International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 24 (2017) pp. 15986-15993

Enhancing Performance of Deep Learning Based Text Summarizer

Maya John[1] and Jayasudha J.S.[2]

[1]Research Scholar, Department of Computer Science and Engineering, Noorul Islam University, Thuckalay, Kanyakumari, Tamil Nadu, India.

[1]Orcid: 0000-0002-0825-1184

[2]Professor, Department of Computer Science and Engineering, Sree Chitra Thirunal College of Engineering, Pappanamcode, Thiruvananthapuram, Kerala, India.

Abstract

In this world of rapidly increasing digital information, text summarizers are highly essential to provide the most important information to the users. This paper employs deep learning based method to generate summary by classifying sentences. The summary sentences form the minority class which is relatively very small in size compared to the majority class consisting of non- summary sentences. This leads to poor classification performance. Inorder to enhance the performance the data is resampled before applying deep learning method. The resampling techniques include various oversampling techniques, undersampling techniques and combination of both oversampling and undersampling techniques. The performance of the summarizer is enhanced by applying resampling methods.

Keywords: Classification algorithms, Feature extraction, Deep learning.

INTRODUCTION

Text summarization is used to generate a shorter version of the given text in such a way that the main points present in the text are preserved. There are different types of summaries [1]. The type of summary generated depends upon the objective and approach involved in summarization. There are three types of summary namely indicative, informative and critical, based on the purpose of summary.

Based on the presence or absence of input query, text summary may be query oriented or generic in nature [2]. Text summarization is classified into single document summarization and multi-document summarization depending upon the number of documents considered to generate the summary. Text summary can be abstractive or extractive- the former type has sentences different from those in original text, while the latter adopts sentences and phrases present in original textual material. Summary can be neutral summary or biased summary [3]. A neutral summary contains the important information discussed in the input text while biased summary is generated from the view point of a particular person.

Summaries help the readers to evaluate a document without reading the whole text. For example by reading the abstract of a research article one can get an overview of the work presented. The text snippet provided along with Google search engine results help the user in determining whether a web page serves his purpose. News portals can provide information collected from multiple sources.

RELATED WORKS

H.P. Luhn is considered as a pioneer in automatic text summarization. The earliest methods for summarizing text revolved around the concept of frequency and distribution of words in documents [4]. The sentences were assigned a score based on the significance factor and the top ranked sentences were selected as summary. A combination of methods such as cue method, key method, title method and location method can be used to generate summary [5].

With the advent of machine learning techniques and natural language processing in 1990s, several works on text summarization were carried out effectively using machine learning methods. Naives bayes can be used to identify important sentences in a document [6]. Hidden Markov Model and Pivoted QR Matrix Decomposition based method can be used to generate text summary [7]. Osborne proposed a text summarization technique which employed log linear models [8]. Neural networks can be used to generate text summary [9]. Fattah and Fuji used genetic algorithm and mathematical regression technique to summarize text [10]. PadmaPriya and Duraiswamy used Restricted Boltzmann Method (RBM) for shortening a given text [11]. Deep learning based on multilayer perceptron can be used to summarize text [12].

OVERVIEW OF PROPOSED SYSTEM

The proposed system extracts the important sentences from a given text. Deep learning based classifier is used to classify the given text for summarization into summary sentences and non-summary sentences. Since the number of summary sentences are very less compared to non-summary sentences, the classifier tends to be biased towards the majority class which consist of non-summary sentences. Hence in order to enhance the performance of classifier it is essential to do some preprocessing before passing the data to the classifier. In this paper several resampling mechanisms are adopted on the training dataset so as to improve the classifier performance.

The four steps involved in generating summary includes text preprocessing, feature extraction, resampling and classification. Text preprocessing is highly essential to generate good summaries. The text preprocessing methods include sentence extraction from text, tokenization, stop word removal, stemming and lemmatization. Apart from the aforesaid preprocessing reference details, url and email id information are removed from sentences [13].

