Categorization of B2B Service Offers: Lessons learnt from the Silex Use case

Molka Tounsi Dhouib[1 2] Catherine Faron Zucker[1] Andrea Tettamanzi[1]

[1] Université Côte d'Azur, Inria, CNRS, I3S, Sophia Antipolis, France
[2] Silex France
{dhouib, faron, tettamanzi}@i3s.unice.fr

Résumé

Dans le domaine de la recherche d'information et du traitement automatique du langage, la tâche de classification de textes est devenue une tâche cruciale. Dans cet article, nous partageons notre expérience de la classification de textes dans un contexte industriel et présentons une évaluation comparative de différents algorithmes de classification binaire et multi-label appliqués à des textes décrivant des offres de services, issus de la plateforme B2B SILEX pour la recommandation de prestataires de services. Nous montrons que dans certains cas pratiques comme celui que nous considérons, une représentation des données sous la forme de "bags of words" donne de meilleurs résultats de classification qu'une représentation réputée plus prometteuse par "word embeddings".

Mots Clef

Catégorisation de textes, Apprentissage automatique, Sac de mots, Plongement lexical

Abstract

In the domain of Information Retrieval and Natural Language Processing, text classification has become a crucial task. In this article, we share our experience of text categorization in an industrial context and we present a comparative evaluation of binary and multi-label classification algorithms applied to texts describing service offers, in the SILEX B2B platform. We show that for some use cases like the one we consider, a traditional representation of texts by "bags of words" gives better classification results than the promising representation by "word embeddings".

Keywords

Text categorization, Machine Learning, Bag of words, Word embedding

1 Introduction

The Silex France company offers a SaaS sourcing tool for identification of the service providers that are best suited to meet the service requests expressed by companies. The Silex platform is used by more than 5000 professionals to quickly identify and exchange with B2B service providers. Silex's ultimate aim is to provide a network of qualified companies to service buyers combined with the best functionality to consult it.

In the framework of a collaborative project between Silex and the I3S research laboratory, we aim at introducing semantics into the B2B platform in order to enable automatic reasoning on service requests and offers and improve the recommendation of service providers.

As a first step, we are interested in automatically categorizing the textual description of companies, service requests and service offers. With data's exponential growth, text categorization has become a crucial issue in Natural Language Processing (NLP). It consists in classifying texts according to their contents, into one or more predefined categories [2]. In recent years, the number of machine learning (ML) techniques that automatically generate text categorization has increased considerably [3].

In this paper we report some experiments we conducted to answer the Silex use case, using supervised ML techniques to classify Silex textual data into predefined categories. Meanwhile, we addressed the following questions:
(1) What is the best representation of Silex textual descriptions of service offers and requests to categorize it?
(2) What is the best ML algorithm to categorize Silex textual descriptions of service offers and requests?

This paper is organized as follows: Section 2 presents state-of-the-art text categorization methods. Section 3 introduces our approach of text categorization to answer the SILEX use case. Section 4 reports and discusses the results of our experiments of categorization of NL descriptions of service requests and offers. Section 5 draws some conclusions and directions for future work.

2 Related Works

Text categorization is the task of classifying data into a predefined set of categories. In other words, given a set of categories and a set of textual documents, text categorization is the process of automatically finding the correct category for each document [4].

2.1 Feature Vector Models

In the text categorization process, the first step consists in preprocessing textual documents, to convert them into feature vectors, a representation that can be automatically interpreted by machine. This step includes tokenization,

stemming and removing of stop words before the creation of feature vectors.

Bag-Of-Words (BOW). is the most common feature vector model. In this model, the features are the frequencies of each word in the textual document. The feature space's dimension is the number of all different words in all documents [4]. The limitation of this model is that it ignores the semantic relations between words.

Word Embedding (WE). In order to overcome the weakness of BOW, a new model called Word Embedding (WE) has been successfully used in several NLP tasks. Word embeddings are projections in a continuous space of words that preserve the semantic and syntactic similarities between them [6]. There are many models that can produce a word embedding. In the following, we introduce the most important ones.
[7] proposes Word2Vec, a popular tool that produces word embeddings based on two models: *CBOW* and *Skip-gram*. *CBOW* is a Neural Network and log-linear model. It removes the non-linear hidden layer, and projects the contextual words on the same position. Word's prediction is obtained according to its past and future contexts. The process consists in computing the average of the contextual word vectors and running a log-linear classifier on the averaged vector.
Skip-gram is similar to *CBOW* and is also a Neural Network and log-linear model. Contrary to *CBOW*, *Skip-gram* predicts the contextual words given the current word [6].
Another model was introduced by [8] and is called *GloVE*. This model is based on global matrix factorization that calculates the co-occurrence of words in the corpus [6].
Finally, *FastText* [1] is a library for learning word representations and sentence classification. FastText published pre-trained word vectors for 294 languages, trained on Wikipedia. These vectors with dimension 300 were obtained using the skip-gram model [13].

