Estimating the text preprocessing algorithms for sentence classification

Suleyman Suleymanzade
Institute of Information Technology ANAS, Baku, Azerbaijan
suleyman.suleymanzade@bhos.edu.az

Abstract – **Estimating the text preprocessing algorithms in machine learning classification task helps to select the most accurate combination of word embedding text processing algorithms to map the test data samples to appropriate labels. Increasing the accuracy in classification of text data improves afterwards the response to queries in search engines and gives more relevant results. This work includes several preprocess text data techniques and their combinations for following sentence classification by using support vector machine (SVM).**

I. INTRODUCTION

Text classification is one of the important fields of artificial intelligence that includes Natural language processing, machine learning [1] [2] and deep learning techniques [3] [4] [5]. The automation of the text classification process gives ability to classify and store documents (or metadata with link associated with web document) by topics. In search engines it helps to search document not only by the keywords that exist in the content of document but also by the additional semantic term that text classifier added to the searched web content. Also, the classification helps to store content metadata in the backend in sorted and structured way. The search engine becomes more relevant.

II. PROBLEM STATEMENT

The goal of this work is to use varios text preprocessing methods and their combinations to review and analyzing text data which was gathered *(mined)* from the internet for the further text classification process. The classification in this work is done on the text data. So, because of the labeled data the classification method belongs to a field of supervised learning.

There are many machine-learning and deep-learning methods for a supervised learning. In this work the different representing words techniques will be processed by the SVM (support vector machine) [6] with different kernel functions and other exclusively belonged to SVM’s parameters. The problem of the text data that machine-learning algorithms such as SVM cannot handles this type of data. The only data representation that can be fitted by ML models are the number representation of the model. This type of data representation calls feature-vector. All the preprocess techniques that presented in this work use to transform the mined text data to feature vectors.

III. TEXT PREPROCESSING

Text processing in Natural Language Processing (NLP) represents the mapping process between the word (or phrase, document or even documents’ clusters, depends on the granularity of selected data and task) and numerical value. To produce such data representation the following approaches are used. There are two main strategies to separate word embedding:

- The strategy based on frequency
- The strategy based on predictions.

The prediction-based vectors used to predict the next word. To predict a text data there are popular methods such as continues bag of words (CBOW) [7] and Skip-Gram [8]. The text data that processed by these methods are processed by machine learning or deep neuron networks models. The frequency based methods. In current work, all methods to classify the text data are frequency based.

A. *The Vector Space Model*

The Vector space model [9] [10] also known as traditional method represented as one-hot binary vector. Where vector’s size equals to the total unique documents’ vocabulary. In this binary vector representation: one -

DOI: 10.25045/NCInfoSec.2018.43

shows the index position where certain word existed, other one zeros respectively.

The implementation of this data structure is easy, but it uses huge amount of memory because of redundant zeros. One solution for memory optimization – is to create data structure where instead of very long vector sets of zeros there is special key with number which shows number of zeros between ones. But even such optimization saves memory to store data, but not during the processing where data must be converted again to the one-hot binary vector.

Second problem with such representation of data that it not stores semantic information about stored data. Traditional model not belongs to a frequency or prediction-based models because this representation of the word embedding shows only the existence of text in data in whole corpus and not shows the number of occurrence and sequence.

B. *Count based model*

The count-based model unlike the traditional method stores number of words that appears in corpus the documents. If the number of the unique words appeared in the document is n and the number of the document is d then the matrix size that stores the whole data representation must be $n * d$.

The number of appeared words gives additional information about the word value. As well as in the traditional method the count-based model is easy to program, but the cons of this approach is that not always the word that is repeating more has more values. For example, in English language the articles such as “the”, “a” repeat many times in the docutments but they cannot be considered as determined factor for the classification of document’s text. Considering the sample that given below

C. *TF-IDF*

TF-IDF [11] [12] stand for - *“term frequency - inverse document frequency”*. Unlike count based model, TF-IDF based on idea that if word has appeared in most of the documents, then probably that word is not relevant to a document. For example: “and”, “the”, “or” they appear in most of the text documents, but they don’t give meaning to documents’ semantic.

The relevant term must appear many times in small group of documents from the observed document corpus. This observation can be described mathematically. *TF* – the proportion of term number appearing in one document to the number of words in the particular document. And IDF is the logarithmic proportion of total number of documents to the number of documents where the term appears.

$$TF(t,d) = f(t,d) / \textstyle\sum_{t'd} f_{t',d} \qquad (1)$$

$$IDF(t,D) = \log \frac{N}{|\{d \in D : t \in d|}$$

Example:
There are two documents.

