Comput Syst Sci & Eng (2020) 6: 467–475


International Journal of
Computer Systems
Science & Engineering

# Text Classification for Azerbaijani Language Using Machine Learning

**Umid Suleymanov[1], Behnam Kiani Kalejahi[1,2,*], Elkhan Amrahov[1], Rashid Badirkhanli[1]**

[1] *Department of Computer Sciences, School of Science and Engineering Khazar University, Baku, Azerbaijan*
[2] *Department of Biomedical Engineering, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran*

Text classification systems will help to solve the text clustering problem in the Azerbaijani language. There are some text-classification applications for foreign languages, but we tried to build a newly developed system to solve this problem for the Azerbaijani language. Firstly, we tried to find out potential practice areas. The system will be useful in a lot of areas. It will be mostly used in news feed categorization. News websites can automatically categorize news into classes such as sports, business, education, science, etc. The system is also used in sentiment analysis for product reviews. For example, the company shares a photo of a new product on Facebook and the company receives a thousand comments for new products. The systems classify comments like positive or negative. The system can also be applied in recommended systems, spam filtering, etc. Various machine learning techniques such as Naive Bayes, SVM, Multi-layer Perceptron have been devised to solve the text classification problem in Azerbaijani language.

Keywords: Text classification; embedding system; SVM; neural network.

## 1. INTRODUCTION

### 1.1 Definition

Text classification is the task of automatically assigning one of the predefined labels to a paragraph or article. More formally, if some Di is a document in the entire set of documents D and {c1, c2, c3, …, cn} is the set of all the categories, the text classification assigns one category $C_j$ to a document $D_i$. In our project, each article belongs to only one category. And when a document can only belong to one category, it is called single-label and if the opposite is true we call this multi-label. A single-label text classification task is also divided further into a binary class and multi-class classification when the document is assigned to n mutually exclusive classes (Yaying Qiu and et al, 2010). Text classification can help us divide up documents conceptually and has many important applications in the real world.

For example, news articles are often categorized by subject categories; academic papers are typically organized by technical domains, and so on (Yaying Qiu and et al, 2010). On the other hand, spam filtering is the widespread application of text classification. In this application of text classification, email messages are decided to be either spam or non-spam. The incoming email is automatically categorized based on its content. Language detection, analysis, and intent are based on supervised systems. Email routing and sentiment analysis are also another application of text classification. Labeled data is deployed to the machine learning algorithm and the algorithm gives the desired predefined categories. In text classification labeled data is used as training data to derive a classification system and then it can automatically classify unlabeled text data using the derived model. Most of the data is collected from the web, especially news websites for training our system.

### 1.2 Purpose

As the number of digital data that is in Azerbaijani is increasing day-by-day, there is a growing need for classifying and categorizing such data. Especially, in the news sector, readers face such articles that they do not want to read. Assigning data to some classes can be a feasible solution to this problem. Therefore, text classification based on the topic could solve such kind of issues. Several scholarly articles and surveys have been

*Corresponding Author

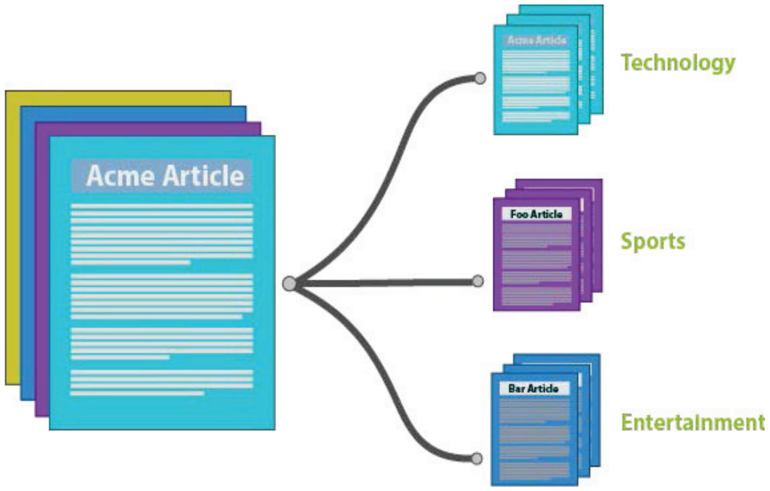


**Figure 1** Text Classification in different domains.

studied during the process of research. The main target was practical systems and surveys conducted on the area of text categorization.

