Couto, T., Gonçalves, M., & Meira, W. Jr., (2008). Understanding temporal aspects in document classification. In *Proceedings of the international conference on Web search and web data mining* (pp. 159–170). ACM.

Pouliquen, B., Steinberger, R., & Ignat, C. (2003). Automatic annotation of multilingual text collections with a conceptual thesaurus. In *Proceedings of the workshop ontologies and information extraction (EUROLAN'2003)* (pp. 9–28).

Purpura, S., & Hillard, D. (2006). Automated classification of congressional legislation. In *Proceedings of the 2006 international conference on digital government research. Digital government society of North America* (pp. 219–225).

Quinn, K., Monroe, B., Colaresi, M., Crespin, M., & Radev, D. (2010). How to analyze political attention with minimal assumptions and costs. *American Journal of Political Science, 54*(1), 209–228.

Rocha, L., Mourão, F., Pereira, A., Gonçalves, M., & Meira, W. Jr., (2008). Exploiting temporal contexts in text classification. In *Proceeding of the 17th ACM conference on information and knowledge management* (pp. 243–252). ACM.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR), 34*(1), 1–47.

Steinberger, R., Ebrahim, M., & Turchi, M. (2012). JRC eurovoc indexer JEX – a freely available multi-label categorisation tool. In *Proceedings of the eight international conference on language resources and evaluation (LREC'2012)*. Istanbul, Turkey (ISBN:978-2-9517408-7-7).

Van Mun, P. (1999). Text classification in information retrieval using winnow. Technical report, Catholic University of Nijmegen.

Yu, B., Kaufmann, S., & Diermeier, D. (2008). Classifying party affiliation from political speech. *Journal of Information Technology & Politics, 5*(1), 33–48.

Zhu, M. (2004). *Recall, precision and average precision*. Waterloo: Department of Statistics and Actuarial Science, University of Waterloo (Working paper).