TABLE I. SENTENCE REVIEW

Document 1		**Repeating**
RNN		6
uses		2
To		2
calculate		1
Data		1
Document 2		**Repeating**
RNN		0
uses		2
To		2
calculate		2
Data		1

$$TF(RNN, Document1) = \frac{6}{12} = \frac{1}{2}$$
$$IDF(RNN) = \log\left(\frac{2}{1}\right) = 0.301$$
$$TFIDF(RNN, Document1) = \tfrac{1}{2} * 0.301 = 0.1505$$
$$TF(To, Document1) = \frac{2}{12} = \frac{1}{6}$$
$$IDF(To) = \log\left(\frac{2}{2}\right) = 0$$
$$TFIDF(To, Doument1) = 0$$

IV. MODEL

One of the popular methods in machine-learning to classify the samples on different categories is the support vector machine (SVM) [6]. There are different types of SVM models: linear and non-linear. *Linear* If the samples that must be classified separated linearly then the idea based on the “optimal” boundary of linear hyperplane that separates one set of samples from the other.

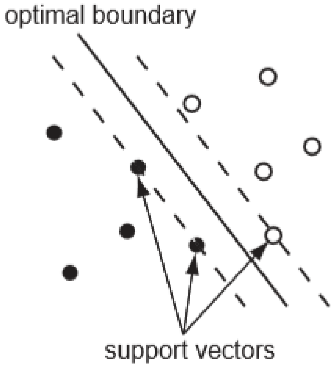

Figure 1 SVM with closest boundary

The vectors with the closest to the boundary they are the support vectors. The boundary of the hyperplane represented by formula.

$$\boldsymbol{w^T x + b = 0} \qquad (2)$$

$\boldsymbol{w}$ – weight coefficient, $\boldsymbol{b}$ – bias. Consider the black dots are the set $\boldsymbol{x_+}$ and the white dots is other set $\boldsymbol{x_-}$ respectively. The length between trained $\boldsymbol{x_i}$ which located on the opposite sites (the margin) is calculated by

$$\boldsymbol{margin} = |\overline{\boldsymbol{x_-}} - \overline{\boldsymbol{x_+}}| \frac{\overline{\boldsymbol{w}}}{||\boldsymbol{w}||} \qquad \textit{(3)}$$

Because the aim is to maximize the margin. The optimization is considered as
$\boldsymbol{||\overline{w}||^2 = w^T w}$ minimize subject to $\boldsymbol{y_i(w^T x_i + b) \geq 1}$,

$$\text{Where} \quad \begin{cases} \boldsymbol{y = 1}, & \boldsymbol{y \in \{x_+\}} \\ \boldsymbol{y = -1}, & \boldsymbol{y \in \{x_-\}} \end{cases} \qquad (4)$$

The optimization provided by Lagrange’s [13] [14] multipliers method.

$$\boldsymbol{L(\alpha_i, b, w) = \frac{1}{2} w^T w - \sum_i \alpha_i [\, y_i(w^T x_i + b) - 1]} \qquad (5)$$

Where $\alpha_i \geq 0$ is the indeterminate coefficient. The partial derivative of w and b is

$$\frac{\delta L}{\delta w} = w - \sum_i \alpha_i y_i x_i$$
$$\frac{\delta L}{\delta b} = - \sum_i \alpha_i y_i$$

By rewriting equation 5

$$\mathrm{L(\alpha_i, b, w)} = \frac{1}{2}\left(\sum_{\mathrm{i}} \mathrm{\alpha_i y_i x_i}\right)^{\mathrm{T}} \left(\sum_{\mathrm{j}} \mathrm{\alpha_j y_j x_j}\right) - \sum_{\mathrm{i}} \mathrm{\alpha_i y_i} \left(\sum_{\mathrm{j}} \mathrm{\alpha_j y_j x_j}\right)^{\mathrm{T}} \mathrm{x_i} + \sum_{\mathrm{i}} \mathrm{\alpha_i}$$
$$= \frac{1}{2}\mathrm{w^T w} - \sum_{\mathrm{i}} \mathrm{\alpha_i y_i w^T x_i} - \mathrm{b} \sum_{\mathrm{i}} \mathrm{a_y y_i} + \sum_{\mathrm{i}} \mathrm{\alpha_i}$$

$$\boldsymbol{L(w, b, a) = -\frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i} \qquad (6)$$

L has to be maximized subject to α. The optimization defines as the reduction of the quadratic problem. The solution of this problem represented by the following equation.
Find Max

$$\boldsymbol{\frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i} \qquad (7)$$

According the

$$\boldsymbol{\sum_i \alpha_i y_i = 0, \alpha \geq 0} \qquad (8)$$

V. EXPERIMENT

The dataset that was used in presented work is “20 Newsgroups”[1]. This dataset was collected by Ken Lang and it represents the collection of approximately 20,000 newsgroup documents. The data is mapped into 20 different groups *(table II)* but not preprocessed as vectors of numbered data.

