An Intelligent System for the Categorization of Question Time Official Documents of the Italian Chamber of Deputies

A. Cavalieri[a], P. Ducange[b], S. Fabi[c], F. Russo[d], N. Tonellotto[b]

[a]Department of Cultures, Politics and Society, University of Torino, Torino, Italy;
[b]Department of Information Engineering, University of Pisa, Pisa, Italy;
[c]NBS srl, San Benedetto del Tronto, Ascoli Piceno, Italy;
[d]Department of History, Society and Human studies, University of Salento, Lecce, Italy.

ARTICLE HISTORY
Compiled January 13, 2023

Abstract
In this work we present an intelligent system for the automatic categorization of political documents, specifically the documents containing the parliamentary questions collected during the weekly Question Times at the Chamber of Deputies of the Italian Republic. The proposed intelligent system leverages text classification models to perform the document categorization. The system is aimed at supporting and facilitating the research activities of political science scholars, who deal with comparative and longitudinal analysis of thousands of documents. To select the best classification models for our specific task, several classical machine learning and deep learning-based text classification models have been experimentally compared.

KEYWORDS
Text Classification, Bag-of-Words, Word Embeddings, Machine Learning, Deep Learning, Comparative Agendas Projects, Parliamentary Texts

1. Introduction

The pervasive availability of Internet-connected devices, and the digitalization of business processes in public administrations, banks, hospitals and private companies, has favored an ever-growing availability of digital contents in many contexts such as medicine, social sciences, healthcare, psychology, law, engineering, etc (Hussain, 2017). Within political science, scholars of legislative studies increasingly resort to the analysis of official documents generated within parliaments to study party politics, policy-making and broader questions related to political representation. In recent times, most democratic parliaments have released the archives of their official documents in different digital formats. In doing so, vast collections of textual data regarding parliamentary debates, legislative bills and parliamentary questions are now easily accessible. The availability of these information sources has stimulated the interest of legislative and political researchers for computer-aided techniques able to automatically extract information and to provide analytics and decision services from the political texts (Grimmer & Stewart, 2013; Hillard, Purpura, & Wilkerson, 2008; Slapin & Proksch, 2014; van Atteveldt, Welbers, & van der Velden, 2019).

Corresponding author: P. Ducange. Email: pietro.ducange@unipi.it.

The Comparative Agendas Project[1] (CAP) is a network of political scientists around the world collecting and classifying various textual documents on political topics, in order to detect the issues that are debated across countries over time (Baumgartner, Breunig, & Grossman, 2019). CAP enables scholars, students, policy-makers and the media to investigate trends in policy-making across time and between countries. Among its activities, CAP collects information on policy activities and classifies them into a single, universal and consistent coding scheme. The key to make the collected data comparable is the adoption of a common coding system consisting of 21 major topics and more than 200 sub-topics to code those activities. To date, CAP covers 21 political systems around the world, forming the *CAP coding system* or *CAP master codebook*. CAP monitors policy processes by tracking the actions that governments take in response to the challenges they face. These activities can take many different forms, including debating a problem, delivering speeches, e.g., the Queen's speech in the United Kingdom, holding hearings, introducing or enacting laws, e.g., Bills and Public Laws in the United States, or issuing judicial rulings, e.g. rulings from the European Court of Justice. Among these activities, Parliamentary Questions (PQs) are tools that can be used by members of parliaments to force ministers to justify their actions and explain their policy choices. All democratic parliaments have at least one procedure to allow their members to ask questions to the government. Scholars involved in the CAP often monitor the policy content of PQs for oral answer dealt with during plenary sittings to understand the issues emphasized by political parties (Borghetto & Chaques-Bonafont, 2019). The PQ documents classified by CAP are provided as semi-structured, heterogeneous textual data in many different formats. Scholars of the CAP community initially relied on trained students to classify the content of documents, but the huge amount of data produced by political institutions made soon apparent the need to perform document classification with automatic text analytics systems (Burscher, Vliegenthart, & De Vreese, 2015; Collingwood & Wilkerson, 2012; Loftis & Mortensen, 2020).

Text classification is a technology that allows us to categorize unstructured texts encoded in digital documents. For this purpose, a text preprocessing phase plays a key role. In fact, during this phase, unstructured texts are transformed into structured data suitable for feeding classification models (Khan, Baharudin, Lee, & Khan, 2010; Mirończuk & Protasiewicz, 2018; Onan, Korukoğlu, & Bulut, 2016). Over the last few decade, text classification problems have been widely studied and addressed in many real application contexts, such as smart cities (D'Andrea, Ducange, Lazzerini, & Marcelloni, 2015), medicine (Hughes, Li, Kotoulas, & Suzumura, 2017), social media analysis (Yoo, Song, & Jeong, 2018), psychology (Burdisso, Errecalde, & Montes-y Gómez, 2019), sentiment & opinion identification (Onan, 2018, 2020a, 2020b), and political science (Grimmer & Stewart, 2013). Most of text classification approaches discussed in the specialized literature are based on machine learning and artificial intelligence models (Ikonomakis, Kotsiantis, & Tampakas, 2005; Kowsari et al., 2019), such as Support Vector Machines, Bayesian classifiers, Logistic Regressions classifiers, Convolutional Neural Networks, Long Short Term Memory Neural Networks and transformer architectures (Onan & Toçoğlu, 2021; Wolf et al., 2020). Moreover, in order to provide a text as an input of a classifier, the text must be transformed into a numerical vector. To this aim, several text representation approaches have been discussed in the specialized literature. Among them, the most relevant are those based in Bag Of Words (BOW) (Zhang, Jin, & Zhou, 2010) and Word Embeddings (WE) (Kusner,

[1] `https://www.comparativeagendas.net`

Sun, Kolkin, & Weinberger, 2015; Liu, Liu, Chua, & Sun, 2015; Onan, 2019).

Within the context of the CAP community, scholars followed two approaches to utilize machine learning. On the one hand, several researches demonstrated the usefulness of text classifiers without specifying the software they used (Collingwood & Wilkerson, 2012; Hillard et al., 2008). On the other hand, a group of studies explicitly relied on specialized packages to be used with the free software R. In particulr, the package RTextTools (Jurka, Collingwood, Boydstun, Grossman, & van Atteveldt, 2013) has been widely used within the CAP community, but it is no longer maintained. More recently, political scientists are turning to the R package quanteda (Benoit et al., 2018), which offers many tools for quantitative textual analysis including a Naive Bayes algorithm for text classification. Both packages are primarily addressed to researchers with some programming knowledge, limiting the target audience to a minority of political scientists. Political scientists sometimes resort also to other R packages such as mlr (Bischl et al., 2016) and CARET (Kuhn, 2008) to exploit machine learning techniques in their research activities. However, both of them require solid programming skills.

In this work, we discuss the results of a multi-disciplinary research activity involving political scientists and computer scientists researchers. Specifically, we describe the design, implementation and testing of an Intelligent System (IS) aimed at automatically classifying and cataloguing political texts according to a predetermined classification scheme. The purpose of the system is to make available a large set of state-of-the-art text classification methods for political scientists with little or no programming skills. In particular, the system was experimented to classify the text of Parliamentary Questions presented during the weekly *Question Time* (QT) in the Italian Chamber of Deputies according to the major topics of the CAP coding system. However, users can train the system with different data according to their research goals.

The proposed IS is based on text classification models. Indeed, in order to determine the most effective models to embed into our system, we carried out an experimental study in which we implemented, trained and compared a collection of state-of-the-art approaches for text representation and classification. The most accurate text classification models were integrated into the actual final system, implemented as a user-friendly Web application, that we called QTIS. Whenever a new collection of institutional documents, specifically PQs, is produced for the QT, the application allows users to upload the documents and to automatically associate each PQ to a CAP topic. In order to carry out the experimental campaign for selecting the most accurate text classification models, we adopted a real dataset composed by 5,672 PQs, collected during the QTs of the Italian Chamber of Deputies, spanning from 1996 to 2019. Moreover, after selecting the most accurate text classification models, we adopted the QTIS Web application to classify 222 unlabelled PQs, from July 2019 to May 2020. A team of political scientists belonging to the CAP Italian Team evaluated the report generated by QTIS and confirmed that the systems allowed us to correctly recognize up to 73% of the documents containing PQs and classified by the QTIS Web application.

The paper is organized as follows. Section 2 describes the most recent works related with the study carried out in this work. Section 3 briefly introduces the QT in Italy and the dataset adopted in experimental comparison campaign. Section 4 describes the experimented methods for text representation and classification. Section 5 discusses the details of the experimental comparison campaign, by showing and comparing the results achieved by 12 text classification models. Section 6 gives details on the design and implementation of the proposed QTIS and presents the results of the analysis carried out using QTIS on a previously unseen collection of unlabelled PQs. Finally, Section 7 summarizes our findings and provides directions for future work.

2. Related Work

To date, the CAP community has yielded a rich corpus of studies about policy processes, the result of a shared time-consuming and costly human-labelling effort – approximately 400 publications (Baumgartner, Jones, & Mortensen, 2017). The increasingly accessible and impressive body of textual data, along with the outstanding development of computing power, have made the use of machine learning and artificial intelligence techniques extremely appealing even for public policies researchers. Additionally, because the policy agenda is not static over time because of the changing dynamics and actors that drive its contents (Mettler, 2016), the use of such techniques would also allow retroactive adjustments (Gilardi & Wüest, 2020), particularly useful in case of long term projects such as the CAP, which are too costly to implement otherwise. Broadly speaking, the crucial aspect of adopting machine learning and artificial intelligence methods lies in the choice of using automatic tools, based on unsupervised or supervised models, "to classify documents into classes" (Loftis & Mortensen, 2020). As discussed in the following paragraphs, these tools can be used as an alternative or as a support to manual labelling.

Although the tools based on unsupervised models limit the involvement and the tasks of policy agendas scholars in the initial design stages of the models, thereby minimizing pre-analysis costs, a laborious ex-post validation process is often required. Usually, this process needs to combine experimental, statistical and substantive evidences to prove the conceptual validity of the classification results (Quinn, Monroe, Colaresi, Crespin, & Radev, 2010). Moreover, since policy categories come out of data inductively as results of mainly clustering algorithms (Grimmer & King, 2011) and estimating statistical topic models Grimmer (2010), it may raise doubts about the validity of categories themselves. As matter of fact, unsupervised techniques seem rather unfitting for the characteristics and purposes of the CAP, mostly because of the risk of mismatching with the 21 major topics of the CAP master codebook (Baumgartner et al., 2019).