As the technology develops, it causes an increase in the amount of resources which is in text format. They can be online documents, articles, or social media correspondence on the web in Azerbaijani language. There was a need for some way of analyzing and classifying the given data for the company, organizations, and individuals. And text classification can be used to solve this categorization problem for the Azerbaijani language. The developed text classification system will help to solve the text classification problems in the Azerbaijani language. There are some text-classification applications for foreign languages, but we tried to build a newly developed system to solve this problem for the Azerbaijani language. Firstly, we tried to search and find out potential practice areas that need this type of system to be applied to. The system will be useful in a lot of areas and in the future, it is expected to be used in all text related areas after the digitization process which leads to making electronic versions of handwritten documents. It will be mostly used in the news feed categorization process. News websites can automatically categorize news into classes that were defined beforehand such as sport, business, education, science, etc. The system can also be adjusted to be used in sentiment analysis for product reviews. For example, the company shares a photo of a new product on Facebook and it receives a thousand comments for the new product. The systems classify comments like positive or negative. In this way, it can help Azerbaijani companies to increase customer care, find out their weakness and build development plans in terms of correcting their mistakes. Overall, our purpose for creating text classification in Azerbaijani language is to help news websites, organizations, and companies easily categorize or classify their data.

### 1.3 Problem Statement

For the project, to get a highly accurate results, many machine learning algorithms need to be applied to the project. After applying the supervised learning algorithms to the project, it needs to be compared and taken the most suitable and efficient one. Each algorithm will have its advantages. Selecting a suitable algorithm for the project does not solely determine the outcome. The text representation models and text pre-processing options also have a substantial impact on the results. When there is no prior information, BOW architecture is frequently used for text representation.

Finding the right categories which will be most appropriate for the articles is a very difficult problem. There are some conjugations between two, or even three categories which will affect the result and will decrease the preciseness of the found label. That is why, it is desirable to merge categories which are too close into one category. On the other hand, stop words are another problem in the determination of the right category.

Another problem that affects the percentage of finding the right output will be the data which will be used as training for the algorithms. This data was collected from different websites and each website defines their categories. One website categorizes news differently than another. Of course, training this kind of data will negatively affect the project output. Therefore, the collected data needs to be reviewed, analyzed, and corrected.

## 2. LITERATURE REVIEW

There is a variety of research on text classification.Naive Bayes is used pervasively for its speed and simple architecture (Schneider, 2005). Naive Bayes classifiers utilize Bayes rule as its foundation. We can approach these problems and show that they can be solved by some simple modification. Modification can be like feature engineering, exploiting the language's lexical structure, and semantic relations using morphological resources. Some of these techniques have already been used (McCallum & Nigam, 2017). Support Vector Machines are also widely utilized in text classification problems as Naive Bayes does and they are as a supervised machine learning algorithm (Joachims, 2006). The paper demonstrates the relative advantages of applying Support Vector Machines. It has high dimensional input space and few irrelevant features. For each text, the text vector consists of few entries which are not zero. Another advantage is that Support Vector Machine is a robust algorithm. Support Vector

**Table 1** Sentence distribution and descriptive statistics.

| Mean sentence count | 17.461187 |
|---|---|
| std | 19.988552 |
| min | 0.000000 |
| 25% percentile | 8.000000 |
| 50% percentile | 12.00000 |
| 75% percentile | 22.00000 |
| Max | 1915.000 |

**Table 2** Character distribution and descriptive statistics.

| Mean character count | 1466.51758 |
|---|---|
| std | 1781.99540 |
| min | 1.00000000 |
| 25% percentile | 629.000000 |
| 50% percentile | 1037.00000 |
| 75% percentile | 1693.00000 |
| Max | 186679.000 |

Machines suggests a new justification for good performance and consequently allows us to construct learning algorithms that generalize well by robustifying non-consistent algorithms. Thus, it has good results in text classification. Artificial neural networks have been widely applied to text categorization (Renato & Teresa, 2004). Multilayer Perceptron and Decision tree algorithms are also applicable for text categorization. The paper describes the experiment of a decision tree algorithm for text categorization. The decision tree algorithm is widely used in text classification. The algorithm works on a tree structure where the internal node is labeled by the term, branches represent weight and leaves represent the class. After performing experiments with decision tree algorithms in text classification, it turns out that decision trees are capable of learning disjunctive expressions. However, it has some disadvantages such that it will not always return the globally optimal decision tree.

