Proceedings of the International Conference on
Information Technologies (InfoTech-2018)
20-21 September 2018, Bulgaria

MODELLING AND IMPLEMENTATION OF MACHINE LEARNING TECHNIQUES FOR HATE SPEECH DETECTION IN MOBILE APPLICATIONS

Bujar Raufi, Ildi Xhaferri

South East European University, Canadian Institute of Technology
e-mails: b.raufi@seeu.edu.mk, ildi.xhaferri@cit.edu.al
Republic of Macedonia, Republic of Albania

Abstract: The proliferation of data through various platforms and applications is in constant increase. The versatility of data and its omnipresence makes it very hard to detect the trustworthiness and intention of the source. This is very evident in dynamic environments such as mobile applications. As a result, designing mobile applications that will monitor, control and block any type of malintents is important. This paper makes an attempt in this direction by implementing a lightweight machine learning classification scheme for hate speech detection in Albanian Language for mobile applications. Initial testing and evaluations indicate good classifier accuracy in mobile environments where frequent and realtime training of the algorithm is required.

Key words: automatic hate speech detection, machine learning, artificial neural networks (ANNs)

1. INTRODUCTION

We live in a ‘ubiquitous’ world of the 5th generation of mobile technology. It is an uncontested truth that the world will become more and more mobile-oriented and the huge amounts of data and traffic produced by all of these devices is constantly challenging developer’s ways of thinking. The Big Data on the other hand has already overwhelmed every aspect of our lives. Having these two directions in mind, the model proposed in this paper explains and integrate different state-of-the-art technologies in order to be able to cope with the huge amount of data, meanwhile delivering excellent fast services.

This paper attempts an approach towards building a mobile application in the android platform to let users cast into vote their opinions about public issues and concerns. Although there are many voting/polling system offered online, most of them are based on or are imitation of social networking like applications [1], [2]. In this direction, an application has been considered, where the content is all-publicly available, but the ability to interact with it (or some part of it) is conditioned by some predefined rules and constraints based on the nature of the application itself. The application offers an easy user-to-user interaction and opportunity to integrate new features in further developments.

The main objective of paper is to introduce a relatively new way of integrating different technologies and platforms, software tools and libraries, to build a mobile application that use machine learning techniques in the backend for hate speech detection for Albanian social network users. The application itself represents a novelty of this kind in Albanian software market. It can be further extended with functionalities to fully offer users the ability to vote or count votes in a transparent way since all the data will be open to anyone. Having in mind the society we are contributing in is not considered as an open and transparent one and often decision making is not consulted with the general public. As a result, this application can be of help on that direction. On the other hand, transition to a new language (or a new target group) is easily since the methodology and engineering part is adhered to the modularity principle which makes it easy seen from reusability standpoint. We consider that the biggest beneficiaries from this work would be the new software engineers and programmers in the mobile development industry (especially in Albanian-speaking countries).

The rest of this paper is organized as follows: section 2 introduces current development related to machine learning approaches related to mobile software solutions, section 3 outlines the system architecture proposed in this paper, section 4 evaluated the classifier model used in this application and section 5 concludes the paper outlining future directions of development.

2. RELATED WORK

Choosing the right platform for first deployment of the application is sometimes a challenging task. Many aspects should be considered such as the type of application, look and feel, user interaction facility, changing content, network speed, optimization etc. As an integral part of the development process, mobile UI design is also very important in the creation of mobile apps. Mobile user interaction (UI) should consider limitation and other contexts factors such as screen, inputs, and mobility outlines for design. Mobile UIs, or front-ends, depend on mobile back-ends to support access to enterprise systems. The mobile back-end make possible data routing, authentication, authorization, working off-line, and services management. These services are supported by a combination of middleware components such as mobile application servers, mobile backend as a service (MBaaS), or service-oriented architecture (SOA) infrastructure.