Therefore, the tools based on supervised learning models are those where policy agendas scholars have put much effort into (Burscher et al., 2015; Collingwood & Wilkerson, 2012; Hansen, Navarretta, Offersgaard, & Wedekind, 2019; Hillard et al., 2008; Jurka et al., 2013; Navarretta & Hansen, 2020; Purpura & Hillard, 2006), especially for the manual labelling of the texts. In the pioneering work discussed in Purpura and Hillard (2006), the authors applied a two-phase hierarchical approach to a prototype of a Support Vector Machine (SVM) to classify the US Congress legislative texts using the CAP coding scheme and human-labelled samples. In Hillard et al. (2008) and Collingwood and Wilkerson (2012), the authors discussed the application of ensemble models, including SVMs and bayesian classifiers, for the classification of documents extracted from the *The Congressional Bills Project.*[2] This project includes more than 400,000 bills introduced in the U.S. Congress. Another interesting dataset recently analyzed in a few works, such as those discussed in Hansen et al. (2019) and in Navarretta and Hansen (2020), is the *The Danish Parliament Corpus 2009 – 2017*, which contains transcripts of parliamentary speeches from the sittings in the Chamber of the Danish Parliament[3]. In Hansen et al. (2019), the authors compared different approaches for text representations and select bayesian classifiers as the most performing solution for text classification. Experiments were carried out using a reduced

[2]http://www.congressionalbills.org

[3]https://repository.clarin.dk/repository/xmlui/handle/20.500.12115/8

corpus of manually labelled documents which considers 19 subject areas. Moreover, Navarretta and Hansen (2020) experimented different text classification models, based on SVM, logistic regression classifiers and multinominal naïve bayes classifiers, to predict the party of the politicians from the speeches. A four-class classification task was approached, considering the Danish People's Party, the Liberal Party, the Social Democratic Party and the Red-Green Alliance.

Because generalizability of the text classification models can be more easily achieved using training data that are "representative of all outlets, time periods, and document types that one wants to study", in Burscher et al. (2015) the authors experimented with a collection of supervised text classification models, considering different types of documents. More specifically, they extracted different training and test sets corpora from three well-known Dutch newspapers and Dutch PQs, considering the period between 1995 and 2011. The adopted models were based on passive aggressive learning classifiers and different text representation schemes.

A number of recent contributions (Loftis & Mortensen, 2020; Wiedemann, 2019) discussed the so-called *active learning workflow*, which alternates human labeling and machine learning labeling. Specifically, firstly an initial text classification model was built using a training set of documents manually labeled by policy agendas scholars. Then, the initial model was adopted as a decision support system for supporting the labeling of a new set of unlabeled documents: the system provides the estimation of the major topic to associate to each document while scholars evaluate the correctness of the provided estimation. In this way, the training set may be easily and quickly extended, and more accurate models may be generated. Loftis and Mortensen (2020) carried out experiments on a subset of texts of agenda items discussed in all Danish municipalities, extracted from the *Causes and Policy Consequences of Agenda Setting* project,[4] and on a subset of U.S. bills extracted for the aforementioned Congressional Bills Project. Text classification models based on bayesian classifiers and bag-of-words for text representation were adopted in the experimental analysis. Wiedemann (2019) instead conducted experiments on a data selection from the *Manifesto Project Database*[5], which includes electoral party manifestos worldwide. Bag-of-words representation with n-grams and latent semantic features were adopted for text representation along with a logistic regression classifier. Finally, Courtney, Breen, McMenamin, and McNulty (2020) tested automated translation of multilingual texts using supervised learning techniques, by constructing four different independent binary class classifiers based on the categories of the Comparative Agendas Project. They worked on a dataset of 4,485 randomly selected paragraphs of three different newspapers to perform computer accuracy tests. Albeit different from pure political documents, the authors confirmed the validity and reliability of machine learning-based models for text analysis.

However, despite all the studies discussed above, to the best of our knowledge, an easy and ready-to-use application, either paid or free, for the classification of political texts is not available for scholars. In fact, in each of the previously discussed studies, specific analyses are performed using models appropriately trained with data related to the context of the analysis. However, after the analyses no software has been released for allowing practitioners and researcher to repeat the analyses or make new experiments. For this reason, in this work, we present all the stages that led us to develop a user-friendly web application that allows CAP scholars to quickly categorize the QT documents of the Italian Chamber of Deputies. In particular, we consider in

[4]https://ps.au.dk/forskning/forskningsprojekter/capcas/

[5]https://manifestoproject.wzb.eu

our experimental comparison campaign the most promising text classification models adopted in the previously discussed works. Moreover, we also include in our experiments text classification models based on deep learning techniques, namely based on word embeddings for text representation and deep neural networks for text classification, deemed as a very promising venue for political science research (among others, Iyyer, Enns, Boyd-Graber, and Resnik (2014)). It is worth noticing that the proposed web application also allows users to easily carry out a text categorization analysis considering new and different types of document. This can be done by training, saving and using new text classification models for the user-specific type of document.

Overall, the main goal of this paper is not to propose a novel text classification scheme: our objective is to present a real application for the categorization of political documents, specifically those containing questions of Italian parliamentary QTs, based on consolidated state-of-the-art methodologies for text classification. Therefore, even though we also carried out a deep experimental campaign, in which we compared 12 different text classification models and a real world dataset of Italian parliamentary QTs documents, a comprehensive comparison among the huge amount of techniques for text classification, currently available in the literature, is out of the scopes of this work. However, to the best of our knowledge, our experimental campaign represents the first extensive text categorization analysis carried out on political documents written in Italian.

3. Dataset of the Italian QT

The Italian Team of the *Comparative Agendas Project* has assembled a dataset including information about PQs asked during the weekly question time session (*Interrogazioni a risposta immediata in assemblea*) in the Italian Chamber of Deputies (Russo & Cavalieri, 2016). The rules governing this parliamentary procedure are set by rule 135-bis of the Rules of Procedure of the Chamber of Deputies, which was introduced in 1997. However, already in 1996 some sessions were held to test the new rules. Since 1997 question time is held once per week, usually on Wednesday. All ministers, including the President of the Council of Ministers, appear before the Chamber of Deputies in rotation. Questions must be submitted the day before through the president of the parliamentary group. Each parliamentary group can present only one oral question per week. The questions must address a topic of general and urgent interest, and the debate on these questions is televised on the public broadcast company. The consolidated dataset, publicly available on the web-page of the CAP, includes 4,535 observations covering the period 1996-2014.[6] The policy content of each question is classified according to the CAP classification system, which has two levels: one level corresponds to 240 detailed topics (sub-topics) which are then grouped into 21 major topics (second level), identified by numbers ranging from 1 to 23[7]. Through manual labelling, each question has been assigned to a single major topic. In the experimental comparison campaign, we adopted an extended version of the dataset which includes 5,672 PQs, spanning from 1996 to 2019. Table 1 and Figure 1 show the details of the dataset. As we can see, also in the distribution of documents across the different major topics is extremely unbalanced. Taken together, the three most numerous ma-

[6]https://www.comparativeagendas.net/italy

[7]The CAP coding scheme is based on the scheme developed in the early 1990s within the US Policy Agendas Project (www.policyagendas.org). Over time some major topics were folded together, and this is the reason for the missing major topics number 11 and 22.

jor topics (Law and Crime, Government Operations, and Transportation) account for more that one third of the documents. By contrast, the three least populated major topics (Country Development, Public Lands, and Foreign Trade) jointly account for about 3.3% of them. Moreover, for experimenting the developed web application, we also adopted a very recent dataset composed by 222 PQs, spanning from July 2019 to May 2020. Details on this dataset can be found in Section 6.2.[8]

Table 1.: Distribution of the major topics in the experimental dataset.

Code	Description	Count	%
1	Domestic Macroeconomic Issues	452	7.97
2	Civil Rights and Minority Issues	150	2.64
3	Health	438	7.72
4	Agriculture	218	3.84
5	Labour and Employment	345	6.08
6	Education	292	5.15
7	Environment	255	4.50
8	Energy	115	2.02
9	Immigration	231	4.07
10	Transportation	549	9.68
12	Law and Crime	769	13.56
13	Welfare	116	2.05
14	C. Development and Housing Issues	78	1.38
15	Banking, Finance, and Commerce	437	7.70
16	Defence	143	2.52
17	Space, Science, Technology	104	1.83
18	Foreign Trade	31	0.55
19	International Affairs	264	4.65
20	Government Operations	524	9.24
21	Public Lands and Water Management	126	2.22
23	Cultural Policy Issues	35	0.62
	Total	5,672	

4. Text Representation and Classification

Text mining deals with extracting valuable knowledge from unstructured text contained in digital documents. Specifically, in this work, we consider text classification models, based on machine learning and artificial intelligence algorithms, for automatically assign a document to a single class, namely a category, selected among a finite set of different classes (Aggarwal & Zhai, 2012). To this aim, unstructured texts needs to be first transformed into structured numeric representations. Then, these representations are exploited as features used as input for machine learning models, which are in charge of estimating the class to assign to each specific text.

[8]The two datasets adopted in this work are still not publicly available. The Italian Team of the CAP plans to release a new version of the dataset after the termination of the current legislative period.

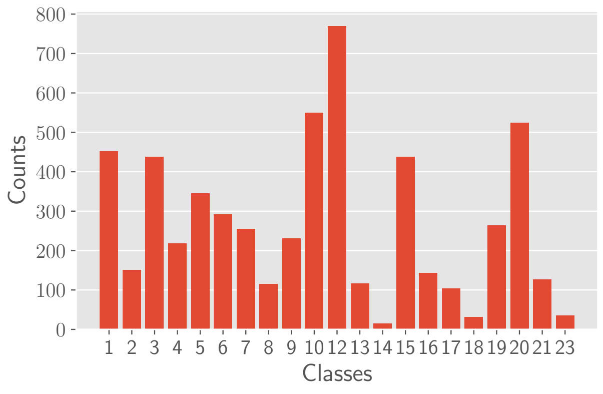

Figure 1.: Major topic distribution in the experimental dataset.