2.2 Binary classification

After the construction of feature vectors, the second step of text categorization is the learning step. We present here some of the many learning algorithms for binary classification and multi-label classification.

Naive Bayes (NB) [9]. is a supervised, probabilistic algorithm based on Bayes theorem and the hypothesis of the independence between features. This algorithm is powerful due to the independence between features that ignores features' order, and consequently, the presence of a feature does not affect other features in classification tasks [14].

Support Vector Machine (SVM) [10]. is one of the most common and successful supervised classification algorithm used for text classification tasks [14]. This algorithm transforms training data into higher dimensions and searches for linear optimal separating hyperplane [1].

[1] https://github.com/facebookresearch/fastText

Neural Networks (NN). The NN is composed by many layers to perform text categorization. The perceptron is the simplest kind of a neural network and it has only two layers: the input nodes receive the feature values, the output nodes produce the categorization status values, and the link weights represent dependence relations. The NN text categorization process starts by loading feature weights into the input nodes; the categorization's final result is present in the output nodes after the propagation of the activation of the nodes forward through the network. The neural networks are trained by back propagation in order to minimize the error. In case of misclassification, the error is propagated back through the network and modifies the link weights [15].

2.3 Multi-label classification

Multi-label classification is an approach to classification problems that allows each data point to be assigned to more than one class at the same time. There are two multi-label classification methods: (i) problem transformation and (ii) algorithm adaptation [11]. Here we describe the most popular algorithms of each category.

Problem transformation. A first approach consists in transforming the multi-label problem into one or more single label classification problem. Among the methods adopting this approach, we can highlight:

- **The Binary Relevance method (BR)** is based on one-vs-all ensemble approach. BR transforms any multi-label problem into binary problems to predict the relevance of each label to a data point. All binary classifiers are then aggregated to form a set of relevant labels [11].
- **The label PowerSet method (LP)** considers the multi-label problem as a single multi-class classification problem. In order to create a transformed multi-class dataset, each combination of relevant labels is mapped to a class [11].
- **The Random K-label set (RAKEL)** aims at finding a better balance between BR and LP. RAKEL generates a series of label subsets and builds a label powerset model for each of them [11].

Algorithm adaptation. Algorithm adaptation extends specific algorithms to carry out multi-label classification.

- **Multi-label lazy algorithm (ML-KNN)** is one of the most famous adaptation algorithms. MLKNN uses the maximum a posteriori principle in addition to the K-nearest neighbour algorithm [11].
- **Multi-label decision tree (ML-DT)** is an extension of C4.5 decision tree algorithm. ML-DT allows the creation of multiple labels in the leaves, and chooses node splits based on a pre-defined muti-label entropy function [11].

- **AdaBoost** is based on the addition of simple classifiers to a pool and the use of their weights to define the final classification [16].

3 Experimental methodology

To answer the Silex use case, we conducted some experiments to compare the performance of the above described state-of-the-art feature vector models and learning algorithms for the categorization of textual descriptions of service offers or requests.

3.1 Dataset

Silex distributes a service to purchasing departments that support the sourcing system within the company. Silex users can describe their company (company description), their needs (service request description) and the description of the offers provided by the company (service offer description) in Silex web application. All descriptions are written in French.

The dataset considered in our experiments comprises 3188 company descriptions, 580 service offers descriptions and 155 service requests descriptions.

Six main categories are defined by Silex to classify these descriptions. One description can be categorized in one or more classes. The categorization phase is done manually by experts in the current version of the Silex platform. Table 1 shows the distribution of all the descriptions into the different categories. We observe that the distribution of our dataset is unbalanced, for instance the *Informatique* (Information Technology) class has 915 more companies descriptions than the *Services_industriels* (Industrial Services) class.

Categories	Companies descriptions	service offer descriptions	service request descriptions
Informatique	1043	147	42
Finance	393	40	13
Services généraux	929	149	65
Marketing	926	160	18
Ressources humaines	297	57	7
Services industriels	128	32	19

Table 1: Distribution of the textual descriptions in the corpus.

