Document Enrichment using DBPedia Ontology for Short Text Classification

Jernej Flisar
UM FERI
Maribor, Slovenia
jernej.flisar@um.si

Vili Podgorelec
UM FERI
Maribor, Slovenia
vili.podgorelec@um.si

ABSTRACT

Every day, millions of short-texts are generated for which effective tools for organization and retrieval are required. Because of the short length of these documents and of their extremely sparse representations, the traditional text classification methods are not effective. We propose a new approach that uses DBpedia Spotlight annotation tools, to identify relevant entities in text and enrich short text documents with concepts derived from those entities, represented in DBpedia ontology. Our experiments show that the proposed document enrichment approach is beneficial for classification of short texts, and is robust with respect to concept drifts and input sources. We report experimental results in three challenging collections, using a variety of classification methods. The results show that the use of DBpedia ontology significantly improves the classification performance of classifiers in short-text classification.

CCS CONCEPTS

• **Information systems** → **Clustering and classification**; *Web mining*; *Ontologies*;

KEYWORDS

short text classification, DBPedia, ontology, text enrichment

ACM Reference Format:
Jernej Flisar and Vili Podgorelec. 2018. Document Enrichment using DBPedia Ontology for Short Text Classification. In *WIMS '18: 8th International Conference on Web Intelligence, Mining and Semantics, June 25–27, 2018, Novi Sad, Serbia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3227609.3227649

1 INTRODUCTION

Text classification (TC) deals with the problem of automatically assigning predefined categories to text documents. It plays an important role in many research domains like information retrieval, text mining and machine learning. TC has been successfully applied in wide variety of application domains, ranging from news and spam filtering to web page categorization and document organization. The majority of text documents contains texts from letters, news, articles, blogs etc. Those text documents are traditionally represented as bag of words (BOWs) [24]. In a BOW, each document is represented as a feature vector, which consists of all the words in the documents. Different classification methods, such as Naïve Bayes (NB), k-nearest neighbors (kNN), decision trees (DTs), support vector machines (SVMs), and ensemble methods (like Random Forest, RF) have been used over BOW on different benchmark collections (20Newsgroup, Reuters-2157, etc.), achieving solid results [2, 24].

With the explosion of online communication, social networks, and micro-blogging, short texts have emerged. Short texts are substantially different from classical text documents, since they are usually more informal, noisier and less topic-focused. Because texts are short, they do not provide enough word co-occurrences, shared context or informal sentence expressiveness, therefore, conventional classification approaches usually fail to achieve reasonable results. This is due to high data sparseness, since BOW representation does not handle short texts well [21].

Attempts have been made in the literature to overcome data sparseness for improving short text classification (STC). The most common approach to solve this problem is to reduce sparseness by enriching short text representation models, which can then be handled by traditional classification methods. To enrich document representation, short texts are expanded with the utilization of external resources such as search engines (e.g. Google) or open source knowledge bases (e.g. WordNet and Wikipedia) to enrich BOW representation with new dimensions (e.g additional words). Researches have shown positive improvements in classification performances using text enrichment techniques [10, 21, 28, 31].

Our work was inspired by the idea of using external data sources mentioned above. As a knowledge base we used DBpedia[1], which extracts structured information from Wikipedia and combines those information into enormous, cross-domain knowledge base. The advantage of using DBpedia is the representation of knowledge as a single ontology. The DBpedia ontology consists of 320 classes, which form subsumption hierarchy and are described with 1,650 different properties. By providing a hierarchical classification of concepts, it allows one to search for sub- or super-classes of a particular concept, which can be then used to compute the similarity of different, but related concepts. Additional advantage is provided by extended datasets, like Topical Concepts, Lexicalization and Topic Signatures, which have been specially made to support natural language processing (NLP) and text mining tasks [16].

The most common way to utilize ontologies for text classification is to mach terms in a document with ontology concepts. Matched ontology concepts are then either used as a replacement or are added as additional features [28]. To detect an ontology concept

[1] http://wiki.dbpedia.org/

WIMS '18, June 25–27, 2018, Novi Sad, Serbia

ACM ISBN 978-1-4503-5489-9/18/06…$15.00
https://doi.org/10.1145/3227609.3227649

in a text, we use the DBpedia Spotlight tool [18], since it takes full advantage of the DBpedia ontology. This tool is very efficient and accurate in identifying meaningful terms in text and linking them to DBpedia resources [20, 22]. Additionally, DBpedia Spotlight computes scores such as *commonness* and *topical relevance* for each annotated resource. *Commonness* represents the number of times a resource is mentioned in Wikipedia and *topical relevance* shows resource relatedness to text.

Several researches used DBpedia to solve different NLP tasks [18], like entity extraction, question answering, word sense disambiguation and others [1, 11]. Yet, to the best of our knowledge, DBpedia ontology and its datasets were not used to solve short text classification tasks.

We evaluated our approach over two available open datasets – search snippets [21] and article news [26] – and one tweets dataset, which we collected and built for the testing purposes. On all datasets, multiple mainstream text classification methods (e.g. SVM, NBM, kNN, Rocchio) were used as the base classifiers, with BOW representation model. Then we enriched the BOW representation with different proposed enrichment techniques, using DBpedia and its datasets. Our proposed enrichment approach improved the classification accuracy on different datasets from 6% to 10% when compared with the baseline approach (without enrichment) and the same selected classifiers. This confirmed the superiority of our proposed approach using DBpedia ontology over baseline method and are comparable with other state-of-the art methods.

The rest of this paper is organized as follows. Section 2 presents related work on short text enrichment methods for improving classification performance. The DBpedia alongside with its special datasets and tools, which we have been used in our research, are presented in Section 3. The proposed method for document enrichment using DPpedia is explained in Section 4. Section 5 provides the experimental settings, while in Section 6 the obtained results are discussed in detail. Finally, we provide some discussions and implications of the proposed method in Section 7, after which the final remarks and some plans for future work conclude the paper in Section 8.