In the following, we describe the different text representation techniques and the classification models we experiment with.

Text representation The textual representation of a document, i.e., a string, needs to be transformed into a numeric vector before being elaborated by a classification model. This transformation is usually composed by two steps: firstly, the string is pre-processed to remove uninformative contents such as common words, punctuation and special symbols. Secondly, the pre-processed text is transformed to obtain a numerical representation of the original documents, i.e., a real-valued vector. More formally, given a document collection $D = \{d_1, \ldots, d_n\}$, every document d_i is pre-processed into a new textual representation $\delta_i \in \Delta$ through a text transformation function $f : D \mapsto \Delta$, and the resulting text is represented as a vector in $\mathbb{R}^d$ through a feature extraction function $g : \Delta \mapsto \mathbb{R}^d$.

The text pre-processing steps implemented in the text transformation f include:

(1) normalization: punctuation marks and special symbols, e.g., accents, hyphen, are removed from the original document text and the resulting text is lower-cased;
(2) tokenization: the stream of characters in the input text is transformer into an ordered list of processing units called tokens, e.g., words in our case;
(3) stopwords filtering: common and/or uninformative words providing little of no useful information, such as articles, conjunctions, prepositions and pronouns, are removed from the token list;
(4) stemming: each token is reduced to its stem or root form, so to group words having closely related semantics.

The feature transformation g aims at computing a numerical representation of the pre-processed text. We distinguish between *bag-of-words* (BOW) representation and *word embeddings* (WE) representation, characterized by different sizes d of the target vector space $\mathbb{R}^d$. In the BOW representation, every pre-processed text is represented as a sparse vector in $\mathbb{R}^{|V|}$, where V is a term vocabulary, e.g., in the simplest form, the set of all stems appearing in any pre-processed text. Given a pre-processed text δ_i, the entries in its BOW representation $\mathbf{x_i} \in \mathbb{R}^{|V|}$ corresponding to terms not appearing in the text are set to 0. The entries corresponding to terms appearing in the document can be computed according to some projection method, such as:

- term count: each entry corresponds to the number of occurrences of the term in the document;

- term frequency: each entry corresponds to the term count normalized by the total number of words in the document;
- term frequency-inverse document frequency: the term frequency is weighted by the inverse frequency of the term in the whole collection, a proxy of the term's specificity.

In the BOW representation, the dimension $|V|$ of the feature space to represent documents can be very large. In order to reduce the size of the feature space and to avoid high variance in the model's parameters estimation, several *feature selection* strategies can be used. In these strategies a subset of relevant terms is selected to be used in the model training. In our case, we set F_{max} as the maximum number of selected features, namely the maximum number of terms of the vocabulary used in the BOW representation. The feature selection strategy we adopted exploits the information gain as relevance measure for ranking all terms in the original vocabulary to select the best F_{max} terms to be used. More details on the BOW text representation and feature selection can be found in D'Andrea, Ducange, Bechini, Renda, and Marcelloni (2019).

In the BOW representation, each term is treated independently from the other terms, losing any relationships between terms appearing the same document. Conversely, in the embedding representation, every pre-processed text is represented in an e-dimensional vector space, with $e \ll |V|$. The embedding representation is based on the well-known word embeddings approach: each terms is represented in a p-dimensional vector space in such a way that semantically-related terms are mapped into closer vectors (Kusner et al., 2015). The elements of the WE vector space are learned through deep learning techniques (Liu et al., 2015) and p usually spans from 50 to 300 (we used $p = 300$ in our experiments). Given a pre-processed text δ_i, its constituent terms are mapped to the corresponding word embeddings, which are then combined, e.g., concatenated or summed up together, to compute the text representation. The most commonly used WE models, pre-trained on the Wikipedia corpus and available to be used directly for several languages (Grave, Bojanowski, Gupta, Joulin, & Mikolov, 2018), are:

- Word2Vec[9]: the word embeddings are learned with a neural network given the context of the word, i.e., a configurable window of co-occuring words;
- GloVe[10]: the word embeddings are learned by extending the Word2Vec learning algorithm with the latent semantic analysis matrix factorization method;
- FastText[11]: the word embeddings are computed by considering n-grams at character level and surrounding n-grams instead of single word and its surrouding words.

A WE captures the meaning of a given word, modeling it as a vector. In the case of multiple meanings of a given word that depend on the context, the WEare not able to correctly disambiguate the meaning. For example, the word *apple* may refer to the fruit or to the company: in both cases, the word is represented by the same WE. Recently, the introduction of the transformer architectures and contextualized language models (Vaswani et al., 2017) has paved the way to contextualized word embeddings, i.e., different representations of word, depending on the context of their usage. The main advantage of the transformer architecture and its implementations such as BERT (Devlin, Chang, Lee, & Toutanova, 2019) and GPT (Brown et al., 2020) with respect to causal language models based on recurrent neural networks, e.g., long-short term memories and gated recurrent units, lies in the adoption of an *attention*

[9]http://hlt.isti.cnr.it/wordembeddings/skipgram_wiki_window10_size300_neg-samples10.tar.gz
[10]http://hlt.isti.cnr.it/wordembeddings/glove_wiki_window10_size300_iteration50.tar.gz
[11]https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md

mechanism (Vaswani et al., 2017). This technique processes entire sequences of words at once, in contrast with the sequential processing of recurrent neural networks, retaining information about the position of each word and modifying the final word embeddings depending on the surrounding words, e.g., the context, both at training and at inference time. Moreover, contextualized language models based on transformer architectures allow for the pre-training and fine-tuning learning paradigm: first, the model is pre-trained on a general language learning task and a large training collection, and then the model is fine-tuned for the specific downstream task, e.g., classification. Currently, graph neural networks (Scarselli, Gori, Tsoi, Hagenbuchner, & Monfardini, 2009) are exploited in conjunction with BERT models for text classification tasks, but with marginally better results on some datasets (Lin et al., 2021).

Text classification Models Using the input texts represented as real-values vectors, supervised machine learning algorithms can be applied to classify new documents into different classes. This work focuses on Italian PQ documents, that need to be accurately and automatically classified according to 21 different major topics (see Table 1). The following classification models have been trained and compared:

- *support vector machine* (SVM): a support vector machine constructs a set of separating hyper-planes in the inputs' vector space, which can be used for classification among other tasks (Cortes & Vapnik, 1995);
- *complement naïve bayes* (CNB): it is a modified version of the multinomial naïve bayes classifier for texts, addressing word dependencies assumption and unbalanced classes (Rennie, Shih, Teevan, & Karger, 2003);
- *passive aggressive classifier* (PAC): an online machine learning algorithm building a set of separating hyper-planes in the inputs' vector space based on a single input at a time (Crammer, Dekel, Keshet, Shalev-Shwartz, & Singer, 2006);
- *multi-layer perceptron* (MLP): a artificial feedfoward neural network with one or mutiple hidden layers composed by different perceptrons that learns a non-linear function approximator for classification (Haykin, 1998);
- *convolutional neural network* (CNN): an artificial neural network with multiple hidden layers, i.e., a deep neural network, where the layers exploit convolving filters that are applied to local features (Kim, 2014);
- *long-short term memory network* (LSTM): an artificial recurrent neural network with multiple hidden layers and feedback connections that can process sequences of data, such as speech or text (Hochreiter & Schmidhuber, 1997).
- *transformer network* (BERT): a deep neural network with multiple attention layers, encoding documents terms as contextualized word embeddings, using a feed-forward layer at the top to perform document classification (Vaswani et al., 2017).

As discussed by Aggarwal (2018) and D'Andrea et al. (2019), the BOW text representation is usually adopted in combination with classical machine learning classification algorithms, such as SVM, CNB, PAC, and MLP, whereas the word embendings are used in combination with deep learning models, such as CNN and LSTM (Rosenthal, Farra, & Nakov, 2017). Hence, in the following experiments, we consider the following combinations of text classification models based on BOW and classical supervised machine learning models: BOW_SVM, BOW_CNB, BOW_PAC, and BOW_MLP. Regarding the deep learning based CNN and LSTM text classification models, the texts are represented as input vectors using pre-trained Word2Vec and FastText embeddings (we do not report experiments with GloVe word embedding due to poor results). Thus, we consider the following combinations of text classification models based on word embedding and deep learning models: Word2Vec_CNN, FastText_CNN, Word2Vec_LSTM, and FastText_LSTM. Finally, the BERT model processes directly the input document text, exploiting a custom tokenizer, a pre-trained word embedding (dbmdz/bert-base-italian-uncased)[12] and a maximum sentence length equal to 128. As regards the word embedding, as stated on the official repository page, a recent Italian Wikipedia dump and various texts, in Italian, from the OPUS corpora collection[13] were used for the pre-training stage. The training of the classification layer, namely the fine tuning stage, has been carried out, during each fold of the cross validation, considering the training set portion of the dataset introduced in Section 3. No domain-specific training stage has been carried out.

5. Experimental Comparison Campaign

In this Section, we discuss the results of the experimental comparison campaign that we carried out for identifying the most effective text classification models to embed into the QTIS Web application. First, in Sec. 5.1 we describe the experimental setup that we adopted. Then, in Sec. 5.2 we discuss the overall effectiveness metrics achieved by the different models along with a statistical comparison among them (Sec. 5.3). Finally, in Sec. 5.4 we discuss the recognition capability of per-class of each model and, for the sake of brevity, we focus on the least represented major topics in the dataset discussed in Section 3. Indeed, these major topics are the most difficult to recognize.