In our application, we further consider the extension with text classification as an important part because we want the system to discard the text that contains offensive words or hate speech. To be able to achieve the above mentioned, the system should distinguish between hate speech and a legitimate comment. In general, automatic text classification comes in three flavors: pattern-matching, algorithms, Artificial Neural Networks (ANNs) [3]. Pattern matching is kind of a 'brute-force' approach where various word patterns are matched against pre-determined patterns and classified accordingly. The approach requires a constant expert system authoring and update of the "bag of patterns" which renders the approach unmanageable at large scale. Algorithmic approach is relatively good but it degrades with the increase of data variability. For our case, we considered the artificial neural networks as an optimal choice since it offers the best combination of precision and classification time, and it seems to be the most appropriate classifier for this purpose [4].

Neural network based methods have obtained a great progress on a variety of natural language processing tasks. The primary role of the neural models in text classification is to represent variable-length text as a fixed-length vector. These models generally consist of a projection layer that maps words, subwords units or n-grams to vector representations and then combine them with the different architectures of neural networks. There are several kinds of models to represent text such as Neural Bag-of-Words (NBOW) model, recurrent neural network [5], [6], recursive neural network [7], [8] and convolutional neural network [9], [10], [11]. These models take as input the embedding of words in the text sequence, and summarize its meaning with a fixed length vectorial representation.

Recent advancement in machine learning algorithms (and of course in hardware processing speed) have made this a very active research field, both in academia and industry. Continuous advancement is being made, such as Hierarchal Attention Network [12], topic labeling [13], [14], sentiment classification [15], [16], spam detection [17], [18].

In text mining and classification, the selection of the methods is made according to individual case of application. As for now, it is impossible to define the best text classifier that fits all profiles. In research fields such as computer vision, there is a strong consensus about generic approaches of designing models such as deep learning and deep neural networks with lots of residual connections. In contrary to that, text classification is still far from convergence on some narrow area [18]. Currently, text classification it is widely used in sentiment analysis (IMDB, YELP reviews classification), stock market, Google's smart email replay etc. This field is depended on the natural language under study because it is tightly linked to Natural Language Processing. However, to the best of our knowledge, the applicability of machine learning algorithms in mobile devices is yet to come and new methods and approaches are needed due to the versatility of environments and data at hand. The paper attempts a small contribution in this direction.

3. REQUIREMENTS ANALYSIS AND SYSTEM ARCHITECTURE

The mobile platform for the application is chosen due to the nature, the content as well as the context of use. Since it is a forum like application, the focus is on the user's opinion sharing and feedback, so the availability and mobility will be always important. The approach followed on this project was to allow user to post a polling questions, and detect whether that question is offensive and contains hate speech. Based on that, the system will classify them accordingly and recommend the admin whether such publication should be allowed or not. For this purpose, the machine learning techniques are used for detecting and classifying posts accordingly. The requirements for the application should fulfill the following basic requirements:

- The application should be completely reliable
- The application should be maintainable
- The application should perform hate speech detection of more than 85%
- The application should allow fast opinion posting and hate speech detection
- The application should use artificial intelligence
- The application should be totally modular
- The application should be flexible towards scalability

The general process of hate speech detection during voting is depicted as in fig. 1