2 RELATED WORK

Text classification is a well-researched problem in field of text mining. On the contrary, classification of short texts is more difficult, since the traditional methods for traditional text mining cannot be applied efficiently, because of high data sparseness. Some highly related text documents have no words in common, since they are very short, which makes it very difficult to measure similarity of those documents.

Many studies in the literature tried to handle the problem of high sparseness in short text classification tasks. Several approaches have been made to overcome this problem in order to get better classification results. Current approaches try to expand document representation with additional features, by generating or adding additional information. Most approaches rely on external resources like Wordnet and Wikipedia to generate features for related words, which do not occur in text documents.

Many attempts have been made in literature to add background knowledge to document representation for short text classification. Gabrilovich et al. proposed a method to enhance text classifier with generated content from Wikipedia [10]. They map text documents to the most relevant concepts in Wikipedia by comparing the textual overlap between each document and article. Downside of this feature generation procedure is the high processing effort, because each document need to be processed multiple times. The experimental results have shown that their method is better than the traditional BOW method using SVM as a text classifier. An opposite direction was used by Sun [25]. Instead of expanding a feature vector, he reduced the number of features to represent a document.

Janik and Kochut developed an ontology-based text classification using ontology derived from Wikipedia. Their approach does not require a training set, so it is very useful for classification where there is small or none training data available [14]. Wang and Domeniconi proposed a method that attempts to highlight semantic content of document representation by embedding background knowledge constructed from Wikipedia into semantic kernel, which is then used to expand the BOW representation of documents. With their proposed method they kept multi-word concept unbroken and captured the semantic closeness of synonyms [27].

Another approach used in literature is to integrate the context data with a set of hidden topics discovered from related corpora. Phan manually built a large and rich universal dataset and derived a set of hidden topics through topic models such as Latent Dirichlet Allocation (LDA) from Wikipedia as external corpora [21]. Because a single external dataset cannot cover enough topics, Zhu et al. [32] used multiple external corpora to identify topics for better classification performance. Those researches are focused on a single granularity topic which is usually not sufficient to model feature space effectively. Chen et al. [5] improved those methods by introducing the multi-granularity topics.

Rather than enriching BOW representation, Daniele Vitale et al. [26] utilized Wikipedia concept vector to represent the text document, using TAGME tool [9], which performs mapping of short texts to Wikipedia. They selected top-k Wikipedia concepts to characterize a specific category and computed the semantic relatedness to each category a document belongs to. This work is the most similar to our approach, because it uses an annotation tool to detect relevant concepts in text documents. The differences between this study and ours is the selected annotation tool and that we use supervised methods for classification instead of computing semantic relatedness. The DBpedia was used also in [15] to measure semantic relatedness of words using DBpedia, where words were nodes in RDF graph, and the shortest the path between the nodes was the more similar are the words.

3 DBPEDIA KNOWLEDGE BASE

Wikipedia has grown into one of the central knowledge sources repository and is maintained by thousands of contributors. Wikipedia articles does not only contain unstructured text, but also contain different types of structured information, such as infobox templates, categorization information, links to internal articles and external Web pages. The DBpedia uses Wikipedia structure to extract various kinds of structured information from Wikipedia editions in 125 languages and combines this information into enormous, cross-domain knowledge base. For every article in Wikipedia, DBpedia

creates its own uniform resource identifier (URI), to identify articles as concepts. Resource Description Framework (RDF) is used for representing extracted information [3].

The English version of the DBpedia describes 4.58 million things, out of which 4.22 million are classified in an ontology, which consists of 320 classes. Altogether the DBpedia consists of 3 billion pieces of information (RDF triples) that describe 583 million facts [7]. DBpedia is divided into several different datasets that can be useful for several NLP tasks, like entity recognition, word disambiguation and others. The core datasets from DBpedia include general facts about extracted resources, as well as inter-language links. A DBpedia resource is described by a label, a short and long English abstract, a link to the corresponding Wikipedia page, different classification schemes, types and properties. Datasets were created to support the DBpedia Spotlight framework [19].

3.1 Annotation tools

Many tools for detecting meaningful terms in text and linking them to relevant knowledge base concepts, especially Wikipedia, have been developed (DBpedia Spotlight[2], TagMe[3], Wikify!, Zemanta[4], OpenCalais[5], AlchemyAPI[6]) [22]. In provided text, these tools can detect known concepts in their knowledge bases. Detected concepts are mapped to topics or predefined taxonomy classes and are disambiguated using URIs. For example, let us consider the following short text: *"Goran Dragic re-sign with the Miami Heat"*. Annotator tool is able to detect *Goran Dragic* and *Miami Heat* as meaningful concepts, which are mapped as resources to one of available external knowledge bases (e.g. Wikipedia). Those detected concepts can be represented by topics like *basketball player* and *professional sport team*. The use of annotation tools can be very important, because it may help in detecting similar texts with no words in common, which is one of the limitations of the classical BOW model.

Obviously, in order to enhance the short text classification performance, we should enrich short texts without introducing too much noise, because it can lead to decreased performance [30]. Disambiguation process of annotation tools prevents such errors due to word ambiguity. For example, consider the following two texts: *"Michael Jordan played basketball for Chicago Bulls"* and *"Michael Jordan is professor at Berkeley"*. In a word-based approach it would be difficult to figure out their different semantic meaning, while annotator tool would link *Michael Jordan* from the first text to different URI than *Michael Jordan* from the second text, since it would be able to detect that context of these two concepts are semantically far apart.

Each tool has its own strengths and weaknesses, which has been widely studied and evaluated in [22]. We decided to use the DBpedia Spotlight annotation tool because it is free, open source and has good performance results. Additionally, it maps concepts to DBpedia ontology that has a wide range of datasets available which we used in our study.