Feature Extraction

A total of 22 features are extracted from each sentence [13]. Sentence position feature computes a value based on the position of the sentence with respect to the whole text. The sentence length feature indicates the length of sentence with respect to the longest sentence. Presence of numbers in sentence increase the importance of the sentence. Numeric data feature corresponds to the fraction of numeric tokens in a sentence. Cue phrases are phrases which enhance importance of sentences [14]. Examples of cue phrases include in short, to sum up, finally, moreover etc.

The noun count feature is an indication of fraction of noun tokens in a sentence. Similarly the count of proper nouns and adjectives are used to compute feature values. The tokens present in heading tag (<h1> tag), <title> tag and <meta> tag keywords are found out and feature values are computed. The top ten most frequent words in sentences are selected as keywords of the given text and they form the basis for feature based on presence of keywords in sentence. The degree of attachment of each sentences with other sentences are found out. Cosine similarity is used to compare the similarity between two sentences. Sentences which have url or email id are deemed to be important. Hence binary feature is used to indicate the presence or absence of url or email id in sentences.

Contents written within parenthesis in sentences are usually additional information. Feature value is computed based on the effective length of sentence ignoring the content within parenthesis. Information written within quotation marks in sentences are important in nature. Hence a feature is computed based on number of tokens within quotation mark in each sentence. The coreference connectivity in sentences are indicated by pronouns. A binary feature is used to indicate the presence or absence of pronoun in a sentence. The presence of week day or month in a sentence indicates that the sentence is important. Hence two binary features are used separately to refer to presence of week day or month in sentence. When words like because, furthermore etc. are present at the beginning of a sentence, it implies that they specify some additional information which can be ignored. The feature value is set to zero if aforesaid words are present in a sentence and one otherwise. The feature based on modality indicates the certainty of facts present in sentences. There are four types of sentences based on mood namely indicative, conditional, imperative and subjective. Depending on the mood different values are assigned for sentences. The mean term weight of sentences are computed based on Term frequency-inverse sentence frequency (TF-ISF).

RESAMPLING TECHNIQUES

The major challenge faced while employing binary classification based approach for summarizing text is that one has to deal with imbalanced data. A dataset is considered to be imbalanced if the class of interest (positive class) is relatively small (minority class) with respect to other classes (negative class/ majority class). In this context the summary sentences (positive class) is the minority class when compared to non-summary sentences (negative class/majority class). Hence prior to deep learning based classification, resampling methods are used to improve the performance of classification process. The different types of resampling methods used in this work are described in the following sub sections. The resampling methods used here are classified into three types namely under-sampling methods, over-sampling methods and hybrid methods.

Undersampling Methods

Undersampling create a new dataset from the given dataset by randomly eliminating some data (usually from majority class) so as to reduce the class imbalance. The different types of under-sampling methods are described in the following subsections.

Random Under-Sampling

Random undersampling is the simplest under sampling method. It is a nonheuristic method which strives to balance class distribution by randomly eliminating majority class examples. In this approach there are chances of useful data being eliminated. Dataset is very large in size, this method helps in reducing the run time cost.

NearMiss

The NearMiss family of methods [15] considers only a subset of the majority class for resampling. There are three variants of near-miss namely NearMiss1, NearMiss2 and NearMiss3. NearMiss1selects negative examples that are close to a certain number of positive examples. NearMiss2 keeps those data from the majority class whose mean distance to the k most distant point in minority class is lowest. In the case of

NearMiss3, a definite number of majority class data that are closest to a positive class data are retained.

CondensedNearestNeighbour(CNN)

The technique of CondensedNearestNeighbour was proposed by Hart [16]. The goal of CNN undersampling is to choose a subset of the training set such that for every point in the training set its nearest neighbor in subset is selected from the same class. CNN requires many passes over the training dataset and hence it is slow in resampling. CNN performs best when the classes in the dataset are separable. A drawback of CNN is that consistent subset depends on the order in which data is processed.