3.2 Implementation

In our implementation, we focused first on two methods to construct our feature vectors:

- The BOW representation is achieved by applying a traditional process of tokenization, stop words removal and lemmatization. Then we use the TFIDF to construct the matrix. This matrix will be used in different algorithms.

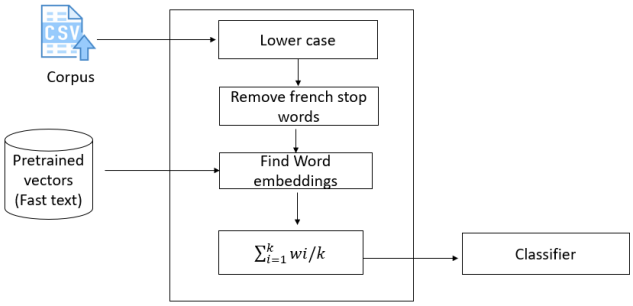


Figure 1: Building WE feature vectors representing the Silex textual descriptions from pre-trained word embeddings

- There are two methods to obtain Word embeddings representation: (i) build word embeddings from scratch using a model like word2vec or FastText. This approach needs a very large corpus to train these word embeddings; and (ii) use a pre-trained word embeddings in any desired language. For our case, we used the latter approach to obtain word embeddings because we do not have a large enough dataset. After studying the different pre-trained word embeddings, we decided to use the FastText model because it is the only model that provides French word embeddings pre-trained on a Wikipedia dump. This word embedding model [2] contains 1,152,449 tokens, which are mapped to vectors in a space with 300 dimensions.

  Our approach to produce the feature vectors representing the SILEX textual descriptions is inspired by [12] and [17]: As illustrated in Figure 1, the first step is to convert textual descriptions to lower case and remove all French stop words. Then the words are checked against the pretrained French vector. If the word exists in the dictionary, the vector representation of description is constructed by averaging the word embeddings vectors along each dimension for all the words in the description. If the dictionary does not contains the word, the zero vector is returned:

$$DescriptionWordEmbeddings = \sum_{i=1}^{k} \frac{w_i}{k}$$

  where k is the number of words in a description and $w_i \in \mathbb{R}^n$ denotes the word embeddings vector of the ith word. The vector representation of each description has exactly the same dimensions as the word embeddings vector w_i [17]. The vector representations of each description are then used as features and input for each classifier.

[2] https://fasttext.cc/docs/en/pretrained-vectors.html

In order to test our feature vectors, we trained two methods of categorization: (i) three binary classification algorithms (NB, SVC, RNN) and (ii) three multi-label classification algorithms: two of them are problem transformation methods (Binary Relevance, Label PowerSet) and three of them are algorithm adaptation methods (Multi-label lazy algorithm (ML-KNN)). All these algorithms are implemented using scikit-learn[3], and executed on a machine with 8 GB of memory and an Intel Core i7-7500U CPU.
To deal with the unbalanced data, there are two main state-of-the-art approaches to adjust the distribution of dataset: (i) over-sampling adds copies of instances from the under-represented class. The most common technique of over-sampling is called Synthetic Minority Over-sampling Technique (SMOTE), (ii) under-sampling deletes instances from the over-represented class. In our implementation, we tested first a very large parameter with a grid function in scikit-learn to identify the best parameters for each classifier and the best method to deal with unbalanced data. We obtained the best results with SMOTE method for SVM algorithm, and RandomOverSampler method for NB. Table 2 shows all parameters used in binary classification and Table 3 shows all parameters used in multi-label classification.

Classifier	Unbalanced data	Parameters
NB	RandomOverSampler	
SGDClassifier	SMOTE	penalty=l2 alpha=0.0001
LinearSVC	SMOTE	C=0.1 penalty='l2' loss='squared hinge'
SVC	SMOTE	kernel='rbf' C=1000.0 gamma=0.0001
RNN	balanced	hidden size=32 Activation('relu') Activation('sigmoid')

Table 2: Parameters used for binary classification

Classifier	multi-label methods	Parameters
LSVC	Binary Relevance	C=1, penalty='l2'
LSVC	Label Powerset	C=1, penalty='l2
MLkNN	Algorithm adaptation	-

Table 3: Parameters used for multi-label classification

4 Experiments and discussion

4.1 Evaluation procedure

The experiments were conducted using the 5-fold cross-validation methodology. To evaluate our experiments, we calculated the classification duration and various evaluation measures inspired by [18] and described in the following.