[2]https://github.com/dbpedia-spotlight
[3]https://tagme.d4science.org/tagme/
[4]http://www.zemanta.com/
[5]http://www.opencalais.com/
[6]https://github.com/AlchemyAPI

3.2 DBPedia spotlight

DBpedia Spotlight is an open source tool for automatic detection of DBpedia resources written in natural language texts [6]. It provides interface for detection and annotation of concepts as resources in a REST-based web service. It also provides word sense disambiguation scores for words which can have multiple meanings. Disambiguation algorithm is based upon cosine similarities and modified TF-IDF weights.

The detection is performed by substring matching using a kind of prefix tree and the Aho-Corasick algorithm [18]. The DBpedia Spotlight recognizes that words describing concepts have been mentioned in text (e.g. "Apple"), and subsequently matches these names to unique identifiers (e.g. company or the fruit). The topical relevance of the detected resource for the given context can be measured by the similarity score returned by the disambiguation step. The score is higher for texts which match more closely to the recorded observations for a DBpedia resource [18]. Additional knowledge about resources can be obtained by querying provided datasets like Topic Signature Data Set, Abstract Data Set, Categories Data Set and Lexicalization Data Set, where concept categories, topics and their descriptions are provided.

Concepts in DBpedia are fine-grained, which can increase the capacity to distinguish between closely related resources. For example, given the resource *LeBron James*, we can obtain from DBpedia many fine-grained concepts such as topics (nba, player, game), types (Athlete, Basketball Player) and categories (Basketball Players, Olympics Medalists, Sports people). These concepts can serve as additional knowledge to enrich document representation with new features.

4 THE PROPOSED METHOD

Our method aims to improve short text classification performance by taking advantage of DBpedia and its datasets to add additional knowledge to short texts. By adding external knowledge, we can better represent short texts, so that semantically similar texts with no words in common can have similar context. In order to utilize this enrichment approach we built the framework, presented in Fig. 1.

Our framework consists of three steps. In the first step, the identification of concepts in texts is done using DBpedia Spotlight as annotation tool. Identified concepts are annotated as DBpedia resources. In the second step, the concept selection process is done. Concepts with similarity value higher than certain threshold t are kept. By default, we use $t>=0.1$. We examined the impact of using different values for t in our experiments.

After the final set of concepts is selected, in the third step, we query and extract additional information for these concepts from DBpedia. This newly extracted information is then added to original text to enrich it with additional knowledge like categories, types or topics of identified concepts. We applied four different text enrichment techniques, which are described in Section 4.2. In the last step, the traditional text classification process is applied.

4.1 Annotation framework

Given an input short text d, we use DBpedia Spotlight to detect the set of DBpedia resources in text, with their similarity scores which measure resource relatedness with context. Then, additional

Table 1: Document vectors of two texts

Anze	Dragic	Goran	Heat	Kings	Kopitar	Miami	Slovenia	camp	hockey	hold	re-sign	star	training	with
0	1	1	1	0	0	1	0	0	0	0	1	0	0	1
1	0	0	0	1	1	0	1	1	1	1	0	1	1	0

Table 2: Document vector of two enriched texts

GoranDragic	Athelete	BasketballPlayer	MiamiHeat	SportTeam	BasketballTeam	LAKings	HockeyTeam	AnzeKopitar	HockeyPlayer
1	1	1	1	1	1	0	0	0	0
0	1	0	0	1	0	1	1	1	1

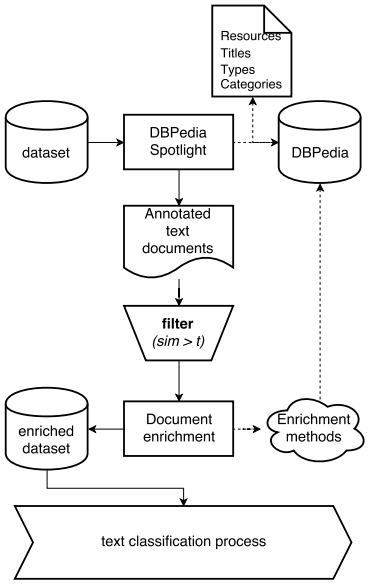

Figure 1: The proposed text enrichment framework

concepts of these resources are extracted from DBpedia ontology, like types, categories, topics and abstract. These concepts are later added as text to original document if value of similarity score is higher than certain threshold *t*.

As an example, two short sports articles are given: *"Goran Dragic re-sign with the Miami Heat"* and *"Kings star Anze Kopitar holds hockey training camp in Slovenia"*. Using BOW as a document representational model would produce the following feature vectors which are presented in Table 1.

The above two texts share no common features, despite both texts belong to the same category `Sport`. But if we can extend texts, so that *Goran Dragic* and *Anze Kopitar* can be represented by the same concept `Athlete`, and *Miami Heat* and *Kings* with their shared concept `SportsTeam`, we could importantly improve their relation to the Sport category. When our annotation process is applied to above texts, extended texts would look like *"Goran Dragic* `#Goran_Dragic #Athlete #BasketballPlayer` *re-sign with the Miami Heat* `#Miami_Heat #BasketballTeam #SportTeam` *"* and *"Kings* `#LA_Kings #SportTeam #HockeyTeam` *star Anze Kopitar* `#Anze_Kopitar #Athlete #HockeyPlayer` *hold hockey training camp in Slovenia"*. This would extend document representation model with additional features as shown in Table 2. With additional features, now both texts share common features like `SportTeam` and `Athlete`.