EditedNearestNeighbours (ENN)

Editing methods process the training data by removing border and noisy instances or by making other necessary cleaning, with the aim of improving classification accuracy of learning algorithm on test data. K-nearest neighbor rule generates less errors when compared to simple nearest neighbor rule as it filters outliers. Based on this concept Wilson [17] proposed the concept of ENN wherein the majority class data is under sampled by eliminating data points which belong to different class when compared to majority of its k nearest neighbours.

RepeatedEditedNearestNeighbours

Tomek [18] proposed RepeatedEditedNearest Neighbour (RENN) based resampling which is variation of ENN. The RENN repeats the ENN algorithm until a stable set is obtained where no more samples are edited out that is repeat the ENN algorithm till there are no more misclassified instances to be eliminated.

TomekLinks

A pair of data points form a Tomek Link [19], if they belong to different classes and are each other's nearest neighbor. In order to perform under sampling the majority class instance of the Tomek links are removed. In this method the retained boundary points are chosen in a better way.

Over Sampling Techniques

Oversampling methods [20] balance the class distribution either by creating new instances of minority class or by replicating the existing data. The two types of over sampling techniques used in this work are Random over sampling and Synthetic Minority Oversampling Technique (SMOTE).

Random Over Sampling

In case of random over sampling the minority class data is randomly replicated in order to improve the number of instances in the minority class. Random over sampling increases the chances of overfitting.

SMOTE

Inorder to overcome the problem of overfitting in random over sampling Chawla et. al. [21] developed SMOTE technique which generates synthetic data to reduce the wide disparity in class distribution. Here oversampling is done by randomly choosing points which fall on the line segments connecting a minority class instance and some/all of its k nearest neighbours. This oversampling process is done with respect to all minority class instances and the number of k nearest neighbours chosen depends upon the specified over sampling ratio.

Hybrid Methods

Since oversampling may result in loss of vital data and undersampling can cause overfitting, hybrid methods based on undersampling and oversampling have been proposed. The hybrid methods include SMOTE+Tomek Link and SMOTE+ENN [22].

SMOTE+Tomek Link

At first data is oversampled using SMOTE during the sampling process and then undersampling is done by eliminating the Tomek links. In this case both majority class instance and minority class object present in the Tomek link are eliminated. This method results in generation of class clusters which are defined in a better way.

SMOTE+ENN

In this method SMOTE oversampling method is followed by ENN undersampling on the training data. This method is less prone to overfitting and has relatively few noisy data.

CLASSIFICATION

Multilayer perceptron (MLP) based deep learning technique is used to distinguish summary sentences from non-summary sentences. The features extracted from each sentence are given as input to the MLP. MLP used here is a fully connected network consisting of three layers namely input layer, hidden layer and output layer as shown in Figure 1. The input layer consist of 22 neurons to capture the feature values of the sentences. The hidden layer consist of 23 units. Since we are performing binary classification with MLP, the output layer consist only of one unit.

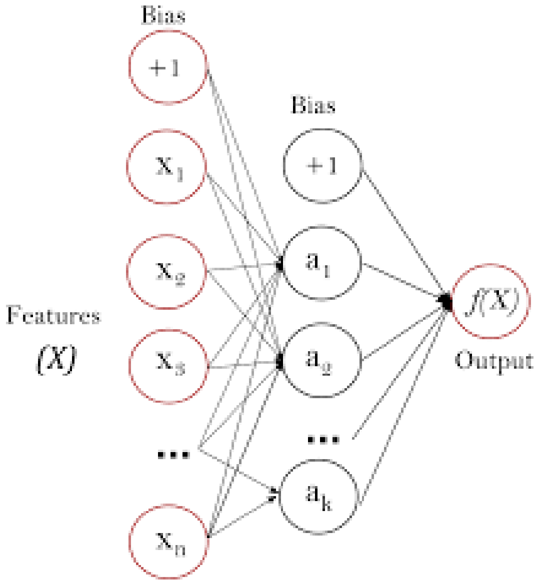

Figure 1: MLP with one hidden layer

The network weights are set to a small random number generated from a uniform distribution, the value of which lies between 0 and 0.05. Sigmoid activation function is used in the output layer as binary classification is done. Either rectifier or tanh activation function is used in the input and hidden layer. The optimizer used works on an efficient gradient descend based algorithm.