5.1. Experimental Setup

In order to compare the different text classification models discussed in Section 4, we carried out a 2×10-fold stratified cross-validation considering the entire dataset composed by 5,672 PQs discussed in Section 3. Indeed, we set two different values of the seed for the random generation function and we repeated the cross-validation procedure twice. In order to evaluate the performance of each classifier, we extracted for each model the following metrics: Precision, Recall, and F1-measure per class. Moreover we also extracted the overall accuracy and the weighted average values, considering all the classes, of Precision, Recall and F1-measure. Details on these metrics can be found in Tharwat (2018). After the experimental campaign, for each models and for each metric, we obtained a distribution composed by 20 values, corresponding to the 20 test sets of the cross-validation procedure. For each distribution, we also calculated the average value and the standard deviation. Moreover, in order to verify if there exist statistically significant differences among the distributions, we carried out non-parametric statistical tests, namely the Friedman test, the Iman and Davenport test, and, when needed, the Holm post-hoc procedure (García, Molina, Lozano, & Herrera, 2009). For the sake of brevity, in this work we show the test results only on the average values of the weighted average F1-measure. Actually, this measure allows us to summarize the overall performance level of a classification model (Tharwat, 2018), especially when dealing with imbalanced datasets.

In Table 2, we show, for each classification model, the parameters of the algorithms

[12]https://github.com/dbmdz/berts
[13]https://opus.nlpl.eu

Table 2.: Classification model parameters: Grid Search Ranges and Selected Values

Model	Tuned Parameters	Range Values	Selected Value
SVM	kernel	rbf, sigmoid, linear	linear
	γ	$-$, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}	$-$
	C	0.001, 0.10, 0.1, 10, 25, 50, 100, 1000, 10000	10000
CNB	α	0.3, 0.2, 0.1, 0.01, 0.001	0.2
PAC	C	10.0, 2.0, 1.0, 0.1, 0.01, 0.001	1.0
	tol	10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}	10^{-3}
MLP	hidden layer sizes	(128, 128), (192, 128)	(128, 128)
	activation	tanh, relu	relu
	optimizer	sgd, adam	adam
	α	10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}	10^{-3}
	learning rate	constant, adaptive	constant
CNN	conv1D layer size	64, 128, 192	128
	dense layer size	64, 128, 192	128
	activation	tanh, relu	relu
	optimizer	sgd, adam	adam
LSTM	LSTM layer sizes	64, 128, 192	192
	dense layer sizes	30, 50, 100	50
	activation	tanh, relu	relu
	optimizer	sgd, adam	adam
BERT	Batch size	16, 32	32
	Learning rate (Adam)	5e-5, 3e-5, 2e-5	2e-5
	Number of epochs	2, 3, 4	4

that we have optimized and used for the learning procedure. In order to identify the best parameter configurations, we carried out a grid search (Pontes, Amorim, Balestrassi, Paiva, & Ferreira, 2016; Syarif, Prugel-Bennett, & Wills, 2016), and in Table 2 we also show the values of the ranges used during the searching process. As regards the models based on BOW and classical supervised machine learning models, we carried out the experiments using the scikit-learn[14] Python machine learning library, whereas for the model based on deep-learning, we adopted the Keras[15] Python library for deep learning. For more details on the parameters, please check the official documentation of the aforementioned libraries. As regards the feature selection stage, we set F_{max} as the maximum number of features, namely the number of terms of the vocabulary. This value was used as a threshold when we carried out the experiments considering also a feature selection procedure stage, after the text representation stage and before training the classifier. We experimented different values of F_{max}, spanning from $10,000$ to $50,000$. The best results were achieved, on average, considering F_{max} equal to $20,000$. In the following, the models generated considering the feature selection stage will be referred with the _20000 suffix.

[14]http://scikit-learn.org
[15]https://keras.io/

5.2. Overall Performance Analysis

Table 3 summarizes the results achieved by each text classification model. Specifically, for each model, we show the average value (avg), the standard deviation (std dev) and the maximum value (max) (over the 20 test sets of the cross validation) of the accuracy and of the weighted average Precision, Recall and F1-measure. More specifically, when we calculated the average values, we weighted Precision, Recall and F1-measure of each class by the number of samples from that class.

In order to better visualize the comparison among the different classification models, we show in Figure 2, a bar plot of the average values of each metric, which include also a dash for the standard deviation.

Table 3.: Overall results of the cross-validation procedure. The weighted average values of Precision, Recall and F1-measure are reported in %.

Model	Accuracy		Precision		Recall		F1-measure	
	avg (std dev)	max	avg (std dev)	max	avg (std dev)	max	avg (std dev)	max
BOW_SVM	70.6 (±1.5)	73.1	70.8 (±1.5)	73.8	70.6 (±1.5)	73.1	70.1 (±1.5)	72.7
BOW_CNB	72.3 (±1.5)	74.9	72.5 (±1.6)	75.3	72.3 (±1.5)	74.9	71.0 (±1.6)	73.7
BOW_PAC	71.7 (±1.5)	73.8	71.6 (±1.6)	74.3	71.7 (±1.5)	73.8	71.1 (±1.5)	73.5
BOW_MLP	71.7 (±1.3)	73.9	72.2 (±1.4)	75.2	71.7 (±1.3)	73.9	71.2 (±1.3)	73.7
BOW_SVM_20000	62.4 (±1.8)	65.4	63.0 (±1.8)	66.1	62.4 (±1.8)	65.4	62.1 (±1.8)	65.2
BOW_CNB_20000	71.6 (±1.5)	74.6	71.8 (±1.7)	75.3	71.6 (±1.5)	74.6	70.2 (±1.7)	73.3
BOW_PAC_20000	66.3 (±1.7)	69.3	66.7 (±1.8)	70.4	66.3 (±1.7)	69.3	65.8 (±1.7)	69.0
BOW_MLP_20000	70.4 (±1.1)	72.1	70.8 (±1.1)	72.6	70.4 (±1.1)	72.1	69.8 (±1.1)	71.3
Word2Vec_CNN	63.6 (±2.8)	69.0	65.6 (±1.7)	69.0	63.6 (±2.8)	69.0	63.0 (±2.7)	68.3
FastText_CNN	63.5 (±2.4)	68.7	65.2 (±2.3)	71.6	63.5 (±2.4)	68.7	62.9 (±2.7)	68.8
Word2Vec_LSTM	65.8 (±2.9)	69.5	64.5 (±2.4)	66.8	65.8 (±2.9)	69.5	64.2 (±2.6)	67.4
FastText_LSTM	66.2 (±2.1)	69.2	65.8 (±1.8)	68.0	66.2 (±2.1)	69.2	64.7 (±2.1)	67.1
BERT	69.5 (±2.3)	73.8	70.2 (±2.4)	73.3	69.5 (±2.3)	73.8	68.5 (±2.4)	72.9

As a general overview, considering all the metrics, the method based on BOW text representation and classical machine learning classification algorithms, namely SVM, CNB, PAC and MLP, achieved higher performance than text classification models based on deep learning. With regard to the latter, in most cases, the average values of accuracy and the weighted average values of Precision, Recall and weighted F1-measure are lower than 65%, whereas the comparison models achieve values of the performance metrics higher than 70%. Also the standard deviations associated to deep learning-based text classification models are higher than the ones of the comparison models. Thus, we can state that the learning procedures of the models based on BOW and classical machine learning classifiers are more stable than the ones of the deep learning-based models. However, it is worth noticing that the results achieved by BERT are much closer to those achieved by the models based on BOW and classical machine learning algorithms, than those achieved by other deep learning models. Regarding these results, we can state that they are in line with those achieved in the specialized literature (Burscher et al., 2015; Hillard et al., 2008; Loftis & Mortensen, 2020; Navarretta & Hansen, 2020), although the latter were not obtained using the same datasets and experimental setup of those considered in our work.

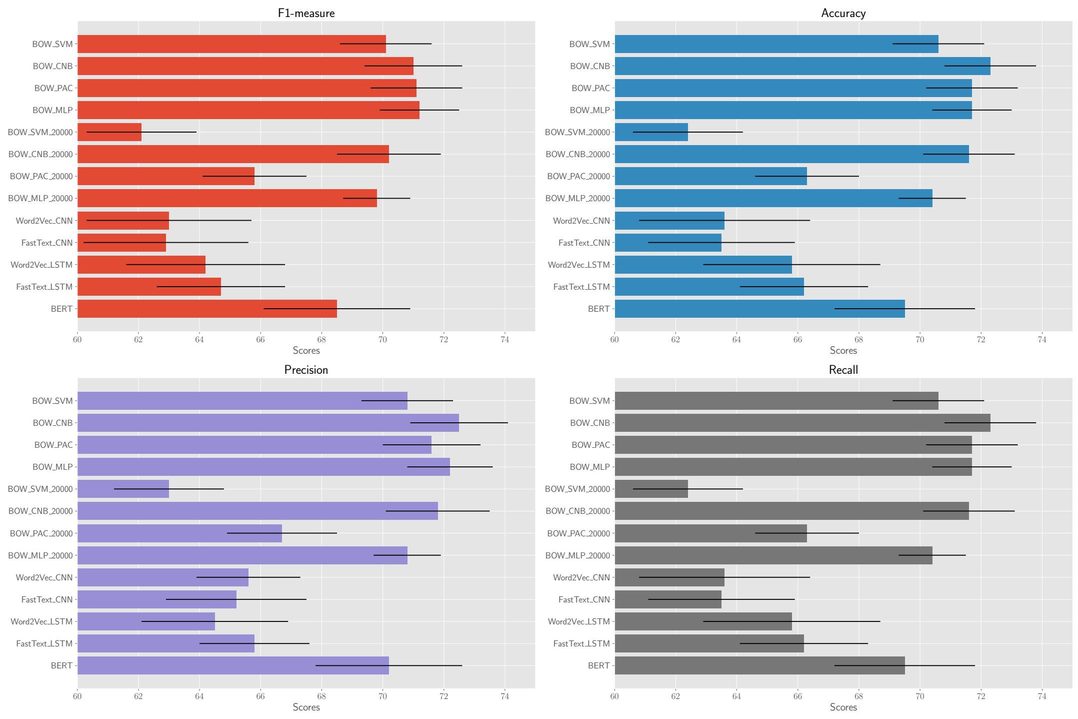

Figure 2.: Overall performance scores of the text classification models (reported in %).