# 3. DESIGN CONCEPTS

## 3.1 Description of Solutions/Approaches

As text classification is a widely encountered problem in machine learning, a lot of research has been done in this area. The text classification process is a composite process that includes pre-processing the data, training, and tuning the model and at the end predicting the label of the given document from the predefined set of labels. Therefore, the accuracy of the final prediction depends not only on the model but also on problem definition and data pre-processing. For preparing the data, it is common to assign a unique number to all the words in the vocabulary and represent each document as zeros and ones where one in the given position means the document has the word in the vocabulary in exactly that position. As this representation is more easy and efficient for the computer to process, a lot of researchers use this representation for text classification. This representation is also called Bags of Words. As not all words are equally important in determining the category of the document, researchers generally use Term Frequency Times Inverse Document Frequency. Moreover, before processing the data, removing stop words, and combining stem words make the calculations more efficient and accurate. Determining most of the hyper parameters and data preprocessing such as stop word removal are language-specific problems. Therefore, doing text classification for the Azerbaijani language requires a lot of novel ideas to achieve the desired accuracy. For example, the data set used for training the classifier is from Azeri news sites. The successful implementation of the classifier depends heavily on the data at hand. Therefore, data should be cleaned and normalized before processing which requires deep investigation of data and getting valuable insights from it. Cleaning and normalizing Azerbaijani news data is a novel problem that requires novel approaches to solve. For example, different news sites divide their articles into different categories. As a result, the news data have a lot of categories some of which are very similar to each other. Therefore, by analyzing the data and categories we tried to lessen the number of categories by merging, re-assigning categories.

## 3.2 Word Embedding

During image processing tasks, high-dimensional, encoded vector representations of the individual raw pixel-intensities of images are used for training machine-learning models. (Daniel Vasic, Emil Brajkovic, 2018) However, text classification techniques traditionally approach words as atomic symbols, and therefore ‘mother’ may be represented as id136 and ‘father’ as id345. These representations provide no useful information to the system regarding the interconnection of the words. Representing words as ids cause the inclusion of many zeros. Using word embedding can contribute to eliminating above mentioned problems.

### 3.2.1 Dataset Creation

There are 1082844 news reports in the overall data. Dataset Statistics results are for 25 February 2019. Initially, 301224

**Table 3** Sentence distribution and descriptive statistics.

| Mean sentence count | 16.502746 |
|---|---|
| std | 12.091614 |
| min | 4.000000 |
| 25% percentile | 8.000000 |
| 50% percentile | 13.00000 |
| 75% percentile | 22.00000 |
| Max | 99.00000 |

**Table 4** Character distribution and descriptive statistics.

| Mean character count | 1382.421532 |
|---|---|
| std | 1162.050153 |
| min | 59.0000000 |
| 25% percentile | 649.000000 |
| 50% percentile | 1050.00000 |
| 75% percentile | 1687.00000 |
| Max | 999.000000 |

**Table 5** Sentence distribution and descriptive statistics:

| | |
|---|---|
| Mean sentence count | 12.868643 |
| std | 10.661528 |
| min | 0.000000 |
| 25% percentile | 7.000000 |
| 50% percentile | 10.00000 |
| 75% percentile | 15.00000 |
| Max | 99.00000 |

duplicate news dropped and then 781636 news remained in the overall data. Dataset statistics before the first phase cleaning is shown below.

First Phase Cleaning involved the following: removing all news containing less than 30 characters; removing all news containing more than 10000 characters; removing all news containing less than 3 sentences; removing all news containing more than 100 sentences. 753011 news articles remained after the first phase of cleaning. The below tables summarize the dataset statistics after the first phase of cleaning. The number of all sentences (raw count) in all news articles was 12426749.

The number of all characters in all news articles was 1040978620. Character distribution and descriptive statistics are illustrated above.

### 3.2.2 Applying Regex

A lot of JavaScript codes were observed inside the content of the news articles. These codes carry no information regarding the statistical distribution of words in sentences and therefore are meaningless for generating word embedding. Moreover, web addresses are used as reference links in some cases which also if kept can deteriorate the quality of word embedding generated at the end. Therefore, regular expressions have been implemented for getting them out of the dataset and increasing the quality of sentences. The number of news articles is 752939. The number of all sentences (dot count) in all news articles after the Regex application is 9689303. Sentence distribution and descriptive statistics:

Character distribution (including whitespaces) and descriptive statistics mean character count: 1299.217275 and the standard deviation is 1141.380408. The number of all characters in all news articles is 978231356. The number of words (not necessarily correct words) in all news articles is 126863549.