PERFORMANCE EVALUATION

CNN dataset is used to carry out the text summarization [23]. The most common sentences in the summaries generated by three evaluators are considered as summary sentences. The sentences are labelled as summary sentences or non-summary sentences. The dataset is divided into two sets, one for training and another for testing. Since there is the problem of class imbalance, before performing classification in training phase the training data is resampled. On the basis of information learned in the training phase the MLP based summarizer classifies the test data. During performance evaluation the efficiency of the classification process on test data is analyzed. The metrics used for evaluating the efficiency of summarizer are F-Measure, G-Mean and AUC. On account of imbalance data problem the traditional performance metric accuracy is not effective in analyzing the performance.

The metric F-Measure is based on the values of precision and recall. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall (Sensitivity) is the ratio of correctly predicted positive observations to the actual total number of positive observations. F-Measure is defined as the harmonic mean of precision and recall. The value of F-Measure is computed using Equation 1.

$$\text{F-Measure} = \frac{2*Precision*Recall}{Precision+Recall} \tag{1}$$

Specificity is the ratio of correctly predicted negative observations to the actual total number of negative observations. G-Mean also known as geometric mean is the square root of product of specificity and sensitivity. Receiver Operative Characteristics (ROC) Curve is used to represent the relationship between true positive rate (TPR) and false positive rate (FPR). It is plotted with FPR on x-axis and TPR on y-axis. Area under the ROC curve is known as AUC. Value of AUC ranges from 0 to 1 and good classifiers have AUC value above 0.5.

The text summarization has been carried out by using MLP classifier with and without prior resampling of training dataset. The different resampling methods used in this work are undersampling methods, oversampling methods and combination of undersampling and oversampling methods. The experiments have been conducted with adam optimizer based MLP. The different epoch values used by MLP are 15, 20, 25, 30 and 35. Apart from varying epoch value of MLP we have also conducted experiments by using rectifier/tanh as the activation function in the input and hidden layer of MLP.

Table 1-10 shows the AUC, F-Measure, G-Mean corresponding to classification using MLP (with/ without prior resampling) with different activation function and epoch values. It is observed from Table 1 that the best classification results are obtained for SMOTE+ENN based resampled data using MLP based on rectifier activation function at epoch value 15.

Table 1: AUC, F-Measure, G-Mean with activation function rectifier and epoch=15

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.67	0.5	0.606
Random Undersampling	0.716	0.55	0.707
NearMiss1	0.696	0.52	0.686
NearMiss2	0.666	0.483	0.638
NearMiss3	0.553	0.196	0.331
CNN	0.701	0.531	0.689
ENN	0.72	0.55	0.715
RENN	0.698	0.533	0.677
TomekLinks	0.667	0.493	0.604
Random Oversampling	0.708	0.537	0.7
SMOTE	0.71	0.539	0.702
SMOTE+Tomek	0.695	0.524	0.679
SMOTE+ENN	0.725	0.552	0.723

Table 2: AUC, F-Measure, G-Mean with activation function rectifier and epoch=20

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.647	0.455	0.564
Random Undersampling	0.731	0.563	0.727
NearMiss1	0.692	0.515	0.683
NearMiss2	0.713	0.549	0.701
NearMiss3	0.629	0.413	0.527
CNN	0.708	0.537	0.7
ENN	0.718	0.547	0.714
RENN	0.735	0.567	0.732
TomekLinks	0.682	0.516	0.637
Random Oversampling	0.707	0.534	0.699
SMOTE	0.698	0.522	0.69
SMOTE+Tomek	0.7	0.524	0.691
SMOTE+ENN	0.72	0.541	0.718

In the case of MLP with activation function rectifier and epoch value 20, classification results are best with RENN resampling of data. Edited Nearest Neighbour resampling based classifier showed best classification performance when the number of epochs are 25 and activation function is rectifier. From table 4, it is observed that RENN resampling led to the best classification with activation function rectifier and epoch value 30. When number of epochs are 35 and activation function is rectifier, random undersampling of training data led to maximum classifier performance.