4.2 Document enrichment models

In order to overcome the problem of high data sparseness when using BOW as representation model in short text classification tasks, we implemented four different documents enrichment methods, utilizing DBpedia. All methods use the same parameters and settings for identifying concepts in text. The difference between them is the addition of textual information that is added to the original text. We consider adding types, topics and categories to original text in four different ways:

- BOC (Bag-of-Concepts): Original text is extended with concepts from DBpedia identified by annotation tool. With this method we solve problem of ambiguity, since added concepts have unique identifiers. Additionally, multi-word terms are combined into a single word, so that a term like *Goran Dragic* is presented as the single word *Goran_Dragic*.

$$BOC = [w_1, \ldots, w_n, c_1, \ldots, c_m]$$

  where w_i are words in the text and c_i are the identified concepts which are added to original text.
- AT (Additional Types): First, the text is extended using the *BOC* method. Additionally, types of identified concepts are acquired from DBpedia ontology and added to the original text. Types are usually hyphens of identified concepts like `Person`, `Athlete`, `Book`, etc...

$$AT = BOC + [t_1, ..., t_m]$$

- ATT (Additional Types and Topics): The text is first extended with *AT* method. Then, topics of identified concepts are added from provided DBpedia dataset. Examples of added topics are single words like *basketball*, *nba*, *player*.

$$ATT = AT + [t_1, ..., t_m]$$

- ATTC (Additional Types, Topics and Categories): In this method, *ATT* method is first applied to the original text. For each concept, its categories are queried from DBpedia and DBpedia extracts categories from Wikipedia. With retrieved categories, text is additionally extended. With addition of categories, we can solve problem of relatedness between

words, since words with similar meanings can have same categories. For example, concepts *hockey* and *basketball* have the same category `Sport`.

$$ATTC = AT + [c_1, ..., c_m]$$

After the text enrichment, we use traditional text classification steps in order to train classification algorithm.

4.3 Classifier learning

Initial phase in any classification process is usually preprocessing, which is composed of three steps: tokenization, stop words removal and stemming.

(1) Tokenization. Tokenization is the process that splits text into words, phrases or other meaningful elements, so-called tokens.
(2) Stop-words removal. Some words like prepositions don't carry much meaning and are used frequently in text documents. Typically, they are filtered out, since they don't contribute to distinguishing different categories of text. We use standard English stop-word list from Weka[7].
(3) Stemming. Stemming is the process of reducing words to their base or root form. We apply Porter stemmer[8] algorithm to reduce each word to its root form. For example, the words "learn", "learning" and "learner" would all be reduced to their base form "learn".

After the texts are preprocessed, we use BOW to represent each text with a word vector. Since we enrich the original texts with additional textual information, we should reduce sparseness of BOW representation and improve text classification results. We examined the classification results using the proposed four enrichment methods over different text classifiers.

5 EXPERIMENTAL SETTINGS

5.1 Classifiers

Five different classification algorithms were employed to investigate contribution of the proposed text enrichment methods to the classification performance: Naïve Bayes multinomial (NBM), Rocchio, support vector machine (SVM), k-nearest neighbors (kNN), and ensemble method Random Forest (RF). All selected classifiers have been commonly used for text classification tasks and are proven as successful classification methods [12, 24].

- *Naïve Bayes Multinomial.* Naïve Bayes classifier assumes that the value of a particular feature is unrelated to the presence or absence of any other feature, given the class variable. A Naïve Bayes classifier considers each of these features to contribute independently to the probability, regardless of the presence or absence of other features. In spite of their naïve design and apparently over-simplified assumptions, Naïve Bayes classifiers have worked quite well in many complex real-world situations. Two different models are commonly used for text classification, Multivariate Bernoulli and Multinomial. We used the latter, since usually it has better performance [17].

[7]https://www.cs.waikato.ac.nz/ml/weka/
[8]https://tartarus.org/martin/PorterStemmer/

Table 3: Google Snippet dataset

Category	#train	#test	Features	Values
Business	1200	300	Number of abstracts	2280
Computer	1200	300	Vocabulary size	10033
Health	1200	300	Term avg. per abstract	13.1
Sports	880	300		
Culture&Arts	1120	300		
Education&Science	1880	300		
Engineering	2360	150		
Politics&Society	1200	300		
Total	10060	2280		

- *Rocchio.* The algorithm is based on distance measurement [23]. First it computes the central vector of every class set with the computing method using the arithmetic mean of all the training documents. For new text to be classified, distance between the feature vector of the text and the central vector of every class is calculated with cosine similarity.
- *kNN.* K-nearest neighbors (kNN) classification is an instance-based learning algorithm that has been applied to text categorization. In this classification paradigm, k nearest neighbors of a text document are computed first. Then the similarities of this document to the k nearest neighbors are aggregated according to the class of the neighbors, and the test document is assigned to the most similar class (as measured by the aggregate similarity) [13].
- *SVM.* Support vector machine (SVM) is a supervised classification algorithm that has been applied very successfully to text classification [29]. The main principle of SVM is to determine the separators in the search space which can best separate different classes. More formally, a SVM constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other machine learning tasks [2].
- *Random Forest.* Random Forest (RF) classifier is based on ensemble learning. Ensemble learning employs multiple classification models. The generalization ability of an ensemble is usually significantly better than that of a single classifier [2]. RF corrects for decision trees' habit of overfitting to the training set. It was developed by Leo Breiman and Adele Cutler. The method combines Breiman's bagging idea and the random selection of features in order to build a collection of decision trees with controlled variance [4].

For each individual classification algorithm the default values have been used for all the parameters, as it is not our aim to optimize the results by fine-tuning the existing algorithms but rather to study the influence of the proposed enrichment methods on the results.