5.3. Statistical Comparison of the Model Performance

In the following, we also discuss the results of non-parametric statistical tests. First, we divide the distributions of the weighted F1-measure metric, achieved on each test set of the cross-validation procedure by each model, into three groups: i) the ones of the models based on BOW and classical machine learning classifiers, without feature selection stage; ii) the ones of the the models based on BOW and classical machine learning classifiers, with feature selection stage; and iii) the ones of the deep learning-based models. We refer to these three distributions as Distribution Group 1 (DG1), Distribution Group 2 (DG2) and Distribution Group 3 (DG3), respectively. Then, for each distribution group, we perform the Friedman test to compute a ranking among the distributions, and the Iman and Davenport test to evaluate whether there exists a statistically significant difference among the distributions. If the Iman and Davenport p-value is lower than the level of significance α (it is assumed the standard threshold value $\alpha = 0.05$), we can reject the null hypothesis of equivalence and conclude that there exist statistical differences among the multiple distributions. Otherwise, no statistically significant difference exists. In case of a statistically significant difference, we apply a post-hoc procedure, namely the Holm test. This test allows detecting effective statistically significant differences between the *control approach*, i.e., the one with the lowest Friedman rank, and the remaining approaches. Details on the aforementioned tests may be found in García et al. (2009).

Table 4 shows the results of the Friedman test (F-Rank, in bold the models with the best rank) and Iman and Davenport test's p-values, obtained considering the aforementioned distribution groups DG1, DG2 and DG3. All the tests highlighted statistically significant differences among the models, thus we carried out the Holm post hoc procedures, considering, respectively, BOW_PAC BOW_CNB_20000 and BERT as control models.

Table 5 shows the results of the Holm post hoc procedures on DG1, DG2 and DG3. We recall that, for this procedure, when comparing the control model against each other model, the level of significance is scaled as α/i, where i is the position of the specific comparison model in the Friedman ranking. As regards DG2, the statistical hypothesis of equivalence is rejected only for BOW_SVM, thus we can state that the remaining models which generate the distribution in DG1 are statistically equivalent. As regards DG1, only BOW_MLP_20000 is statistically equivalent to the control model, namely to BOW_CNB_20000. Thus, we can state that the two other models performs statistically worse than BOW_CNB_20000. Finally, regarding DG2 the statistical hypothesis of equivalence is rejected for all the comparison models, thus we can state that BERTis the best model among those based on deep learning.

Table 4.: Results of the Friedman (F-Rank) and of the Iman and Davenport (p-value) tests on the weighted average values of F1-measure computed on DG1, DG2 and DG3. The best model per DG is denoted in **bold**.

Group	Model	F-Rank	p-value	Null hypotesis
DG1	BOW_SVM	3.23	0.017	reject
	BOW_CNB	2.48		
	BOW_PAC	2.00		
	BOW_MLP	2.30		
DG2	BOW_SVM_20000	4.00	0.000	reject
	BOW_CNB_20000	1.38		
	BOW_PAC_20000	3.00		
	BOW_MLP_20000	1.63		
DG3	Word2Vec_CNN	2.80	0.000	reject
	FastText_CNN	3.80		
	Word2Vec_LSTM	3.20		
	FastText_LSTM	3.03		
	BERT	1.30		

In order to carry out a more in-depth analysis, we considered another distribution group, labeled as DG4, composed by the distribution of DG1 and DG2, except for the distribution generated by models build using BOW and SVM, with and without feature selection stage. Indeed, the models based on SVM achieved the worst ranks both in DG1 and DG2. Table 6 shows the results of the Friedman test (F-Rank, in bold the model with the best rank) and the Iman and Davenport test's p-value carried out considering DG4. Since the null hypothesis of statistical equivalence has been rejected, we carried out also the Holm post hoc procedure, considering BOW_PAC as control model. As shown in the Table 7, the null hypothesis of statistical equivalence has been rejected for BOW_PAC_20000 and BOW_MLP_20000, whereas the other models of DG4 result to be statistically equivalent to BOW_PAC. Among them, only BOW_CNB_20000 is the one generated considering a reduced vocabulary composed by 20,000 terms. For all the remaining models, the feature selection stage led to a decrease of the performance, in

Table 5.: Results of the Holm post hoc procedures on the weight-averaged F1-measure computed on DG1, DG2 and DG3.

Group	i	Model	p-value	α/i	Null hypothesis
DG1	3	BOW_SVM	0.003	0.017	reject
	2	BOW_CNB	0.245	0.025	not reject
	1	BOW_MLP	0.462	0.050	not reject
DG2	3	BOW_SVM_20000	0.000	0.017	reject
	2	BOW_PAC_20000	0.001	0.025	reject
	1	BOW_MLP_20000	0.540	0.050	not reject
DG3	4	Word2Vec_CNN	0.0000	0.013	reject
	3	FastText_CNN	0.0000	0.017	reject
	2	Word2Vec_LSTM	0.0001	0.025	reject
	1	FastText_LSTM	0.0005	0.050	reject

terms of F1-measure, statistically confirmed by the tests.

Finally, we conducted the statistical tests on a distribution group, labeled as DG5, composed by the distributions of weighted average values of F1-measure generated by the best models in DG4 and the best model in DG5,namely BERT. Also the results of these tests are shown in Tables 6 and 7. The tests demonstrate that, even though BERT is the best method among those based on deep learning, it statistically performs worse than the best model based on BOW for text representation and classical machine learning classifiers, namely BOW_PAC. As expected, the remaining models of DG5 are statistically equivalent to BOW_PAC.

Table 6.: Results of the Friedman (F-Rank) and of the Iman and Davenport (p-value) tests on the weighted average values of F1-measure computed on DG4 and DG5. The best model per DG is denoted in **bold**.

Group	Model	F-Rank	p-value	Null hypotesis
DG4	BOW_CNB	2.40	0.000	reject
	BOW_PAC	2.38		
	BOW_MLP	2.45		
	BOW_CNB_20000	3.63		
	BOW_PAC_20000	6.00		
	BOW_MLP_20000	4.15		
DG5	BOW_CNB	2.45	0.000	reject
	BOW_PAC	2.35		
	BOW_MLP	2.45		
	BOW_CNB_20000	3.60		
	BERT	4.15		

Table 7.: Results of the Holm post hoc procedures on the weight-averaged F1-measure computed on DG4 and DG5.

Group	i	Model	p-value	α/i	Null hypothesis
DG4	5	BOW_PAC_20000	0.000	0.010	reject
	4	BOW_MLP_20000	0.003	0.013	reject
	3	BOW_CNB_20000	0.035	0.017	not reject
	2	BOW_MLP	0.899	0.025	not reject
	1	BOW_CNB	0.966	0.050	not reject
DG5	4	BERT	0.0003	0.013	reject
	3	BOW_CNB_20000	0.0124	0.017	reject
	2	BOW_MLP	0.8410	0.025	not reject
	1	BOW_CNB	0.8410	0.050	not reject

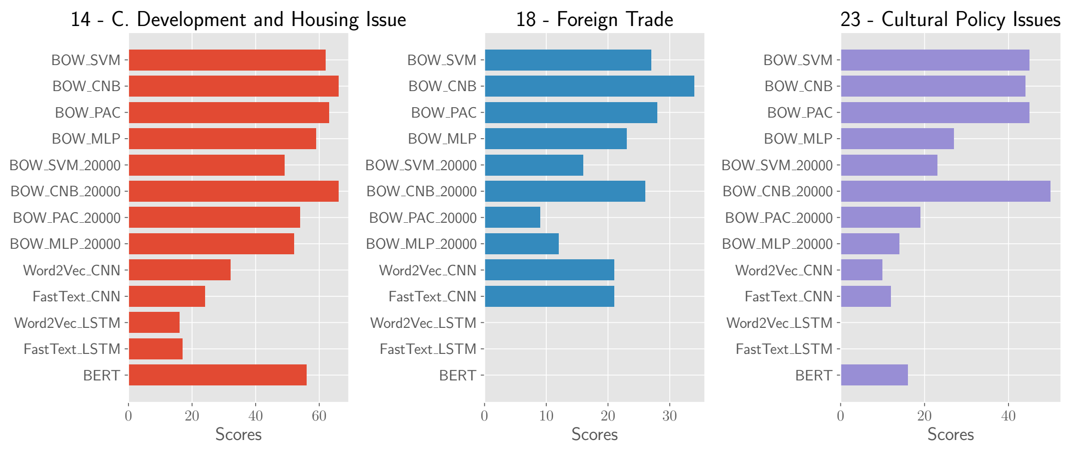

Figure 3.: Average values of the F1-measures for the 3 major topics less represented (reported in %).

5.4. Per-Class Performance Analysis

For the sake of brevity, in the following we discuss the capability of each model of recognizing the three minority classes, namely major topic 14 (C. Development and Housing Issue), major topic 18 (Foreign Trade), and major topic 23 (Cultural Policy Issues). However, on a web repository[16], we show for each major topic the average values of the Precision, Recall and F1-measure associated to each text classification model.

In Figure 3, we show for the three minority classes the average values of the F1-measure associated to each text classification model.

The plots highlight that the models based on LSTM cannot recognize at all documents belonging to major topic 18 and major topic 23. However, the classification models based on CNN manage to recognize some of the documents belonging to major topic 18 and major topic 23, even if with average values of F1-measure lower than

[16]`https://bit.ly/39HP2ss`

the models based on BOW representation and classical machine learning classifiers. Finally, it is worth to notice that BERT is not able recognize the document belonging to major topic 18 whereas its level of F1 measure is very close to that of the models based on BOW representation and classical machine learning classifiers. As regards the documents belonging to major topic 23, they are scarcely recognized by BERT.

Overall, in most of cases, the values of F1-measure associated to deep learning-based models are lower than the ones associated with the comparison models. As regards the models based on BOW representation and classical machine learning classifiers, all but BOW_SVM_20000, achieve an F1-measure higher than 50% on major topic 14, where the best F1-measure, equal to 66%, is achieved to BOW_CNB and BOW_CNB_20000. On major topic 18 the best value of F1-measure, equal to 34%, is achieved by BOW_CNB. Finally, on major topic 23 the best F1-measure, equal to 48%, is achieved by BOW_CNB_20000. In conclusion, we can state that all models still have difficulties in recognizing major topic 18, which is the less represented major topics. However, major topic 14 is almost well recognized by most of the models based on BOW representation and classical machine learning classifiers. Finally, major topic 23 is enough well recognized by only BOW_CNB_20000.