The histogram above shows the distribution of sentences in the dataset. The vertical axis shows the number of news articles in the dataset and the horizontal axis shows the number of sentences in articles. For example, as you can see in the histogram 40000 news articles contain 12 sentences. Briefly, the histogram represents most of the articles have 5–16 sentences.

## 4. RESEARCH METHODOLOGY AND TECHNIQUES

An in-depth analysis of parameters and weights of classifiers is an essential part of the research. These parameters and weights give further insights into the classification problems. This enables us to make reasonable decisions and increase the performance results of the classifiers incrementally. The techniques used for the analysis of classifiers are as follows:

Analyzing precision, f1, and other metrics of the classifier gives a lot of guidance on where the classifier suffers and how it can be fixed. Besides these metrics, there is another metrics called confusion matrix which determines how the classifier performs on the test data. More specifically it shows the number of articles our classifier classifies correctly for each category as

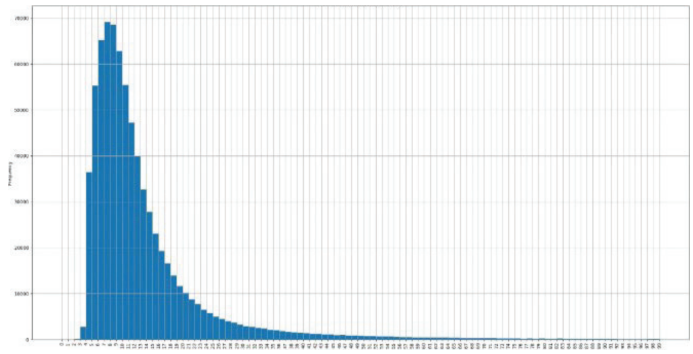

**Figure 2** Word distribution in all news articles.

```
[[584   2   7   2   7   4]
 [  4 128   1   0   2   2]
 [  7   0 684   1  13  12]
 [  1   0   1  67   5   9]
 [  7   6  16   3 365  29]
 [  9   1  12   4  11 620]]
```

**Figure 3** Confusion Matrix of SVM classifier for the merged data set.

well as the number of articles it confuses with each one of the other categories. Fig. 3 displays an instance of a confusion matrix that was used for analysis purposes.

## 4.1 Architecture, Model, Diagram description

After data preprocessing, we began researching and using supervised machine learning approaches to our project so that we can optimize the prediction results. The research on the text classification was not a linear process from data preprocessing to building models and optimizing them. Rather, it was an iterative process, namely after developing the models we were analyzing the weights and coefficients to further develop our understanding of the structure of the data and to optimize the performance results of the classifier. After all these steps, the model is ready to label real-world documents on its own.

The project consists of two parts. The main part of the project is intended to train a model using cleaned and categorized data and use it to classify input data in the second part of the project which is the web.

The image below illustrates an approach to the classification problems. Firstly, having enough data is the most essential factor in text classification problems. It is not straightforward to find clean, sanitized data for the specific problem you are solving. Therefore, you need to do some pre-processing on your data before introducing it to the classifier. You need to clear and relabel it if necessary. The data that we are using for classification has been collected from Azerbaijani news websites. The next steps are about working with data that is ready.

## 4.2 Data Loading

Unlike other data sources, CSV and Excel files can easily be loaded and processed. The data we utilize to train the model consists of 6 columns. After the loading process, 10% of the data is kept for testing and the rest 90% is passed to the classifier. Starting from the main part of the project, until now, different approaches have been applied, tested, and evaluated.

## 4.3 Bag of Words

Bags of words (BOW) are a set of various words that a document contains. The basic idea is to take any document and count the frequency of words. Based on the values of frequency, we calculate probability. The outcome of the BOW is a bunch of tuples and integers. The way to interpret the first row of the outcome is that the word number 131607 appears only once in the first document.