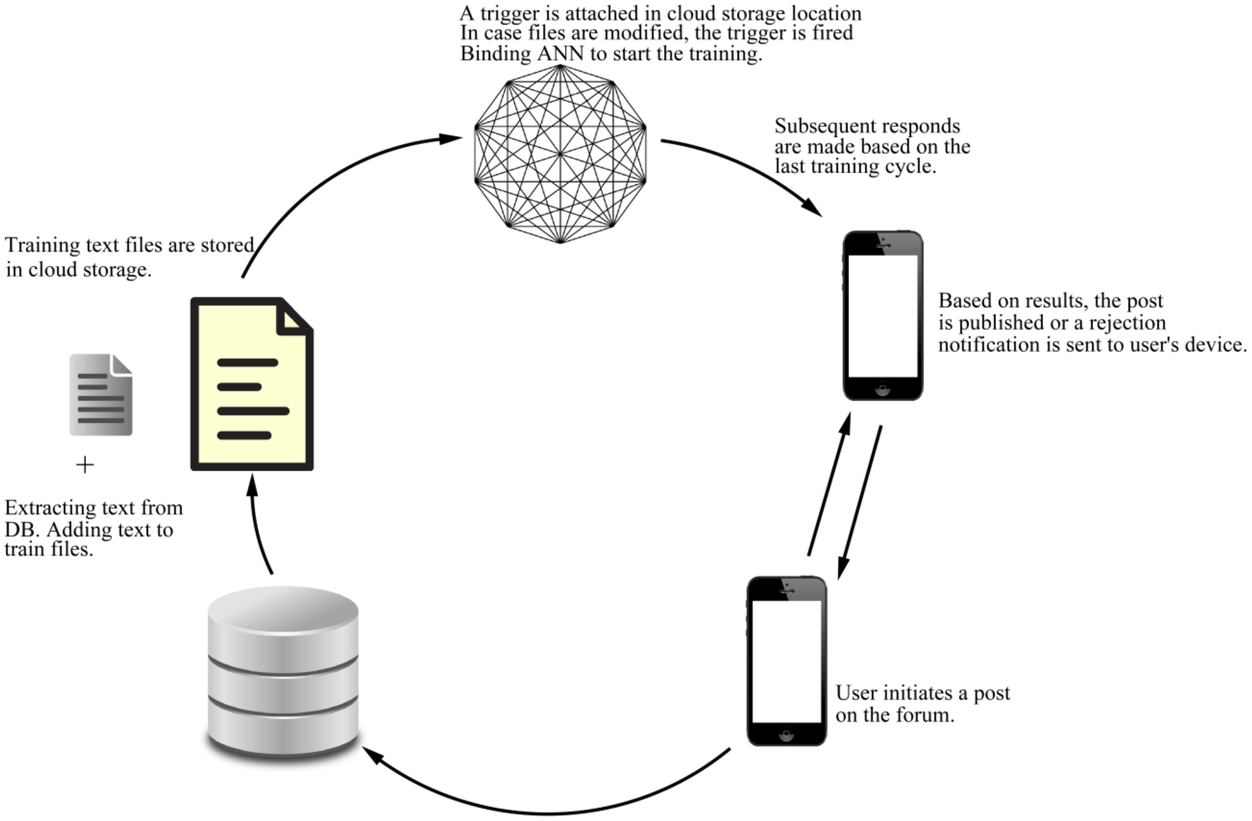
A trigger is attached in cloud storage location
In case files are modified, the trigger is fired
Binding ANN to start the training.
Subsequent responds
are made based on the
last training cycle.
Based on results, the post
is published or a rejection
notification is sent to user's device.
User initiates a post
on the forum.
Extracting text from
DB. Adding text to
train files.
+
Training text files are stored
in cloud storage.

Fig. 1. The process of hate speech detection during opinion voting

The process starts by user submitting a comment on the forum. The comment is assigned a pending status and is registered on the database. The extraction process from the database starts by removing stop words and stemming the text which on the next phase is fed into ANN trained model which classifies the words as a regular text or offensive and this notifies the admin whether the post should be allowed or not. It is worth mentioning that in every subsequent posts, the ANN model uses the newly classified words as a repository for future classifications, thus increasing the classifier accuracy.

Artificial Neural Network (ANN) techniques are used to help distinguish text-containing offensive or hate speech. Hate speech generally refers to expressions, speech, gesture or writing that advocates, threatens, or encourages violent acts. Popular social media websites, blogs, forums are frequently being misused by many groups to promote online radicalization or any sort violence.