5.2 Datasets

We conducted experiments on three different datasets from different domains, namely search snippets, news and tweets, using a variety of classification methods. The first dataset, Google Snippet, was collected by Phan [21]. It contains 10k train and 2k test search snippets that were selected from the results of web search transactions using predefined phrases of different domains. Snippets

Table 4: NEWS dataset

Category	#train	#test	Features	Values
Business	2676	2675	Number of documents	15216
Entertainment	1643	1643	Vocabulary size	30791
Health	925	926	Term avg. per document	33.7
Science	1436	1436		
Sport	3018	3018		
US	2392	2391		
World	3126	3127		
Total	15216	15216		

Table 5: Twitter dataset

Category	#train	Features	Values
Business	269	Number of documents	2371
Entertainment	500	Vocabulary size	9199
Health	223	Term avg. per document	13.6
Science	177		
Sport	500		
US	202		
World	500		
Total	2371		

have length of about 13 words, and are labelled with 8 categories. Snippets from test and train data share very few common words, which makes them really difficult to predict using traditional text classification methods. The second dataset, called NEWS, was built by [26]. It is composed of 32k short texts collected from RSS feeds of three newspapers (nyt, usatoday, reuters). There are no predefined training-test partitions for this dataset, so we randomly split it into two equal halves to create train and test sets. Detailed information about the dataset is presented in 4. The last dataset Twitter is a collection of 5k tweets that were posted in July 2015. We selected seven hash tags as classes: World, US, Science, Sport, Business, Health, and Entertainment. For each hash tag, we randomly selected from 200-500 tweets and removed hyperlinks and twitter reserved words that starts with '#' (hash tag) or '@' (username). Basic information about datasets is presented in Tables 3 to 5.

5.3 The baseline method

In order to measure the improvement of the proposed approaches, we compared each of our four document enriched methods against original text dataset as a baseline method (BASE). Each text dataset is represented as BOW model where *tf-idf* term weighing scheme was applied. The same classification algorithms are then trained over all different enrichment methods and the baseline. Weka implementations were used for all classifiers with the exception of SVM, where LibSVM[9] was used, and Rocchio which we implemented in Java. All classifiers were trained using the default parameter values. For kNN, k was set to 1 and for RF, the number of trees was set to 50. For SVM, the linear kernel was used.

[9]https://www.csie.ntu.edu.tw/~cjlin/libsvm/

Table 6: F1 score of different text enrichment methods for Snippet dataset using different classifiers

Classifiers	NBM	Rocchio	SVM	kNN(1)	RF(50)
BASE	0.8047	0.7527	0.6846	**0.4005**	0.6070
BOC	0.8213	0.7626	0.6946	0.3135	0.6077
AT	0.8275	0.7758	0.7081	0.3108	0.6070
ATT	**0.8395**	**0.8161**	**0.7448**	0.3501	0.6302
ATTC	0.8223	0.8089	0.7329	0.3067	**0.6423**

Table 7: F1 score of different text enrichment methods for NEWS dataset using different classifiers

Classifiers	NBM	Rocchio	SVM	kNN(1)	RF(50)
BASE	0.8888	0.8164	0.8963	0.7939	0.8492
BOC	**0.8947**	**0.8239**	**0.9010**	0.7839	0.8555
ATT	0.8866	0.8094	0.8995	**0.8293**	0.8680
ATTC	0.8547	0.8104	0.8956	0.7966	**0.8781**
AT	0.8938	0.8196	0.8988	0.7906	0.8630

Table 8: F1 score of different text enrichment methods for Twitter dataset using different classifiers

Classifiers	NBM	Rocchio	SVM	kNN(1)	RF(50)
BASE	0.5982	0.5958	0.5610	0.4016	0.5281
BOC	0.5961	0.6019	0.5713	0.3983	0.5331
AT	0.6063	0.6105	0.5824	0.4033	0.5379
ATT	**0.6100**	0.6054	0.5831	0.4101	**0.5557**
ATTC	0.5952	**0.6143**	**0.5889**	**0.4181**	0.5449

5.4 Evaluation methods

We evaluated the performance of text classification using macro-averaged F1 score [24]. The F1 score can be interpreted as a weighted average of the precision and recall, where precision is the number of correct positive predictions divided by the number of all positive predictions, and recall is the number of correct positive predictions divided by the number of all positive instances:

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{1}$$

The micro-averaged F1 score operated at document level and is primarily affected by classification accuracy of larger categories, while macro-averaged F1 score averages results over all categories. Following an established practice, we used fixed data split for all datasets and the combination of Friedman test and Wilcoxon signed-rank post-hoc test to statistically evaluate the proposed enrichment methods against the baseline over multiple datasets [8].

6 EXPERIMENTAL RESULTS

Our main experiment was two-fold. First, we tested the classification performance of short text classification using different proposed document enrichment methods and compared them with the baseline approach, over five text classifiers (SVM, NBM, Rocchio, kNN, and RF). Besides, in order to tune the *similarity* threshold parameter of added concepts, during enrichment process of short

text documents, we measured how different number of selected concepts affects the classification performance.

6.1 Enrichment techniques

We used the four proposed document enrichment methods for each of the text documents in three datasets. In the experiment we evaluated the classification performance, with F1 score, where different enrichment methods were used with selected classifiers. The results are presented in Tables 6 to 8 with best F1 score, among each method, shown in bold.

For the Snippet dataset (see Table 6), the best overall result was achieved using ATT enrichment method with NBM classifier. Additionally, improvements in classification performance with document enrichment methods over baseline were achieved in 4 out of 5 classifiers, where only performance of kNN decreased. The improvements in classification performance of the best enrichment method over baseline were between 3% (with NBM and RF classifiers) and 6% (with Rocchio and SVM classifiers), while in case of kNN the performance decreased for 5%. For all the enrichment methods and the baseline, the best performance in the Snippet dataset was achieved using the NBM classifier.

In the NEWS dataset (see Table 7), the best overall result was achieved using BOC method with SVM classifier. The F1 score was 90.10%, which is a slight (0.5%) improvement over the baseline. With all the classifiers, at least one of the proposed enrichment methods outperformed the baseline, with the best improvement of 3.50% in the case of ATT method and kNN classifier. It is interesting to note that the biggest performance improvement in the NEWS dataset was achieved with the kNN classifier, the only classifier whose performance in the Snippet dataset decreased with the use of enrichment methods. For the NEWS dataset, the best performance was achieved using the SVM classifier for all the enrichment methods as well as the baseline.