[3]http://scikit-learn.org/stable/

Binary algorithms. To evaluate binary classification algorithms, we used the **F1 score** measure, defined as the harmonic mean between precision and recall:

$$F1score = \frac{1}{N}\sum_{i=1}^{N}\frac{2\times|h(xi)\cap yi|}{|h(xi)+yi|}$$

Multi-label algorithms. To evaluate multi_labels algorithms we used the following metrics:

- **Accuracy** allows to compute the percentage of correctly predicted labels among all predicted and true labels. It is defined as follows:

$$Accuracy(h) = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{h(xi)\cap yi}{h(xi)\cup yi}\right|$$

- **Micro-precision** is defined as the precision averaged over all the example/label pairs:

$$Micro_{precision} = \frac{\sum_{j=1}^{Q} tpj}{\sum_{j=1}^{Q} tpj+\sum_{j=1}^{Q} fpj}$$

  where tpj, fpj are the number of true positives and false positive for label λ_j .

- **Micro-recall** is defined as recall averaged over all the example/label pairs:

$$Micro_{recall} = \frac{\sum_{j=1}^{Q} tpj}{\sum_{j=1}^{Q} tpj+\sum_{j=1}^{Q} fnj}$$

  where fnj is the number of false negatives for the label λ_j.

- **Micro-F1** is the harmonic mean between micro-precision and micro-recall and is defined as:

$$Micro - F1 = \frac{2\times micro_{precision}\times micro_{recall}}{micro_{precision}+micro_{recall}}$$

- **Hamming loss** allows to evaluate how many times an example-label pair is misclassified. The performance is perfect when the value of this metric is equal to 0. Hamming loss is defined as

$$Hamming_loss(h) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{Q}|h(xi)\Delta yi|$$

  where Δ is the symetric difference between two sets, N is the number of examples and Q is the total number of possible class labels.

4.2 Results and Analysis

The results of binary classification algorithms reported in Table 4, Table 5 and Table 6 show that there is not a significant difference between the results with a BOW representation and the results with a word embeddings representation. For example, for category "Informatique" , we obtain the same precision value that equals to 0.82. The recall value equals to 0.82 for BOW representation, and is a bit better compared to WE recall value that equals to 0.79. The classification processing time is much better for WE (0.48 seconds) than for BOW (1572 seconds). This is due to the BOW expansive phases of lemmatization and matrix building compared to the vector representation of word embeddings.

Table 4: Best classifier of binary classification of companies descriptions

Category	Method	P	R	F1	Time
Informatique	LSVC + WE	**0.82**	0.79	0.80	0.48
	SGD+BOW	**0.82**	**0.82**	**0.82**	1572
Finance	LSVC+WE	0.90	**0.87**	**0.88**	0.21
	NB+BOW	**0.92**	**0.87**	**0.88**	1793
Services généraux	LSVC+ WE	0.87	0.84	0.85	0.4
	NB + BOW	**0.90**	**0.89**	**0.89**	1925
Marketing	LSVC+WE	0.82	0.80	0.81	0.41
	SGD + BOW	**0.85**	**0.85**	**0.85**	1517
Ressources humaines	LSVC + WE	0.91	0.83	0.86	0.19
	NB + BOW	**0.92**	**0.84**	**0.87**	1904
Services industriels	LSVC+WE	0.94	0.89	0.91	0.05
	NB+BOW	**0.97**	**0.94**	**0.95**	1914

Table 5: Best classifier of binary classification of service offer descriptions

Category	Method	P	R	F1	Time
Informatique	RNN+WE	**0.80**	**0.78**	**0.79**	8.53
	RNN +BOW	0.78	**0.78**	0.78	332
Finance	SVC-RBF+ WE	0.82	0.87	0.84	0.083
	SGD + BOW	**0.91**	**0.90**	**0.86**	323
Services généraux	RNN+ WE	**0.82**	**0.81**	**0.81**	6.97
	RNN+BOW	0.79	0.79	0.79	348
Marketing	SGD+WE	0.80	0.77	0.78	0.028
	NB + BOW	**0.90**	**0.90**	**0.90**	459
Ressources humaines	RNN + WE	**0.90**	**0.91**	**0.90**	7.42
	NB + BOW	0.87	0.87	0.87	419
Services industriels	SVC-RBF+ WE	0.91	0.92	0.91	0.09
	SVC-RBF+BOW	**0.96**	**0.97**	**0.96**	347