The overall process follows a generic text classification strategy [3] as depicted in fig. 2

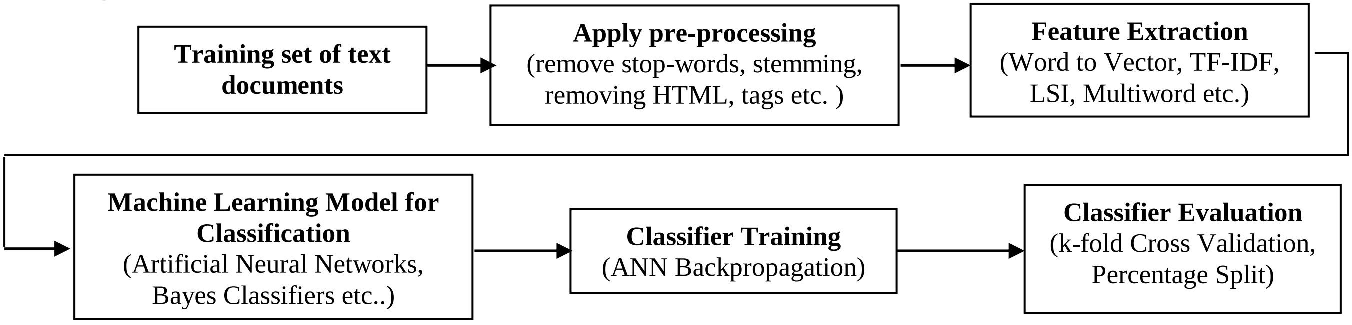
Training set of text documents
Apply pre-processing
(remove stop-words, stemming, removing HTML, tags etc.)
Feature Extraction
(Word to Vector, TF-IDF, LSI, Multiword etc.)
Machine Learning Model for Classification
(Artificial Neural Networks, Bayes Classifiers etc..)
Classifier Training
(ANN Backpropagation)
Classifier Evaluation
(k-fold Cross Validation, Percentage Split)

Fig. 2. Generic Strategy for Text Classification

Neural network initial training is done using multiple text files respectively for each class, OFFENSIVE and NORMAL. In one text file group, collected text is considered ‘normal’. In the other file group, text is considered of offensive language, hate speech and/or bad words. Each sentence is broken down by word (stemmed) and each word becomes an input for the ANN. The source used in preparing these text files is quite random, using social network statuses, group discussions, comments etc. This first pre-classified text is not so decisive, since other will be added to it later with the increased usage of the application. This will serve as the initial feed for the ANN in order to return some acceptable results starting from the beginning.

The files grouped as NORMAL consisted of approximately 300 posts collected from local Albanian forums resulting in 3,620 collected words. The files collected as OFFENSIVE consisted of 421 posts with a total of 6,648 words. The final file resulted in an initial training set consisted of 590 generated attrıbutes that are the words kepts for training. The complete list is given in table 1.

Table 1 Collected training set for hate speech detection

Forums	*Class*	*Posts*	*Words*
Jeta osh Qef	*NORMAL* *OFFENSIVE*	*220* *380*	*2.654* *6,000*
Xing me Ermalin	*NORMAL* *OFFENSIVE*	*80* *41*	*965* *648*

3. HATE SPEECH DETECTION MODEL EVALUATION

Considering that the initial training sample collected from forums is small in sample we used sampling methods to increase the sample size. For this purpose we used two sampling approaches: simple resample and SMOTE. Simple resample produces a random subsample of a dataset using either sampling with replacement or without replacement. SMOTE uses a dataset resampling technique by applying the Synthetic Minority Oversampling TEchnique. SMOTE represents a more natural and class imbalanced representations which are important for text classification processes and the results indicated a good training for the ANNs.

We use a Neural Network library written in JavaScript called natural-synaptic [20]. Natural-Synaptic is a natural language classifier library for Node Natural using a Synaptic neural network. Synaptic library includes a few pre-built network architectures like multilayer perceptrons and Hopfields nets. Type of the network used is a feed-forward network with backpropagation architecture. The perceptron architecture of this library give us the possibility to create a multilayer neural networks. This consist of a collection of layers, each fully connected to the next one. To classify a sentence as being offensive or normal each word serves as an input of the first layer of the network. We have to provide a minimum of three layers (input, hidden and output), but we can use as many hidden layers as we wish. In our case, we use the default: 50 inner layers and it has converged relatively quickly with an error no bigger than 0.003. Table 2 depicts the initial training conditions for the Artificial Neural Networks(ANN) used.