The ATTC enrichment method with Rocchio classifier provided the best overall performance in the Twitter dataset (see Table 8). Like in the NEWS dataset, the improvement over baseline in performance was made when using enrichment methods (in at least one enrichment method) for all classifiers. The best performance improvement over baseline (2.7%) was made when using the RF classifier and ATT enrichment method. In the case of the Twitter dataset, no single classifier was the best for all cases – Rocchio achieved the best result in 3 enrichment methods, while NBM achieved the best result in the remaining enrichment method and the baseline.

To evaluate the statistical significance of these results, we first applied the Friedman test by calculating the Friedman asymptotic significance for various classification algorithms. The test confirmed (p<0.001) that there are statistically significant differences between specific classifiers. The best results were obtained with NBM (average rank of 4.40 among 5 algorithms – 7 wins, 7 second best results and 1 third place), followed by SVM (average rank 3.67, 5 wins) and Rocchio (average rank 3.47, 3 wins). The post-hoc Wilcoxon signed ranks test with Holm-Bonferroni correction revealed that NBM significantly outperformed all other classification algorithms: SVM (p=0.031), Rocchio (p=0.006), RF (p=0.001) and kNN (p=0.001).

Similarly, we statistically compared the baseline and four enrichment methods (BOC, ATTC, AT, and ATT). The Friedman test

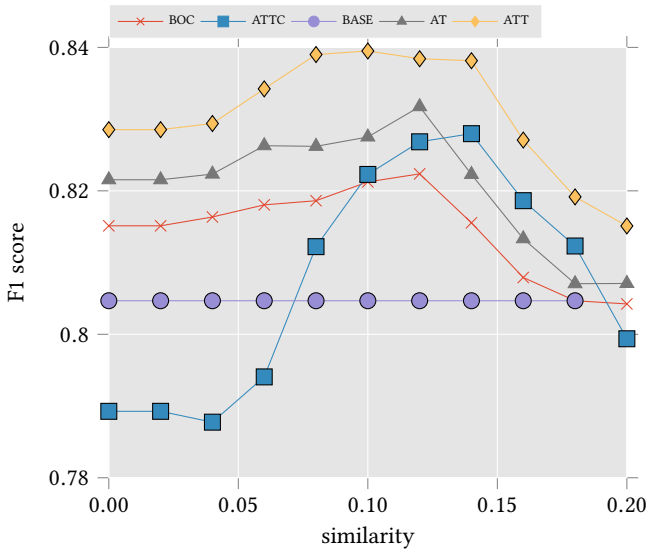

Figure 2: Classification performance of various enrichment methods using different threshold for similarity with NBM classifier on the Snippets dataset.

confirmed significant differences between these methods (p=0.006). The best results were obtained when using ATT (with average rank of 4.00 among five methods – 6 wins, 6 second, 1 third, 1 fourth and 1 worst result), followed by ATTC (average rank 3.33, 5 wins), AT (average rank 3.13, the only method with no win and also no worst result), BOC (average rank 2.6, 3 wins), and baseline (average rank 1.93, 1 win). The post-hoc Wilcoxon signed rank test with Holm-Bonferroni correction revealed that ATT significantly outperformed all other methods (AT, p=0.023; BASE, p=0.015; BOC, p=0.006), with the exception of ATTC (p=0.112).

6.2 Similarity threshold value

The next experiment was performed in order to test how the classification accuracy is affected by the similarity threshold of tagged entity of DBpedia. The similarity value of a resource is computed as a cosine distance between context vectors of the resource and context surrounding the resource in text using tf-icf weighting scheme. We estimated many settings for the DBpedia annotator data with different threshold values for similarity (from 0.0 to 0.2). After the enrichment process, for each enrichment method, eleven NBM classifiers were built on the training data where additional resources were added only if their computed similarity value was higher than the specific threshold. We decided to use the NBM classifier as it outperformed all other classification algorithms, as shown in the previous experiment. Figures 2-4 show how accuracy vary when the similarity threshold for enrichment method changes.

The classification performance curves of different enrichment methods (representation models) using NBM classifier on the Snippets dataset are shown in Fig. 2, from which we can see that enrichment models generally surpassed the basic model. The two exceptions were the ATTC model, whose results were lying behind

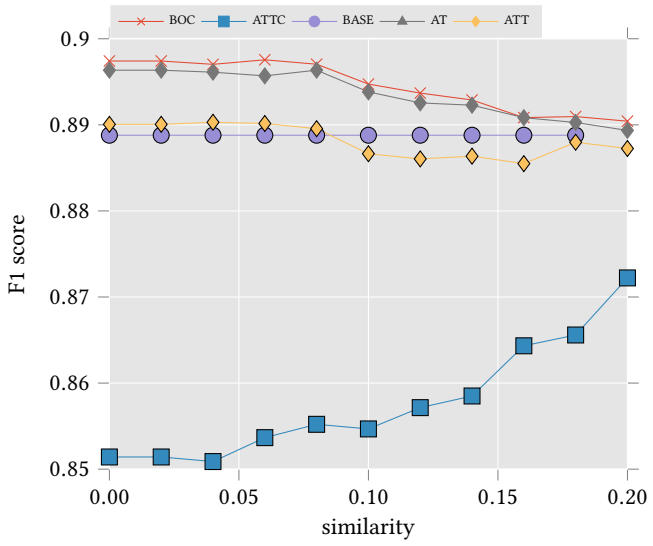

Figure 3: Classification performance of various enrichment methods using different threshold for similarity with NBM classifier on the NEWS dataset.