Table 3: AUC, F-Measure, G-Mean with activation function rectifier and epoch=25

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.672	0.497	0.631
Random Undersampling	0.696	0.527	0.678
NearMiss1	0.689	0.507	0.684
NearMiss2	0.724	0.564	0.714
NearMiss3	0.677	0.506	0.634
CNN	0.716	0.55	0.707
ENN	0.736	0.566	0.733
RENN	0.722	0.551	0.718
TomekLinks	0.671	0.5	0.617
Random Oversampling	0.712	0.537	0.708
SMOTE	0.692	0.518	0.677
SMOTE+Tomek	0.697	0.522	0.687
SMOTE+ENN	0.723	0.542	0.722

Table 4: AUC, F-Measure, G-Mean with activation function rectifier and epoch=30

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.678	0.51	0.631
Random Undersampling	0.702	0.53	0.692
NearMiss1	0.68	0.493	0.675
NearMiss2	0.697	0.53	0.676
NearMiss3	0.673	0.5	0.627
CNN	0.709	0.545	0.695
ENN	0.727	0.555	0.725
RENN	0.73	0.565	0.725
TomekLinks	0.678	0.51	0.631
Random Oversampling	0.7	0.527	0.69
SMOTE	0.715	0.531	0.714
SMOTE+Tomek	0.686	0.502	0.681
SMOTE+ENN	0.726	0.547	0.725

Table 5: AUC, F-Measure, G-Mean with activation function rectifier and epoch=35

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.662	0.483	0.601
Random Undersampling	0.728	0.558	0.724
NearMiss1	0.704	0.519	0.703
NearMiss2	0.698	0.529	0.682
NearMiss3	0.687	0.521	0.653
CNN	0.715	0.552	0.702
ENN	0.715	0.547	0.706
RENN	0.703	0.533	0.69
TomekLinks	0.67	0.494	0.63
Random Oversampling	0.689	0.513	0.674
SMOTE	0.698	0.522	0.69
SMOTE+Tomek	0.689	0.51	0.678
SMOTE+ENN	0.716	0.537	0.715

Figure 2 shows the comparison between the best performance achieved as result of resampling for various epoch values using rectifier as activation function. ENN based resampling led to best classification result for MLP with rectifier as activation function and number of epochs is 25. Degradation in performance is observed on increasing number of epochs beyond 25.

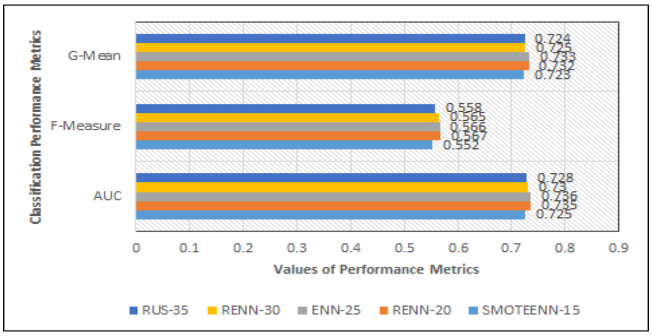

Figure 2: Comparison of best classification results based on activation function rectifier for different epoch values

Table 6: AUC, F-Measure, G-Mean with activation function tanh and epoch=15

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.681	0.517	0.633
Random Undersampling	0.713	0.545	0.704
NearMiss1	0.695	0.517	0.687
NearMiss2	0.678	0.5	0.655
NearMiss3	0.685	0.519	0.648
CNN	0.725	0.558	0.719
ENN	0.737	0.569	0.735
RENN	0.74	0.574	0.738
TomekLinks	0.683	0.519	0.639
Random Oversampling	0.695	0.52	0.684
SMOTE	0.708	0.537	0.698
SMOTE+Tomek	0.687	0.513	0.668
SMOTE+ENN	0.725	0.552	0.723