Table 2 Initial setting values for ANN training

Training Settings	***Values***
ANN Learning rate	*0.4*
Momentum	*0.2*
# of hidden layers	*50*
Error rate	*0.001*
Training epochs	*1000*

The initial training settings, indicated a proper sensitivity from the sense that training the model revealed Mean absolute error of 0.003 and Root mean squared error of 0.0037. Fig. 3 indicates the sensitivity of the convergence from the perspective of the overall standard deviation during classification. From the chart it can be seen overlays between both training values (OFFENSIVE and NORMAL) in relation to class. These class and training overlays deviate by a margin as indicated in the error rates defined during the training phase as well as Mean absolute error and Root mean squared error.

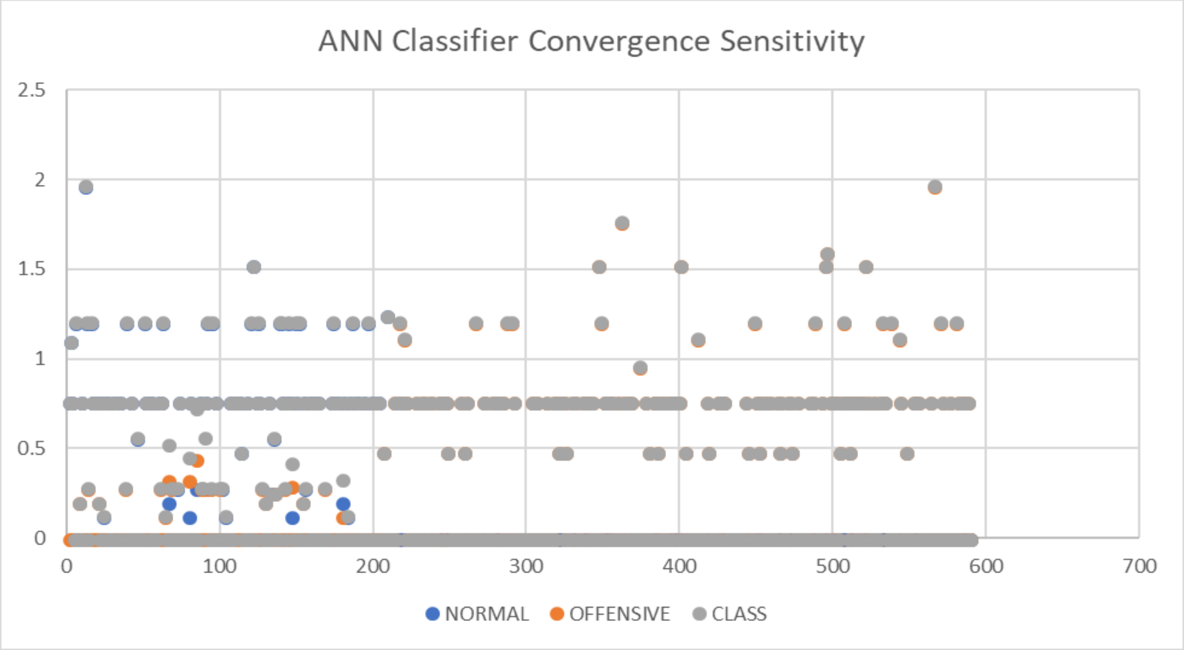

Figure 3 Convergence sensitivity of the training

The training process used 10-fold cross validation (CV) as well as 60/30 percentage split and was evaluated against three runs applied for classical resampling and SMOTE. Cross-validation (CV) is a technique to evaluate predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it. In 10-fold cross-validation, the original sample is randomly partitioned into 10 equal size subsamples. The percentage split method splits the dataset in two parts 60% for training and the rest 30% for testing. The chart in fig. 4 depicts the ANN classifier accuracy for all three runs compared to Resample and SMOTE against both 10-fold cross validation and 30/60 percentage splits training mechanisms.