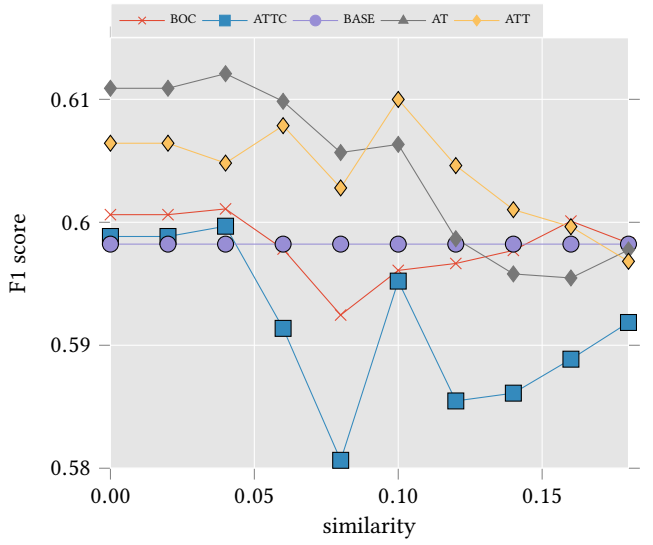

Figure 4: Classification performance of various enrichment methods using different threshold for similarity with NBM classifier on the Twitter dataset.

the baseline when threshold was set lower than 0.08 or higher than 0.18, and the BOC model, whose result became equal to baseline when threshold was higher than 0.18. For all the enrichment methods the best results were obtained with the threshold set between 0.10 and 0.14.

Similarly, the classification performance results on the NEWS dataset are presented in Fig. 3. In this case, we can see that the BOC and the AT enrichment methods surpassed the baseline regardless of the similarity threshold. On the other hand, the results of the ATTC method were behind the baseline for all the tested threshold values. The results of the ATT enrichment method were better than the baseline with the threshold values below 0.1, after which they became worse.

Finally, the classification performance results on the Twitter dataset are presented in Fig. 4. This is a very challenging dataset, since it is very small and contains a lot of noise. Nonetheless, the proposed enrichment methods yield an improvement of classification accuracy over the baseline approach. In this case, the ATT and the AT enrichment methods turned out to provide the best results. The ATT method surpassed the baseline for all the threshold values but 0.20, while the AT method surpassed the baseline for the threshold values up to 0.12. On the other hand, the results of the ATTC method were behind the baseline for the threshold values bigger than 0.05, while the results of the BOC method in this case alternated with the ones of the baseline method.

7 DISCUSSION

The experimental results of the proposed enrichment methods, using three different short text datasets and five different classification algorithms, showed that the application of enrichment methods improved the classification performance significantly. In general, the NBM classification algorithm turned out to be the best, providing better results than all other classifiers with statistical significance. Its dominance was especially evident in the case of Google Snippets dataset, while in the other two datasets its backlog for the best was only marginal.

The comparison of specific enrichment methods showed that generally the best results were obtained with the ATT method, followed by the ATTC and AT methods. Regarding the number of sources added, the enrichment methods can be ordered as follows: BOC, AT, ATT, and finally ATTC. Thus, BOC adds the least additional resources, AT adds some more, and so on. The obtained results suggest that adding resources is helpful to some extent (AT outperforms BOC, ATT outperforms AT), but adding too much resources also adds too much noise, consequentially decreasing the classification performance (ATTC is outperformed by ATT).

Regarding the similarity threshold, for all the used datasets in general, the classification performance results of the enrichment methods are increasing for the similarity threshold values up to some point (approximately 0.1) and then start to decrease slowly. At the similarity threshold of 0.2 all the results almost converge.

In general, the comparison of enrichment techniques and classification algorithms suggested that the best result should be expected when using the combination of ATT enrichment method and NBM classifier with the similarity threshold set to approximately 0.1. Of course, the actual result still depends on the dataset used.

8 CONCLUSIONS

In this paper, we presented methods to enrich short text document representation model with DBpedia knowledge base for improving short text classification performance. We implemented methods for text enrichment with the use of DBpedia Spotlight framework, to identify important concepts from text, to overcome limitations of

BOW representation and to include some rich semantics. We have evaluated four different ways of text enrichment. Our experiments have shown that our approach is robust with the respect to the input source and yields better performance across multiple dataset using wide range of different classifiers. Improvements over the baseline method vary, best improvements were between 6-10%. The performance of classification methods greatly depends on the quality of the annotation produced by DBpedia spotlight since the introduction of noise can hurt the performance.

We introduced the similarity value as an important parameter, which can greatly improve the classification performance which was shown in our experiment. The results confirmed our expectations that the similarity threshold of 0.1 provides the most significant improvements. This confirms that newly created features should not be added if they would introduce too much noise.

In the future, we will focus on improving the similarity measures between texts and resources spotted with annotation tools, for which we can use Wikipedia Link-based Measure instead of tf-idf. Furthermore, the DBpedia ontology with additional datasets can be explored for distance measures and text enrichment with semantics. Additionally, we could analyze space complexity of newly created representation models to compare sparse matrices and their influence on classification accuracy in short text classification tasks.

ACKNOWLEDGEMENT

The authors acknowledge the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0057).

REFERENCES

[1] Rodrigo Agerri, Xabier Artola, Zuhaitz Beloki, German Rigau, and Aitor Soroa. 2014. Big data for Natural Language Processing: A streaming approach. *Knowledge-Based Systems* (nov 2014). https://doi.org/10.1016/j.knosys.2014.11.007

[2] Charu C. Aggarwal and Cheng Xiang Zhai. 2012. *A survey of text classification algorithms*. Vol. 9781461432234. Springer US, 163–222. https://doi.org/10.1007/978-1-4614-3223-4_6

[3] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia - A Crystallization Point for the Web of Data. *Web Semant.* 7, 3 (sep 2009), 154–165. https://doi.org/10.1016/j.websem.2009.07.002

[4] Leo Breiman. 2001. Random Forests. *Mach. Learn.* 45, 1 (Oct. 2001), 5–32. https://doi.org/10.1023/A:1010933404324

[5] Mengen Chen, Xiaoming Jin, and Dou Shen. 2011. Short text classification improved by learning multi-granularity topics. In *IJCAI International Joint Conference on Artificial Intelligence*. 1776–1781. https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-298