## 4.4 TF-IDF Vectorizer

Term Frequency-Inverse Document Frequency (Tf-IDF) Vectorizer is equivalent to Count Vectorizer plus Tf- IDF Transformer and expresses the importance of a word in the document. By using Tf-IDF Vectorizer, we can easily generate a list of the influential words for each class (category). Although Count Vectorizer is more powerful than a simple Binary Vectorizer, it has some limitations. Count Vectorizer just counts the frequency of words showing up in a document without considering the rareness or commonness of words. There is a more advanced concept, Tf-IDF, which not only calculates the frequency of words, it also takes the inverse document frequency into account. The process happens in two steps: The first one is about finding "TF" which is the probabilistic frequency of a word in the given document. The second one is intended to find "IDF".

$$\text{TF ("IT", D1)} = 3/7 = 0.43$$

$$\text{Tf("it", D2)} = 3/6 = 0.5$$

$$\text{IDF("it", D)} = \log(2/2\ ) = 0$$

$$\text{Tf-IDF ("it", D1)} = \text{Tf("it", D1)} \times \text{IDF("it", D)} = 0.43 \times 0 = 0$$

$$\text{Tf-IDF ("it", D2)} = \text{Tf("it", D1)} \times \text{IDF("it", D)} = 0.5 \times 0 = 0$$

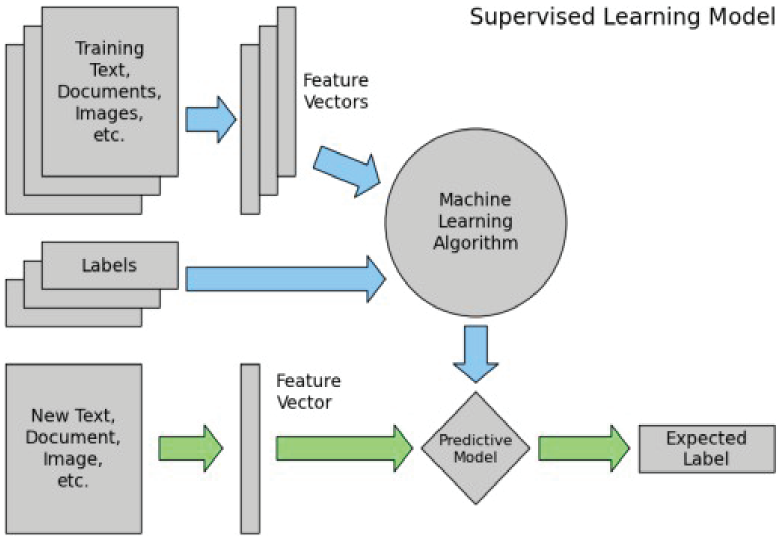


**Figure 4** Supervised Learning Architecture in Text Classification [7].

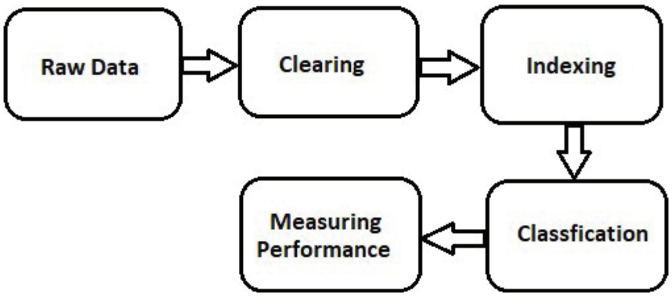


**Figure 5** Approach to classification problems.

```
(0, 131607)   1
(0, 116442)   1
(0, 145501)   1
(0, 194856)   1
(0, 73423)    1
(0, 28620)    1
(0, 180959)   1
(0, 194936)   1
(0, 69305)    1
(0, 24572)    1
(0, 172916)   2
```

**Figure 6** BOW representation.

**Table 6** Document 1 (D1).

| Term | Count |
|---|---|
| lion | 2 |
| it | 3 |
| forest | 1 |
| man | 1 |

which implies that the word “it” is not so influential in the corpus. We can go further to calculate the Tf-IDF of each word.

Thereby, an IDF value of the word which occurs across multiple documents will be low, and it will affect Tf-IDF value. Low Tf-IDF value of a word denotes that the word is less informative. So, the TF-IDF vector does not only contain term frequencies as count based vectorizer does, but also involves IDF values. Even though Naive Bayes classifier is powerful enough and shows satisfactory performance, it has weaknesses and it is the best approach for text classification. The first disadvantage of the Naive Bayes approach is data scarcity. At these moments, we would end up with zero while calculating probability. Generally, there is no such rule that Naive Bayes is weaker than the Support Vector Machine (SVM). It completely