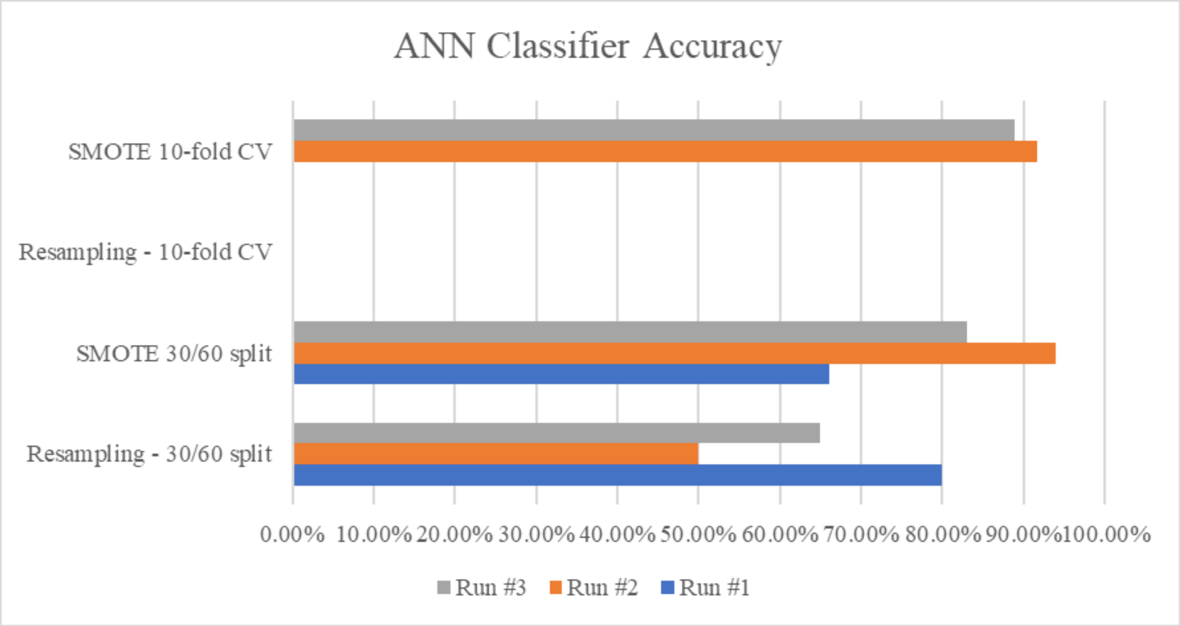

Figure 4 ANN Classifier Accuracy

The results indicate that for the case of Synthetic Minority Oversampling TEchnique (SMOTE) the ANN show good accuracy on the second run of oversampling and there is no need for further runs since the oversampling process deteriorates the classifier accuracy. The classical resampling on the other hand shows insufficiencies from the perspective of lack of samples for folding process. The 10-fold cross validation method splits the dataset into 10 groups and iterates ten times by taking each group as training and testing sets accordingly. Due to small number of posts and consequently words, the resampling process did not yield the necessary results.

4. CONCLUSION AND FUTURE WORK

The aim of this paper was to introduce a mobile development approach and an integration of mobile applications with machine learning algorithms in real time mobile environments. The final objective was to achieve hate speech and offensive language detection and prevention. The results indicated high level of classifier accuracy as well as applicability of the machine learning algorithms in mobile environments. Likewise, the use of machine learning in this kind of text analytics is highly suggested because the need to quickly handle requests/responses is high in the mobile ‘world’. The neural network algorithms have been proven to produce very good results even though we have tested it in a simple feed-forward network library.