[6] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N Mendes. 2013. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems - I-SEMANTICS '13*. ACM Press, New York, New York, USA, 121. https://doi.org/10.1145/2506182.2506198 arXiv:arXiv:1011.1669v3

[7] DBPedia. 2016. DBpedia. (2016). http://dbpedia.org/

[8] Janez Demšar. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* 7 (Dec. 2006), 1–30. http://dl.acm.org/citation.cfm?id=1248547.1248548

[9] Paolo Ferragina and Ugo Scaiella. 2012. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *IEEE Software* 29, 1 (jan 2012), 70–75. https://doi.org/10.1109/MS.2011.122

[10] Evgeniy Gabrilovich and Shaul Markovitch. 2006. Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In *Proceedings of The 21st National Conference on Artificial Intelligence (AAAI)*. AAAI Press, 1301–1306. https://doi.org/10.1.1.66.3456

[11] Younggyun Hahm, Jungyeul Park, Kyungtae Lim, Youngsik Kim, Dosam Hwang, and Key-Sun Choi. 2014. Named Entity Corpus Construction using Wikipedia and DBpedia Ontology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (Eds.). European Language Resources Association (ELRA), Reykjavik, Iceland.

[12] Andreas Hotho, Andreas Nürnberger, and Gerhard Paaß. 2005. A Brief Survey of Text Mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology* 20 (2005), 19–62. https://doi.org/10.1111/j.1365-2621.1978.tb09773.x

[13] Gongde Guo Hui, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. 2003. Using kNN Model-based Approach for Automatic Text. In *In Proc. of ODBASE'03, the 2nd International Conference on Ontologies, Database and Applications of Semantics, LNCS*. 986–996.

[14] Maciej Janik and Krys J. Kochut. 2008. Wikipedia in Action: Ontological Knowledge in Text Categorization. *2008 IEEE International Conference on Semantic Computing* (aug 2008), 268–275. https://doi.org/10.1109/ICSC.2008.53

[15] José Leal. 2013. Using proximity to compute semantic relatedness in RDF graphs. *Computer Science and Information Systems* 10, 4 (2013), 1727–1746. https://doi.org/10.2298/CSIS121130060L

[16] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2014. DBpedia – a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 1 (2014), 1–5.

[17] Andres McCallum and Kamal Nigam. 1998. A Comparison of Event Models for Naive Bayes Text Classification. *AAAI/ICML-98 Workshop on Learning for Text Categorization* (1998), 41–48. https://doi.org/10.1.1.46.1529 arXiv:0-387-31073-8

[18] Pn Mendes and Max Jakob. 2011. DBpedia spotlight: shedding light on the web of documents. *Proceedings of the 7th International Conference on Semantic Systems* (2011), 1–8.

[19] Pablo N Mendes, Max Jakob, and Christian Bizer. 2012. DBpedia: {A} Multilingual Cross-domain Knowledge Base. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012), Istanbul, Turkey, May 23-25, 2012*, Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis (Eds.). European Language Resources Association {(ELRA)}, 1813–1817. http://www.lrec-conf.org/proceedings/lrec2012/summaries/570.html

[20] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia spotlight. In *Proceedings of the 7th International Conference on Semantic Systems - I-Semantics '11*. ACM Press, New York, New York, USA, 1–8. https://doi.org/10.1145/2063518.2063519

[21] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. 2008. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*. ACM Press, New York, New York, USA, 91. https://doi.org/10.1145/1367497.1367510

[22] Giuseppe Rizzo, Raphaël Troncy, Sebastian Hellmann, and Martin Bruemmer. 2012. NERD meets NIF: Lifting NLP extraction results to the linked data cloud. In *CEUR Workshop Proceedings*, Vol. 937.

[23] Gerard Salton (Ed.). 1971. *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice Hall, Englewood, Cliffs, New Jersey.

[24] Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *Comput. Surveys* 34, 1 (mar 2002), 1–47. https://doi.org/10.1145/505282.505283

[25] Aixin Sun. 2012. Short text classification using very few words. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '12*. ACM Press, New York, New York, USA, 1145. https://doi.org/10.1145/2348283.2348511

[26] Daniele Vitale, Paolo Ferragina, and Ugo Scaiella. 2012. Classification of short texts by deploying topical annotations. *Lecture Notes in Computer Science* 7224 LNCS (2012), 376–387. https://doi.org/10.1007/978-3-642-28997-2_32

[27] Pu Wang and Carlotta Domeniconi. 2008. Building semantic kernels for text classification using wikipedia. *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 08* (2008), 713. https://doi.org/10.1145/1401890.1401976

[28] Xiang Wang, Ruhua Chen, Yan Jia, and Bin Zhou. 2013. Short Text Classification Using Wikipedia Concept Based Document Representation. In *2013 International Conference on Information Technology and Applications*. IEEE, 471–474. https://doi.org/10.1109/ITA.2013.114

[29] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems* 14, 1 (jan 2008), 1–37. https://doi.org/10.1007/s10115-007-0114-2

[30] Lili Yang, Chunping Li, Qiang Ding, and Li Li. 2013. Combining Lexical and Semantic Features for Short Text Classification. *Procedia Computer Science* 22 (2013), 78–86. https://doi.org/10.1016/j.procs.2013.09.083

[31] Jiali Yun, Liping Jing, Jian Yu, and Houkuan Huang. 2012. A multi-layer text classification framework based on two-level representation model. *Expert Systems with Applications* 39, 2 (feb 2012), 2035–2046. https://doi.org/10.1016/j.eswa.2011.08.027

[32] Ying Zhu, Li Li, and Le Luo. 2013. Learning to classify short text with topic model and external knowledge. *Lecture Notes in Computer Science* 8041 LNAI (2013), 493–503. https://doi.org/10.1007/978-3-642-39787-5-41