We can draw similar conclusions when comparing multi_label classification algorithms, with a slight advantage to BOW compared to word embeddings in terms of accuracy and hamming_loss but not in term of time.
If we compare binary classification and multi_label classification algorithms, we can see that the results of the first alogrithm are better than the second one. These results are as expected because first we don't have a very large corpus, and second when we use a binary classification we can play on the classifier parameters for each category to enhance its performance, which is not the case for multi_label classification. The different binary classification algorithms perform more or less well in categorizing our dataset using the same representation. For example, only the LSVC algorithm categorized correctly this text as "informatique" *"Ascot accompagne les dirigeants et cadres de PME dans la définition, la maitrise et le suivi de leur transformation numérique. Plus aucune organisation ne peut ignorer l'impact global sur son métier, ses outils et ses processus. Notre expérience, de plus de 20 ans, en Conseil, Assistance technique, Développement, Ingénierie, ... est entièrement dédié à nos clients. Sous contrat ou à la carte, nos prestations leur apportent valeur et sérénité."*.

Table 6: Best classifier of binary classification of service request descriptions

Category	Method	P	R	F1	Time
Informatique	NB +WE	**0.82**	**0.77**	**0.79**	0.006
	NB+ BOW	0.79	0.74	0.76	101.31
Finance	RNN +WE	**0.97**	0.48	0.62	2.44
	LSVC + BOW	0.94	**0.94**	**0.92**	84.82
Services généraux	NB+ WE	**0.74**	**0.74**	**0.74**	0.006
	NB + BOW	**0.74**	**0.74**	**0.74**	103.49
Marketing	RNN + WE	0.73	0.77	0.74	2.74
	NB +BOW	**0.89**	**0.81**	**0.84**	104.10
Ressources Humaines	NB+ WE	0.88	0.94	0.90	0.006
	BOW	-	-	-	-
Services industriels	RNN + WE	0.91	0.90	0.88	2.62
	LSVC+BOW	**0.94**	**0.94**	**0.94**	79.07

Table 7: Best classifier of multi_label classifications.

Classifier	Feature	Micro-F1	Accuracy	Hamming loss	Time
MLKNN	WE	0.66	0.53	0.12	7.00
	BOW	0.72	0.57	0.10	1945
BR	WE	0.61	0.35	0.17	6.54
	BOW	0.73	0.52	0.10	1850
Label Powerser	WE	0.54	0.39	0.19	9.44
	BOW	0.69	0.51	0.13	1945

We obtained also different results when comparing the BOW and WE representations. For example, using the same LinearSVC algorithm, this text *"Agence WebMarketing 360 de nouvelle génération exploitant de nombreux leviers marketing pour optimiser vos campagnes digitales & booster votre chiffre d'affaires Kalipseo développe pour vous les stratégies digitales les plus pertinentes et s'emploie à promouvoir l'image de votre marque."* was wrongly categorized as "Informatique" when using the WE representation, but the categorization was right when using BOW representation.

As a conclusion, to best answer the Silex use case, we decided to use a binary classification algorithm with a word embedding representation. Table 8 shows the algorithms that are selected to classify each category of Silex data.

Table 8: Selected algorithms for Silex Company

Category	Description	Representation	Algorithm
Informatique	Companies	WE	NB
	Service offer	WE	RNN
	Service provider	WE	NB
Finance	Companies	WE	RNN
	Service offer	WE	SVC-RBF
	Service provider	WE	RNN
Services généraux	Companies	WE	NB
	Service offer	WE	RNN
	Service provider	WE	NB
Marketing	Companies	WE	RNN
	Service offer	WE	SGD
	Service provider	WE	RNN
Ressources Humaines	Companies	WE	NB
	Service offer	WE	RNN
	Service provider	WE	NB
Services industriels	Companies	WE	RNN
	Service offer	WE	SVC-RBF
	Service provider	WE	RNN

5 Conclusion

In this paper we reported the results of the experimental evaluations of various vector feature models and machine learning algorithms conducted to answer the real-world use case of the Silex company: how to categorize textual descriptions of service offers and requests with the ultimate goal of recommending service providers that better answer services requests. We compared the two main state-of-the-art feature vector models. A BOW representation is a bit better than a WE representation regarding the evaluation measures. This is due to the use of a generic pre-trained vector for WE descriptions, that is based on Wikipedia dump and that does not cover well our data set. On the other hand, a WE representation has the advantage of being less time consuming because of the additional lemmatization and matrix building phases specific to the BOW representation. Based on these results, we choose to use classifiers with word embeddings data for Silex company as shown in Table 8.