The future of work would involve the following two aspects:

- Generating a greater training dataset by using NLP techniques for specific word analysis (syntactic and lexical word dependencies etc.) of Albanian language. This would be beneficial since it would produce a more rigid and a well defined training set and it would reduce the need for resampling.
- Using Deep Learning techniques for detecting even hidden types of offensive hate speech detection. The current system is developed on “per word" based detection where deeper language constructs are not in scope of our analysis.

REFERENCES

[1] Brugge, T, W. Lautenschlager, H. Orlamunder. (**2005**) Method for performing a voting by mobile terminals. U.S. Patent Application 10/963,819.

[2] Barhydt, W., S. Bhalla, M. Cartabiano, J. Hardy, J. Su, W. Chan (**2008**). U.S. Patent Application No. 11/696,711.

[3] Dalal, M. K., M.A. Zaveri, M. A. (**2011**). Automatic text classification: a technical review. International Journal of Computer Applications, 28(2), 37-40.

[4] Prasanna, P. L., D.R. Rao. (**2018**). Text classification using artificial neural networks. International Journal of Engineering & Technology, 7(1.1), 603-606.

[5] Pengfei Liu, Xipeng Qiu, Xuanjing Huang. (**2016**). Recurrent Neural Network for Text Classification with Multi-Task Learning. Fifth International Joint Conference on Artificial Intelligence.

[6] Chung, J. Hierarchical multiscale recurrent neural networks. (**2016**). International Conference on Learning Representations (ICLR'17). arXiv preprint arXiv:1609.01704.

[7] Socher, R., C. Lin, C. Manning, C., A.Y. Ng. (**2011**). Parsing natural scenes and natural language with recursive neural networks. In Proceedings of the 28th international conference on machine learning (ICML-11) (pp. 129-136).

[8] Luong, T., R. Socher, C.,Manning. (**2013**). Better word representations with recursive neural networks for morphology. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning (pp. 104-113).

[9] Kim, Y. (**2014**). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[10] Grefenstette, E., P. Blunsom, (**2014**). A Convolutional Neural Network for Modelling Sentences. ACL.

[11] Kalchbrenner, N., E. Grefenstette, P. Blunsom. (**2014**). A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188.

[12] Yang, Z., D. Yang, D., C. Dyer, X., He, X., A. Smola, E. Hovy. (**2016**). Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 1480-1489).

[13] Lau, J. H., D. Newman, S. Karimi, S., T. Baldwin. (**2010**). Best topic word selection for topic labelling. In Proceedings of the 23rd International Conference on Computational Linguistics: Posters (pp. 605-613). Association for Computational Linguistics.

[14] Wan, X., T. Wang. (**2016**). Automatic labeling of topic models using text summaries. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (Vol. 1, pp. 2297-2305).

[15] Severyn, A., A. Moschitti. (**2015**). Unitn: Training deep convolutional neural network for twitter sentiment classification. In Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015) (pp. 464-469).

[16] Tang, D., F. Wei, N. Yang, M. Zhou, T. Liu, T., B. Qin, B. (**2014**). Learning sentiment-specific word embedding for twitter sentiment classification. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (Vol. 1, pp. 1555-1565).

[17] Starbuck, B. T., R. L. Rounthwaite, D.E. Heckerman, J.T. Goodman, E.C. Gillum, N. D. Howell, K.R. Aldinger. (**2016**). U.S. Patent No. 9,305,079. Washington, DC: U.S. Patent and Trademark Office.

[18] Crawford, M., T.M. Khoshgoftaar, J. D. Prusa, A.N. Richter, H, Al Najada. (**2015**). Survey of review spam detection using machine learning techniques. Journal of Big Data, 2(1), 23.

[19] Trusov, R. (**2017**). Text Classifier Algorithms in Machine Learning. *Stats and Bots*, May (*available at:* https://blog.statsbot.co/text-classifier-algorithms-in-machine-learning-acc115293278).

[20] Gardideh N, Dengerfield, M. (**2017**) natural synaptic: A natural language classifier for Node Natural using a Synaptic neural network. (*Available at:* https://github.com/nemo/natural-synaptic)