**Table 7** Document 2 (D2).

| Term | Count |
|---|---|
| cat | 1 |
| it | 3 |
| live | 1 |
| house | 1 |

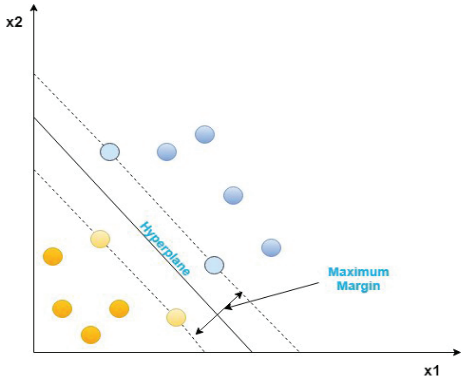


**Figure 7** Categorization.

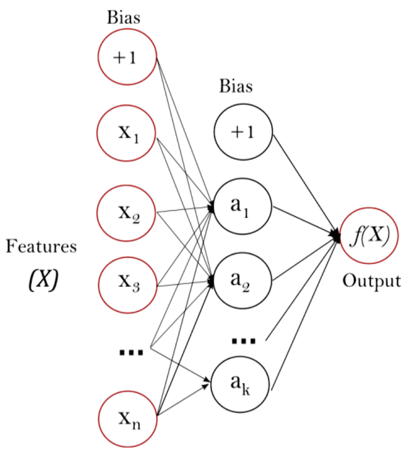


**Figure 8** Multi-layer Perceptron.

depends on the size of the dataset, predefined categories, and how training data is organized.

## 4.5 Naive Bayes

From the algorithmic point of view, there are several techniques to solve the current issue. The basic one is Naive Bayes which is functioning based on Bayes rule. The Naive Bayes classifier estimates the probability of new data by using the given training data. So far, as a team, we have implemented and tested this approach. The outcome appeared unsatisfactory as expected because of the working principle of Naive Bayes. Two similar words varying with a single character are perceived as two distinct strings by Naive Bayes classifier. For the next stage, we are planning to use the Support Vector Machine (SVM) for the classification of texts. The SVM integrates both dimension reduction and classification. However, it is only relevant for binary classification tasks. While using SVM, we can reduce the computational power and storage complexities by dividing the training set into small parts and representing each as support vectors. A more advanced method is a Neural Network in which each unit will represent a single word from the training set. Neural Network produces a score rather than a probability.

Besides the algorithm, clear data has a quite high significance to achieve the desired accuracy. Therefore, before deciding on the algorithms, we aimed at clearing the current data and try to minimize the numbers of categories. Fewer numbers of categories mean the classifier is less prone to make a mistake. Moreover, even the best algorithms cannot perform well on wrongly labeled training data.

## 4.6 Support Vector Machine

SVM is also applied as a machine learning technique in text categorization tasks. It is only suitable for binary classification tasks which mean text classification must be treated as a series of separate categorization problems (Tong & Koller, 2001). At the training stage of Support Vector Machine, documents from two distinct categories are taken and SVM maps all the documents to high-dimensional space. Then, the algorithm attempts to find out a separator line which is also called hyperplane or model. This is between mapped points of two categories, making sure that margin is as high as possible (Joachims, 2006).

By implementing the Support Vector Machine, we have been able to increase the accuracy from 56.53% (Naive Bayes Classifier) to 93%. Then, to get better results than we achieved with SVM, we moved to implement another supervised machine learning algorithm, Artificial Neural Networks.

### 4.7 Neural Network - Multi-Layer Perceptron

Multi-layer Perceptron is a supervised machine learning technique.

The successful approach was to use Multi-Layer Perceptron with a "lbfgs" solver. The accuracy we achieved with Neural Network was better than Naive Bayes' outcome, however, for our dataset, SVM performs better than Neural Network implementation. The left-most layer is called the input layer and is composed of neurons that are input features. The right-most neuron is our actual output which is the classification result, category of the input document.

## 5. CONCLUSION AND FUTURE WORK

As predicted the Tf-IDF Vectorizer performed better than the count based vectorizer because Tf-IDF Vectorizer also considers the importance of a word in the document by using Tf-IDF Transformer. As discussed, the Naive Bayes classifier is our initial and baseline model. The accuracy was approximately 58%, however, other research papers conclude that it can achieve more.

Additionally, determining a suitable classifier is as important as data is. After investigating the Naive Bayes approach, we shifted our attention to Support Vector Machine and we got its performance improvements. The Neural Network showed poorer performance than SVM. The scholarly articles present that the Artificial Neural Network is much more powerful than SVM, for text classification problem of our case it cannot illustrate its full power.