As future work, we aim to define a specific B2B pre-trained vector that best covers the Silex dataset instead of using a general reference like Wikipedia. We also aim to study a combined approach based on both word embeddings and knowledge engineering approaches. The objective is to improve the representation of texts by introducing semantics to increase the performance of algorithms. To achieve this, we are going to use our built ontologies to represent the sourcing domain, and we are going to enrich the original texts with additional information based on semantics relations between concepts such as generalization or specification.

6 Aknowledgement

This work is supported by the ANRT, CIFRE I3S-SILEX 2016/0947

References

[1] Patra, Anuradha and Singh, Divaka A survey report on text classification with different term weighing methods and comparison between classification algorithms *International Journal of Computer Applications*, volume.75, pp.7,2013,Foundation of Computer Science.

[2] Sadiq, Ahmed T and Abdullah, Sura Mahmood, Hybrid intelligent technique for text categorization *Advanced Computer Science Applications and Technologies (ACSAT), 2012 International Conference on*, pp.238-245,2012,IEEE.

[3] Tan, Ah-Hwee and Lai, Fon-Lin, Text categorization, supervised learning, and domain knowledge integration *To appear, proceedings, KDD-2000 International Workshop on Text Mining, Boston*, Vol.20,2000.

[4] Feldman, Ronen and Sanger, James,The text mining handbook: advanced approaches in analyzing unstructured data,2007,Cambridge university press.

[5] Sriram, Bharath and Fuhry, Dave and Demir, Engin and Ferhatosmanoglu, Hakan and Demirbas, Murat, Short text classification in twitter to improve information filtering, Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval,pp.841–842, 2010, ACM

[6] Ghannay, Sahar and Favre, Benoit and Esteve, Yannick and Camelin, Nathalie, Word Embedding Evaluation and Combination,LREC,2016

[7] Mikolov, Tomas and Chen, Kai and Corrado, Greg and Dean, Jeffrey,Efficient estimation of word representations in vector space,arXiv preprint arXiv:1301.3781, 2013

[8] Pennington, Jeffrey and Socher, Richard and Manning, Christopher,Glove: Global vectors for word representation,Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP),pp.1532-1543,2014

[9] Schneider, Karl-Michael, Techniques for improving the performance of naive bayes for text classification, Computational Linguistics and Intelligent Text Processing, pp.682-693, 2005,Springer

[10] Cortes, Corinna and Vapnik, Vladimir,Support-vector networks, Machine learning, Vol.20,3,pp.273-297,1995,Springer

[11] Pakrashi, Arjun and Greene, Derek and MacNamee, Brian,Benchmarking Multi-label Classification Algorithms,24th Irish Conference on Artificial Intelligence and Cognitive Science (AICS'16), Dublin, Ireland, 20-21 September 2016,2016, CEUR Workshop Proceedings

[12] Bayot, Roy and Gonçalves, Teresa,Author Profiling using SVMs and Word Embedding AveragesâĂŤNotebook for PAN at CLEF 2016,2016,CEUR

[13] Bojanowski, Piotr and Grave, Edouard and Joulin, Armand and Mikolov, Tomas,Enriching Word Vectors with Subword Information, arXiv preprint arXiv:1607.04606,2016

[14] Mohare, Punam Kishor Itkar, A, A Survey on Different Types of Approaches to Text Categorization,V5,nternational Journal of Emerging Trends & Technology in Computer Science

[15] Bassil, Youssef, A Survey on Information Retrieval, Text Categorization, and Web Crawling,rXiv preprint arXiv:1212.2065,2012

[16] Benbouzid, Djalel and Busa-Fekete, Róbert and Casagrande, Norman and Collin, François-David and Kégl, Balázs,MultiBoost: a multi-purpose boosting package,V.13,pp.549-553,Journal of Machine Learning Research,2012

[17] Yang, Xiao and Macdonald, Craig and Ounis, Iadh,Using word embeddings in twitter election classification,arXiv preprint arXiv:1606.07006,2016

[18] El Kafrawy, Passent and Mausad, Amr and Esmail, Heba,Experimental comparison of methods for multi-label classification in different application domains,V.114, N.19,Foundation of Computer Science, International Journal of Computer Applications,2015

[19] Sokolova, Marina and Japkowicz, Nathalie and Szpakowicz, Stan,Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation,V.4304, pp.1015-1021,Australian conference on artificial intelligence,2006