On the basis of the analysis carried out in this Section, the BOW_CNB_20000 text classification model achieves the highest values of the average F1-measure for the two of the three minority major topics in the dataset and, in terms of weighted average F1-measure, it is statistically equivalent to the best performing model, namely BOW_PAC. Thus, we decided to adopt it as default text classification method in our QTIS Web application. For the sake of completeness, we show in Table 8 the average values of Precision, Recall and F1-measure, per major topic, associated to the BOW_CNB_20000.

Table 8.: Per-class average scores of BOW_CNB_20000. Precision, Recall and F1-measure are reported in %

Code	**Major Topics**	**CV Scores for Classes**		
		Precision	**Recall**	**F1-measure**
1	Domestic Microeconomic Issues	58.4 (±6.1)	66.0 (±6.2)	61.9 (±5.9)
2	Civil Right, Minority Issues, and Civil Liberties	67.4 (±17.4)	27.7 (±10.7)	38.4 (±11.6)
3	Health	83.3 (±4.8)	90.8 (±4.5)	86.8 (±4.0)
4	Agriculture	77.9 (±6.6)	88.8 (±7.7)	82.9 (±6.3)
5	Labour and Employment	65.5 (±7.2)	65.2 (±6.8)	65.2 (±6.2)
6	Education	72.9 (±6.3)	95.8 (±3.4)	82.6 (±4.1)
7	Environment	68.6 (±4.0)	86.9 (±7.3)	76.5 (±4.0)
8	Energy	73.7 (±11.7)	78.1 (±16.5)	75.5 (±13.3)
9	Immigration	75.0 (±6.8)	79.6 (±9.7)	77.0 (±6.9)
10	Transportation	79.0 (±4.3)	92.2 (±3.2)	85.0 (±3.0)
12	Low and Crime	71.5 (±3.9)	87.1 (±4.0)	78.5 (±2.1)
13	Welfare	83.3 (±16.1)	33.6 (±15.6)	45.5 (±16.7)
14	C. Development and Housing Issue	83.8 (±14.3)	58.6 (±20.6)	66.4 (±17.3)
15	Banking, Finance, and Domestic Commerce	68.7 (±7.4)	48.5 (±5.9)	56.7 (±5.5)
16	Defence	76.0 (±15.4)	52.0 (±10.9)	61.0 (±10.5)
17	Space, Science, Technology, and Communications	73.2 (±11.4)	64.5 (±13.1)	67.9 (±10.4)
18	Foreign Trade	50.0 (±48.7)	17.9 (±16.7)	26.0 (±24.4)
19	International Affairs	68.7 (±8.9)	62.7 (±8.1)	65.2 (±6.8)
20	Government Operations	67.9 (±7.8)	51.0 (±8.8)	58.0 (±7.7)
21	Public Lands and Water Management	71.4 (±14.2)	53.1 (±13.2)	59.8 (±11.6)
23	Cultural Policy Issues	78.3 (±40.9)	38.8 (±27.9)	49.6 (±30.6)

6. The QTIS Web application

In this Section we discuss the design, implementation and testing of the QTIS Web application used to classify new PQs into CAP topics. This application is designed and implemented as a *microservices* architecture (Sec. 6.1) and includes all the text classification models based on BOWrepresentation and classical machine learning classifiers. Then, we discuss the performance of the default text classification models, namely BOW_CNB_20000, on a previously unseen dataset (Sec. 6.2).

6.1. Design and Implementation of the QTIS Web application

In this Section, we argue about the design and implementation stage that we carried out for developing and QTIS Web application[17]. First, we discuss the main functional requirements that we have identified. Then, we discuss the application architecture and the main technologies that we adopted to implement it.

6.1.0.1. Application requirements. We identified two main actors that can interact with the QTIS Web application, namely the *CAP Researcher* (CAPR) and the *Data Scientist* (DS) users. Each user has a different scope. The CAPR is allowed to upload a set of PQs and to ask to the application to associate to each PQ a code, which describes a specific major topic, as shown in Tab. 1. The DS is allowed to create and train new text classification models. After training, the new model will be embedded into the application and may be selected by the CAPR to classify each text of PQ into a specific major topic. Moreover, the DS user also acts as system administrator.

In details, the main functionalities offered to the CAPR user are:

- to upload a set of unlabeled PQs;
- to select a text classification model, from a list of pre-loaded and pre-trained models;
- for each uploaded PQ, to estimate the code which describes a specific major topic;
- to generate a report which resumes the text classification outcomes. Specifically, a table is generated containing a record for each uploaded PQ text. Each record includes the PQ text and the corresponding estimated code;
- to export the report.

The main functionalities offered to the DS user are:

- to upload a set of labeled PQs, containing, for each PQ, the actual code associated to a specific major topic;
- to train a new text classification model, using the uploaded set of labeled PQs;
- to tune the parameters to train a new text classification model, which include the text representation and the specific algorithm for learning the classification model;
- to evaluate the performance of the trained classification model, using different metrics (see Sec. 5.2 for more details);
- to add the trained classification model to the list of models available for being used by the CAPR user.

[17]The QTIS web application will be publicly available upon acceptance.

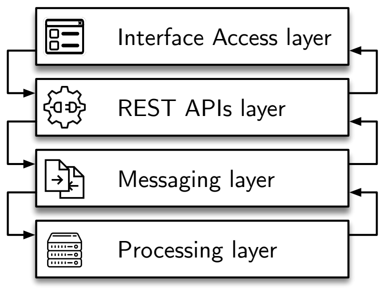

Figure 4.: Multi-tier Architecture of the QTIS Web application.

6.1.0.2. Application architecture. The QTIS Web application has been designed and developed as an event-driven infrastructure, built on REST microservices (Neumann, Laranjeiro, & Bernardino, 2018). This architectural style has been chosen because of the following advantages:

- client's access to RESTful API services is independent of client type and encoding, e.g., web application, mobile app, command line tools, etc.;
- loose coupling of the presentation layer and processing layer on the server;
- decomposition of the processing layer into several loosely-coupled and independently deployable services;
- better fault isolation w.r.t. classical tree-tiers applications
- high scalability of the different services.

Figure 4 shows the multi-tier architecture that we adopted for developing a prototype of the QTIS Web application.

In the following we briefly describe each layer:

- *Interface Access* layer: represents the users interface for accessing REST services. Two main services are offered to the web clients, namely i) the service for estimating the major topics of a collection of PQ documents (CAPR user) and ii) the service for training a new text categorization model, exploiting a collection of labeled PQ documents (DS user).
- *REST APIs* layer: is in charge of publishing the API REST endpoints to submit the requests to the processing layer.
- *Messaging* layer: stores both the requests coming from the Interface Access layer through the REST APIs layer and the replies to the requests provided by the underlying processing layer. It decouples the exchange of data between the web client, e.g., the REST APIs layer and the processing, through an in-memory data structure storage, used as a message broker.
- *Processing* layer: is in charge of actually executing the tasks for providing answers to the requests, namely to classify the PQ documents or to train a new text classification model.

For the implementation of the QTIS Web application we exploited a set of state-of-the-art IT technologies. We mainly exploited the Python ecosystem, including Flask[18] for implementing the Interface Access (along with HTML5, CSS and Javascript) and the REST API layers. The scikit-learn machine learning Python library was adopted for developing the Processing layer. Finally, Redis[19], an open source (BSD licensed) and in-memory data structure store, was used as cache and message broker.

[18]`http://flask.palletsprojects.com`
[19]`http://redis.io/`

6.2. Testing of the QTIS Web application

The QTIS Web application, together with the trained BOW_CNB_20000 text classification model, has been tested on a previously unseen collection of PQs. This test dataset includes 222 observations collected from 1 July 2019 to 8 May 2020. The target classes have been manually identified, and their distribution among the 21 major topics are reported in Table 9 (column Count). The different major topics are strongly unbalanced: as in the training dataset, the three most numerous major topics (Law and Crime, Government Operations, and Transportation) account for more than 40% of the documents.

As reported in Table 9, the performance of the BOW_CNB_20000 model on the test dataset, namely 74% in Precision, 73% in Recall, and 71% in F1-measure, are close to the corresponding average performance of the same model in the cross-validation analysis, reported in Table 3. Moreover, the overall accuracy, namely the percentage of correctly recognized PQs, is equal to 73%.

With respect to the single major topics, the BOW_CNB_20000 model achieved completely satisfactory results (all metrics equal to 100%) for the major topics 6 (Education) and 17 (Space, Science, Technology, and Communication) and it is extremely reliable (F1-measure $\geq$ 80%) also for the major topics 3 (Health), 8 (Energy), 9 (Immigration), 10 (Transportation) and 12 (Law and Crime). By contrast, the least adequate results (F1-measure $\leq$ 50%) were achieved for the the major topics 14 (C. Development and Housing Issue), 15 (Banking, Finance, and Domestic Commerce), 18 (Foreign Trade), 21 (Public Lands) and 23 (Cultural Policy Issues). These results are broadly consistent with those achieved by the BOW_CNB_20000 model in the experimental campaign (see Table 8). In general there is a positive relation between the popularity of a the major topic and the performance of the application on the test set: with respect to the three most popular the major topics, the performance scores are higher than overall weighted average performance with the exception of the class 20 (Government Operations), whose F1-measure is limited by a Recall score as low as 45%. Figure 5 shows that the BOW_CNB_20000 model is especially prone to commit a few types of mistakes. There are 5 instances in which questions falling within the major topic 15 (Banking, Finance and Domestic Commerce) have been predicted as pertaining to the major topic 1 (Domestic Macroeconomic Issues). The opposite situation, i.e. questions falling in the major topic 1 but predicted in the major topic 15, is still quite common, having occurred in three cases. The difficulty to discriminate between these two major topics is not surprising because both major topics concern economic issues, although framed in a different perspective: however it is worth noticing that CAP scholars sometimes collapse the two major topics in a single “economic” dimension (Borghetto & Russo, 2018), containing the severity of this problem. Finally, despite the acceptable level of recognition achieve by BOW_CNB_20000 in cross-validation (average F1-measure for class 14 = 66%), class 14 (C. Development and Housing Issue), represented in the test set by 4 PQs, is never correctly recognized and confused with class 1 (Domestic Macroeconomics Issues), 7 (Environment), 10 (Transportation) and 20 (Government Operation), respectively. It is worth noting that all those questions concerned rather new policy instruments to foster the economic development of disadvantaged geographical areas, namely the ”National Strategy for Inland Areas”, the ”National plan for the social and cultural regeneration of degraded urban areas” and the ”Institutional Contracts of Development”. The text of the questions included several references to transport infrastructures, environmental or economic issues and the capacity of the government to manage European funds which, together with the scarcity of references to these policy instruments in the training dataset, has probably reduced the capacity of the application to correctly label these questions.