TABLE II. “20NEWSGROUP” DATASET LABELS

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

For this experience the various combination of preprocessing algorithms was used *(table 3)*. In first and second experience the datasets passed through stemming [15] with respect to English language dictionary. For data stemming the Natural Language Toolkit NLTK[2] standard library was used. The stemming increases probability of mapped data to the data labels.

Then TF-IDF algorithm created vector of numbered data. After TF-IDF word-embedding process the dataset was separated on 80% of train and 20% of test parts. For training datasets in first experience the linear SVM with logistic regression was used. The accuracy after testing was about ≈*81,67*%. In second experiment the same preprocess method TF-IDF with stemming was used but with *Stochastic Gradient Descent (SGD)* [16] optimization technique the accuracy was increased ≈*82,38%*.

After the same preprocessing method and model the SGD shows better results therefore the same optimization technique was used in third and fourth experiment. In fourth and fifth experiments the filtered n-gram techniques with combination of TF-IDF was used to preprocess the data for the linear SVM model with SGD. In

[1] http://qwone.com/~jason/20Newsgroups/
[2] http://www.nltk.org/

third experiment the n-grams with sizes 1 and 2 was selected. The results show that classification accuracy increased to ≈89.791%. In fourth experiment additional n-grams filters with sizes 1, 2, 3, 4 was added and the accuracy was increased ≈92.456.

TABLE III. CLASSIFICATION RESULTS

Preprocess	**Classifier**	**Regression**	**Accuracy**
TF-IDF with stemming	SVM Linear	Logistic	81.67%
TF-IDF with stemming	SVM Linear	SGD	82.38%
Filtered n-gram with (1,2) range and TF-IDF	SVM Linear	SGD	89.79%
Filtered n-gram with (1,2,3,4,5) range and TF-IDF	SVM Linear	SGD	92.45%

VI. CONCLUSION

As experiments show the different combinations of text data preprocessing algorithms plays significant role in machine learning classification field. Even when the model to train data was selected the same (in case of this work SVM)

REFERENCES

[1] S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," *Informatica ,* vol. 31, pp. 249-268, 2007.

[2] M. Krendzelak and F. Jakab, "Text categorization with machine learning and hierarchical structures," in *13th ICETA*, Stary Smokovec, Slovakia , 2015.

[3] D. B. Kamran Kowsari, "HDLTex: Hierarchical Deep Learning for Text Classification," University of Virginia, Charlottesville, VA.

[4] S. H. Anil Kumar Sharma, "Application of Deep Learning Techniques for Text Classification on Small Datasets," *IJESC,* vol. 8, no. 4, 2018.

[5] D. H. Tom Young, "Recent Trends in Deep Learning Based Natural Language Processing," Cornell University, 2017.

[6] C. C. Vladimir Vapnik, "Support-Vector Machine," in *Machine Learning*, Boston, 1995.

[7] K. C. Tomas Mikolov, "Efficient Estimation of Word Representations in Vector Space," Cornell University, 2013.

[8] B. A. David Guthrie, "A Closer Look at Skip-gram Modelling David," University of Sheffield , Sheffield.

[9] W. Salton, "A vector space model for automatic indexing," *Communications of the ACM,* vol. 18, no. 11, pp. 613-620, 1975.

[10] V. R. Michael Wong, "Vector Space Model of Information Retrieval - A Reevaluation.," in *Proceedings of the 7th annual international ACM SIGIR*, 1984.

[11] J. Ramos, "Document, Using TF-IDF to Determine Word Relevance," Rutgers University, NJ.

[12] S. Robertson, "Understanding Inverse Document Frequency: On theoretical arguments for IDF," Microsoft Research 7 JJ Thomson Avenue, Cambidge CB3 0FB .

[13] P. Busotti, "Historical Paper On the Genesis of the Lagrange Multipliers," *JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS,* vol. 117, 2003.

[14] H. Li, "Lagrange Multipliers and their Applications," University of Tennessee, Knoxville, 2008.

[15] A. G. Jivani, "A Comparative Study of Stemming Algorithms," Department of Computer Science & Engineering The Maharaja Sayajirao University of Baroda , Vadodara, İndia, 2011.

[16] S. M. Herbert Robbins, "A Stochastic Approximation Method," Univercity of North Carolina, North Carolina, 1951.

[17] B. S. Nello Christianini, "Support Vector Machines and Kernel Methods," *AI Magazine,* vol. 23, no. 3, 2002.