Moving from classifier to web & API side of the project, we planned to run the application on the server. First, we tried the Windows machine on Azure to setup the Flask server. However, Windows is not the easiest platform for running the Flask server (Apache server + WSGI module) because the integration of the WSGI module and Apache server was unsuccessful. After some effort, we moved to the Ubuntu machine which performed satisfactorily.

As future work, we are planning to create intrinsic evaluation dataset for embedding. Moreover, some natural language processing tasks such as word stemming, word lemmatization and etc. are still open research questions in Azerbaijani language and the existing approaches needs improvements. The application of more complex neural network models such as LSTMS for the text classification tasks can also be investigated in the future.

## REFERENCES


1. Karl-Michael Schneider, *Techniques for Improving the Performance of Naive Bayes for Text Classification*, International Conference on Intelligent Text Processing and Computational Linguistics CICLing 2005, pp 682–693.
2. N.A. Zaidi, G.I. Webb, M.J. Carman, F. Petitjean, J. Cerquides, ALRn: *accelerated higher-order logistic regression*, Mach. Learn. 104 (2–3) 2016, 151–194.
3. D. Vasic, E. Brajkovic. "Development and Evaluation of Word Embeddings for Morphologically Rich Languages", 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2018
4. F. Zheng, G.I. Webb, P. Suraweera, L. Zhu, *Subsumption resolution: an efficient and effective technique for semi-naive Bayesian learning*, Mach. Learn. 87 (1) 2012, 93–125.
5. D. Bahdanau, Cho K., Y. Bengio *Neural machine translation by jointly learning to align and translate* ICLR 2015.
6. Yaying Qiu, Guangming Yang, Zhenhua Tan. 2010. *Chinese Text Classification Based on Extended Naïve Bayes Model with Weighed Positive Features*, First International Conference on Pervasive Computing, Signal Processing and Applications.
7. Y. Bengio *Practical recommendations for gradient-based training of deep architectures Neural Networks: Tricks of the Trade*, Springer 2012, pp. 437–478.
8. J. Devlin, Chang M., Lee K., K. Toutanova Bert: *pre-training of deep bidirectional transformers for language understanding NAACL-HLT* 2019, pp. 4171–4186.
9. He X., D. *Golub Character-level question answering with attention EMNLP* 2016, pp. 1598–1607.
10. A. Joulin, E. Grave, P. Bojanowski, T. Mikolov *Bag of tricks for efficient text classification EACL* 2017, pp. 427–431.
11. Lei T., R. Barzilay, T. Jaakkola *Rationalizing neural predictions EMNLP* 2016, pp. 107–117.
12. Lin Z., Feng M., C.N.d. Santos, Yu M., Xiang B., Zhou B., Y. Bengio *A structured self-attentive sentence embedding* ICLR 2017.
13. Liu X., Mou L., Cui H., Lu Z., Song S. Jumper: *learning when to make classification decisions in reading* IJCAI 2018, pp. 4237–4243.
14. G. Marcus, Deep learning: *a critical appraisal*, arXiv preprint arXiv:1801.00631, 2018.
15. Mou L., Lu Z., Li H., Jin Z. *Coupling distributed and symbolic execution for natural language queries* ICML 2017, pp. 2518–2526.
16. Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In Proceedings of the 27th International Conference on Computational Linguistics, pages 1638–1649.
17. Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. Journal of Machine Learning Research, 6(Nov):1817–1853.
18. Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In Advances in neural information processing systems, pages 3079–3087.
19. Simon Tong and Daphne Koller, 2001. *Support Vector Machine Active Learning with Applications to Text Classification*
20. Dan Hendricks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. CoRR, abs/1606.08415.
21. Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In IJCAI.
22. Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In Advances in neural information processing systems, pages 3294–3302.

23. Thorsten Joachims (2006). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*
24. Andrew McCallum, Kamal Nigam (1998). *A Comparison of Event Models for Naive Bayes Text Classification.*
25. Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 1631–1642.
26. Fu Sun, Linyang Li, Xipeng Qiu, and Yang Liu. 2018. U-net: Machine reading comprehension with unanswerable questions. arXiv preprint arXiv:1810.06638.
27. Renato Fernandes CorrÍa and Teresa Bernarda Ludermir (2004). *Web Documents Categorization Using Neural Networks*.
28. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems, pages 6000–6010.