Table 9.: Per-class PQs count and performance evaluation of the BOW_CNB_20000 text classifier on the test dataset. Precision, Recall and F1-measure are reported in %.

Code	**Major Topic**	**Precision**	**Recall**	**F1-measure**	**Count**
1	Domestic Microeconomic Issues	57.9	64.7	61.1	17
2	Civil Right, Minority Issues, and Civil Liberties	100.0	50.0	66.7	4
3	Health	78.9	100.0	88.2	15
4	Agriculture	71.40	83.3	76.9	6
5	Labour and Employment	57.1	100.0	72.7	4
6	Education	100.0	100.0	100.0	13
7	Environment	62.5	71.4	66.7	14
8	Energy	66.7	100.0	80.0	2
9	Immigration	72.7	100.0	84.2	8
10	Transportation	76.9	90.9	83.3	33
12	Low and Crime	82.8	96.0	88.9	25
13	Welfare	100.0	40.0	57.1	5
14	C. Development and Housing Issue	0.0	0.0	0.0	4
15	Banking, Finance, and Domestic Commerce	46.7	46.7	46.7	15
16	Defence	100.0	50.0	66.7	2
17	Space, Science, Technology, and Communications	100.0	100.0	100.0	1
18	Foreign Trade	100.0	25.0	40.0	4
19	International Affairs	76.9	71.4	74.1	14
20	Government Operations	73.7	45.2	56.0	31
21	Public Lands and Water Management	25.0	100.0	40.0	1
23	Cultural Policy Issues	100.0	25.0	40.0	4
	Overall (weighted average)	74.0	73.0	71.0	222

7. Conclusions and future work

In this work, we have presented an Intelligent System, based on text classification models, for the automatic categorization of political documents. Specifically, we have focused our attention on documents containing the Parliamentary Questions, for oral reply, collected during the weekly Question Time at the Chamber of Deputies of the Italian Republic. We have described the design and the implementation of a web application, that we named QTIS, developed exploiting some of the most recent IT software architecture and technologies, such as Rest microservices and libraries of the Python ecosystem.

In order to select the most effective text classification models to embed as engine of the proposed Intelligent Systems, we have carried our an intensive experimental comparison campaign. Indeed, we have compared the performance achieved, in terms of accuracy, Precision, Recall and F1-measure, of 12 text classification models. Specifically, we have considered 4 models based on BOW for text representation and classical machine learning classifiers. Moreover, we considered also the versions of the aforementioned 5 models that exploits a reduced version of the terms vocabulary, namely that embeds a feature selection stage in the training process. Finally, we also experimented 4 methods based on consolidated deep-learning techniques, namely based on WE for text representation and deep neural networks as classifiers (CNN, LSTM and transformer-based models).

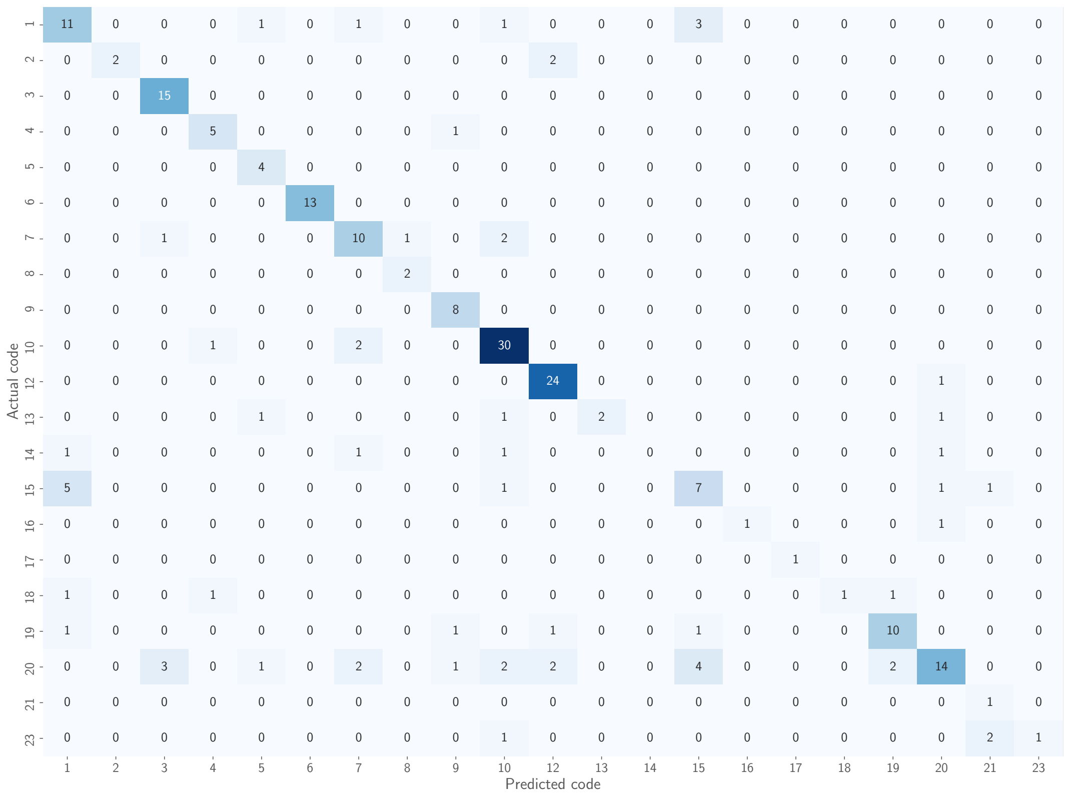

Figure 5.: Confusion matrix of the BOW_CNB_20000 text classifier on the test dataset

Results have shown that text classification models based on deep-learning techniques have achieved the worst text classification performance. These results confirmed that deep-learning models are not suitable for approaching problems similar to ones discussed in this work, in which the dimension of the training set is limited to a few thousands of documents. In such a situation it is impossible to estimate all the parameters of the deep neural networks. Within the field of political science recursive neural networks have been successfully applied, albeit to a different type of problem (Iyyer et al., 2014)[20] Despite their limited performance in this occasion, classifiers based on deep-learning have been also integrated in the QTIS Web application for their increasing popularity and for giving the possibility to the user to re-train them if larger datasets will be available. As regards the other text classification models, we have statistically demonstrated that models based on SVM are worse than the ones based on CNB, PAC and MLP. Moreover, the adoption of a reduced set of terms in the vocabulary have led to a statistically appreciable reduction of the performance of the text classification models, except for the model based on Bayesian Classifiers, namely BOW_CNB_20000.

Although we have integrated in the proposed QTIS Web application all the 9 models based on BOW for text representation and classical machine learning classifiers, we select the BOW_CNB_20000 as default models. Adopting this model, we have experimented the QTIS Web application for the categorization of a collection of previously unseen PQs documents, spanning from July 2019 to May 2020. Political scientists, which have manually analyzed the results of the automatic categorization, have confirmed that the models correctly classified up to 73% of the documents. However, it is worth noting that some major topics can suffer from being correctly recognized, due to the fact that they are underrepresented in the dataset that we adopted for training the text classification models

As the QTIS Web application has produced satisfactory results, the next step will be to utilize them to update the QT dataset in preparation for its next official release. With regard to this purpose, future works will benefit from the cooperation between human coders and the QTIS Web application to achieve increasingly reliable results. To begin with, the new 222 questions that have been manually labelled to evaluate the performance of the BOW_CNB_20000 text classifier will be added to the original dataset. At the end of the current legislative term, which is planned for March 2023 except in case of early elections, we expect to have about 800 new questions to analyze. We plan to collect and label them with the support of QTIS Web application, which will be re-trained for the occasion with a new training dataset including the 222 questions used here as test dataset.

In future research activities, we envision to test the capability of the QTIS Web application, trained with QT questions, to correctly predict policy issues in a different type of political documents, namely PQs for written reply. Within the Italian case, parliamentary questions for written reply have been analyzed so far only with a dictionary-based approach because they are too numerous to be manually labelled (in an average 5 years legislative term about 40,000 of such questions are asked in the Italian Chamber of Deputies). Even though previous attempts to label political texts with applications trained with different data yielded unsatisfactory results, the similarity between the structure and content of questions for written reply and QT questions makes this experiment worth trying. Moreover, we plan to train innovative

[20]It is worth specifying that the study conducted by Iyyer et al. (2014) applied recursive neural network framework to identify political ideology instead of analysing political documents.

word embedding representations for the italian language, such as regularized word embeddings (Novotný, Ayetiran, Štefánik, & Sojka, 2020), to study their classification performance on our models, despite their reported slowdowns at inference time. We also plan to exploit graph neural networks on top of BERT models as in (Lin et al., 2021) to assess the effectiveness improvements and the efficiency costs of such solutions.

As a final discussion item, we argue that, even though we have designed and implemented QTIS Web application for the categorization of the Italian QT, and therefore we have focused our experimental campaign using such documents, the application can be used also for carrying out a new text categorization analysis considering other political documents, such as additional types of parliamentary questions (PQs for written reply, PQs in committees, interpellations) and a wide range of different texts produced in parliament (bills, laws, investiture speeches, debates). This is a substantial improvement, as political scientists have the possibility to train and integrate new text classification models, providing any type of labeled texts as training set. However, the current version of QTIS Web application does not allow to train text classification models for texts written not in Italian. This is due to the fact that data text pre-processing steps have been implemented using tools for the Italian language. Further future extensions of QTIS Web application may include the support of multiple languages

Acknowledgments

The authors of this article wish to thank Linda Basile, Enrico Borghetto and Francesco Visconti for their help in manually checking the output of the QTIS Web application generated using the 222 unlabelled PQs as input.

Declaration on interest statement

The authors declare there are no conflicts of interest.

Funding

This work is partially supported by the Italian Ministry of Education and Research (MIUR) under the CrossLab project (Departments of Excellence); and the University of Pisa under the AUTENS project (Sustainable Energy Autarky).

Notes on contributors

Alice Cavalieri is research fellow at the University of Torino, Italy
Pietro Ducange is associate professor at the Department of Information Engineering of the University of Pisa, Italy.
Silvio Fabi is software engineer at NBS srl, San Benedetto del Tronto, Ascoli Piceno, Italy
Federico Russo is associate professor at the University of Salento, Lecce, Italy

Nicola Tonellotto is assistant professor at the Department of Information Engineering of the University of Pisa.

Data availability statement

A copy of the dataset can be accessed here: [disclosed upon acceptance].

References

Aggarwal, C. C. (2018). *Machine learning for text.* Springer.

Aggarwal, C. C., & Zhai, C. (2012). *Mining text data.* Springer Science & Business Media.

Baumgartner, F. R., Breunig, C., & Grossman, E. (2019). *Comparative policy agendas: Theory, tools, data.* Oxford University Press.

Baumgartner, F. R., Jones, B. D., & Mortensen, P. B. (2017). Punctuated equilibrium theory: Explaining stability and change in public policymaking. In *Theories of the policy process* (4th ed., pp. 55–101). Westview Press Boulder, CO.

Benoit, K., Watanabe, K., Wang, H., Nulty, P., Obeng, A., Müller, S., & Matsuo, A. (2018). quanteda: An R package for the quantitative analysis of textual data. *Journal of Open Source Software*, *3*(30), 774.

Bischl, B., Lang, M., Kotthoff, L., Schiffner, J., Richter, J., Studerus, E., … Jones, Z. M. (2016). mlr: Machine learning in r. *The Journal of Machine Learning Research*, *17*(1), 5938–5942.

Borghetto, E., & Chaques-Bonafont, L. (2019). Parliamentary questions. In *Comparative policy agendas* (pp. 282–299). Oxford University Press Oxford, New York.

Borghetto, E., & Russo, F. (2018). From agenda setters to agenda takers? the determinants of party issue attention in times of crisis. *Party Politics*, *24*(1), 65–77.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., … others (2020). Language models are few-shot learners.

Burdisso, S. G., Errecalde, M., & Montes-y Gómez, M. (2019). A text classification framework for simple and effective early depression detection over social media streams. *Expert Systems with Applications*, *133*, 182–197.

Burscher, B., Vliegenthart, R., & De Vreese, C. H. (2015). Using supervised machine learning to code policy issues: Can classifiers generalize across contexts? *The ANNALS of the American Academy of Political and Social Science*, *659*(1), 122–131.

Collingwood, L., & Wilkerson, J. (2012). Tradeoffs in accuracy and efficiency in supervised learning methods. *Journal of Information Technology & Politics*, *9*(3), 298–318.

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

Courtney, M., Breen, M., McMenamin, I., & McNulty, G. (2020). Automatic translation, context, and supervised learning in comparative politics. *Journal of Information Technology & Politics*, *17*(3), 208–217.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., & Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, *7*(Mar), 551–585.

D'Andrea, E., Ducange, P., Bechini, A., Renda, A., & Marcelloni, F. (2019). Monitoring the public opinion about the vaccination topic from tweets analysis. *Expert Systems with Applications*, *116*, 209–226.

D'Andrea, E., Ducange, P., Lazzerini, B., & Marcelloni, F. (2015). Real-time detection of traffic from twitter stream analysis. *IEEE transactions on intelligent transportation systems*, *16*(4), 2269–2283.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019, June). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of the 2019 conf. of* *the north American chapter of the association for computational linguistics: Human language technologies, vol 1 (long and short papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics.

García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization. *Journal of Heuristics*, *15*(6), 617–644.

Gilardi, F., & Wüest, B. (2020). Using text-as-data methods in comparative policy analysis. In *Handbook of research methods and applications in comparative policy analysis.* Edward Elgar Publishing.

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., & Mikolov, T. (2018). Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893*.

Grimmer, J. (2010). A bayesian hierarchical topic model for political texts: Measuring expressed agendas in senate press releases. *Political Analysis*, *18*(1), 1–35.

Grimmer, J., & King, G. (2011). General purpose computer-assisted clustering and conceptualization. *Proceedings of the National Academy of Sciences*, *108*(7), 2643–2650.

Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political analysis*, *21*(3), 267–297.

Hansen, D. H., Navarretta, C., Offersgaard, L., & Wedekind, J. (2019). Towards the automatic classification of speech subjects in the danish parliament corpus. In *Dhn* (pp. 166–174).

Haykin, S. (1998). *Neural networks: A comprehensive foundation* (2nd ed.). Prentice Hall PTR.

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.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hughes, M., Li, I., Kotoulas, S., & Suzumura, T. (2017). Medical text classification using convolutional neural networks. *Stud Health Technol Inform*, *235*, 246–250.

Hussain, F. (2017). Internet of everything. In *Internet of things: Building blocks and business models* (pp. 1–11). Springer International Publishing.

Ikonomakis, M., Kotsiantis, S., & Tampakas, V. (2005). Text classification using machine learning techniques. *WSEAS transactions on computers*, *4*(8), 966–974.

Iyyer, M., Enns, P., Boyd-Graber, J., & Resnik, P. (2014). Political ideology detection using recursive neural networks. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1113–1122).

Jurka, T. P., Collingwood, L., Boydstun, A. E., Grossman, E., & van Atteveldt, W. (2013). Rtexttools: A supervised learning package for text classification. *R Journal*, *5*(1), 6–12.

Khan, A., Baharudin, B., Lee, L. H., & Khan, K. (2010). A review of machine learning algorithms for text-documents classification. *Journal of advances in information technology*, *1*(1), 4–20.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 conference on empirical methods in natural language processing* (pp. 1746–1751).

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150.

Kuhn, M. (2008). Building predictive models in r using the caret package. *Journal of statistical software*, *28*, 1–26.

Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. (2015). From word embeddings to document distances. In *International conference on machine learning* (pp. 957–966).

Lin, Y., Meng, Y., Sun, X., Han, Q., Kuang, K., Li, J., & Wu, F. (2021, August). Bert-GCN: Transductive text classification by combining GNN and BERT. In *Findings of the association for computational linguistics: Acl-ijcnlp 2021* (pp. 1456–1462). Online.

Liu, Y., Liu, Z., Chua, T.-S., & Sun, M. (2015). Topical word embeddings. In *Twenty-ninth aaai conference on artificial intelligence.*

Loftis, M. W., & Mortensen, P. B. (2020). Collaborating with the machines: A hybrid method for classifying policy documents. *Policy Studies Journal*, *48*(1), 184–206.

Mettler, S. (2016). The policyscape and the challenges of contemporary politics to policy maintenance. *Perspectives on Politics*, *14*(2), 369–390.

Mirończuk, M. M., & Protasiewicz, J. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, *106*, 36–54.

Navarretta, C., & Hansen, D. H. (2020). Identifying parties in manifestos and parliament speeches. In *Proceedings of the second parlaclarin workshop* (pp. 51–57).

Neumann, A., Laranjeiro, N., & Bernardino, J. (2018). An analysis of public rest web service apis. *IEEE Transactions on Services Computing*.

Novotný, V., Ayetiran, E. F., Štefánik, M., & Sojka, P. (2020). *Text classification with word embedding regularization and soft similarity measure.*

Onan, A. (2018). An ensemble scheme based on language function analysis and feature engineering for text genre classification. *Journal of Information Science*, *44*(1), 28-47.

Onan, A. (2019). Two-stage topic extraction model for bibliometric data analysis based on word embeddings and clustering. *IEEE Access*, *7*, 145614-145633.

Onan, A. (2020a). Mining opinions from instructor evaluation reviews: A deep learning approach. *Computer Applications in Engineering Education*, *28*(1), 117-138.

Onan, A. (2020b). Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks. *Concurrency and Computation: Practice and Experience*, e5909.

Onan, A., Korukoğlu, S., & Bulut, H. (2016). Ensemble of keyword extraction methods and classifiers in text classification. *Expert Systems with Applications*, *57*, 232-247.

Onan, A., & Toçoğlu, M. A. (2021). A term weighted neural language model and stacked bidirectional lstm based framework for sarcasm identification. *IEEE Access*, *9*, 7701-7722.

Pontes, F. J., Amorim, G., Balestrassi, P. P., Paiva, A., & Ferreira, J. R. (2016). Design of experiments and focused grid search for neural network parameter optimization. *Neurocomputing*, *186*, 22–34.

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

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

Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning (icml-03)* (pp. 616–623).

Rosenthal, S., Farra, N., & Nakov, P. (2017). Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th international workshop on semantic evaluation (semeval-2017)* (pp. 502–518).

Russo, F., & Cavalieri, A. (2016). The policy content of the italian question time. a new dataset to study party competition. *Rivista Italiana di Politiche Pubbliche*, *11*(2), 197–222.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, *20*(1), 61-80.

Slapin, J. B., & Proksch, S.-O. (2014). Words as data: Content analysis in legislative studies. In *The oxford handbook of legislative studies* (pp. 126–144). Oxford University Press Oxford.

Syarif, I., Prugel-Bennett, A., & Wills, G. (2016). Svm parameter optimization using grid search and genetic algorithm to improve classification performance. *Telkomnika*, *14*(4), 1502.

Tharwat, A. (2018). Classification assessment methods. *Applied Computing and Informatics*.

van Atteveldt, W., Welbers, K., & van der Velden, M. (2019). Studying political decision making with automatic text analysis. In *Oxford research encyclopedia of politics* (pp. 1–11).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In *Proc. of the 31st int'l conf. on neural information processing systems* (p. 6000–6010). Curran Associates Inc.

Wiedemann, G. (2019). Proportional classification revisited: Automatic content analysis of political manifestos using active learning. *Social Science Computer Review*, *37*(2), 135–159.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., . . . Rush, A. M. (2020). *Huggingface's transformers: State-of-the-art natural language processing.*

Yoo, S., Song, J., & Jeong, O. (2018). Social media contents based sentiment analysis and prediction system. *Expert Systems with Applications*, *105*, 102–111.

Zhang, Y., Jin, R., & Zhou, Z.-H. (2010). Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, *1*(1-4), 43–52.