Table 7: AUC, F-Measure, G-Mean with activation function tanh and epoch=20

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.654	0.467	0.58
Random Undersampling	0.713	0.54	0.707
NearMiss1	0.696	0.519	0.688
NearMiss2	0.714	0.548	0.703
NearMiss3	0.668	0.493	0.615
CNN	0.725	0.563	0.716
ENN	0.724	0.554	0.72
RENN	0.73	0.56	0.727
TomekLinks	0.683	0.519	0.639
Random Oversampling	0.71	0.539	0.702
SMOTE	0.697	0.522	0.687
SMOTE+Tomek	0.689	0.513	0.677
SMOTE+ENN	0.717	0.54	0.716

From table 6 -10 it is clear that while tanh is used as the activation function for all different epoch values (15, 20, 25, 30, 35) RENN based resampling led to the best classification results. Figure 3 shows the comparison of the best result for classification based on MLP using tanh activation at different epoch values. It is clear from figure 3 that MLP employing tanh activation function showed best performance when the number of epochs was 15. The performance of the classifier did not improve on increasing the number of epochs beyond 15.

Table 8: AUC, F-Measure, G-Mean with activation function tanh and epoch=25

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.674	0.5	0.636
Random Undersampling	0.692	0.522	0.672
NearMiss1	0.688	0.505	0.682
NearMiss2	0.719	0.557	0.707
NearMiss3	0.693	0.534	0.658
CNN	0.715	0.551	0.705
ENN	0.723	0.55	0.72
RENN	0.735	0.567	0.732
TomekLinks	0.674	0.5	0.636
Random Oversampling	0.703	0.529	0.696
SMOTE	0.691	0.519	0.674
SMOTE+Tomek	0.691	0.515	0.678
SMOTE+ENN	0.719	0.538	0.718

Table 9: AUC, F-Measure, G-Mean with activation function tanh and epoch=30

Resampling	**AUC**	**F-Measure**	**G-Mean**
No Resampling	0.678	0.51	0.631
Random Undersampling	0.698	0.525	0.686
NearMiss1	0.685	0.502	0.679
NearMiss2	0.69	0.519	0.671
NearMiss3	0.678	0.51	0.635
CNN	0.724	0.564	0.714
ENN	0.72	0.545	0.717
RENN	**0.734**	**0.569**	**0.73**
TomekLinks	0.673	0.5	0.627
Random Oversampling	0.704	0.533	0.693
SMOTE	0.725	0.544	0.725
SMOTE+Tomek	0.692	0.512	0.686
SMOTE+ENN	0.705	0.521	0.704

Table 10: AUC, F-Measure, G-Mean with activation function tanh and epoch=35

Resampling	AUC	F-Measure	G-Mean
No Resampling	0.661	0.482	0.596
Random Undersampling	0.705	0.531	0.698
NearMiss1	0.704	0.519	0.703
NearMiss2	0.705	0.54	0.69
NearMiss3	0.684	0.515	0.651
CNN	0.717	0.554	0.706
ENN	0.711	0.542	0.703
RENN	0.733	0.565	0.73
TomekLinks	0.673	0.5	0.627
Random Oversampling	0.696	0.523	0.682
SMOTE	0.703	0.529	0.694
SMOTE+Tomek	0.684	0.508	0.666
SMOTE+ENN	0.703	0.519	0.702

A comparison between the best classification performance achieved using MLP based on tanh and rectifier activation function is shown in Figure 4. It is clear that RENN resampled data based classification using MLP with activation function tanh and number of epochs 15 showed slightly better performance than ENN resampled data based classification using rectifier activation function at epoch value 25. It is observed that resampling the training data prior to classification led to enhancement of classification performance in terms of AUC, F-Measure and G-Mean. Increased value of G-Mean implies that there is an increase in the number of summary sentences predicted correctly. AUC values show improvement due to changes in true positive rate and false positive rate on account on resampling of data in the training phase.

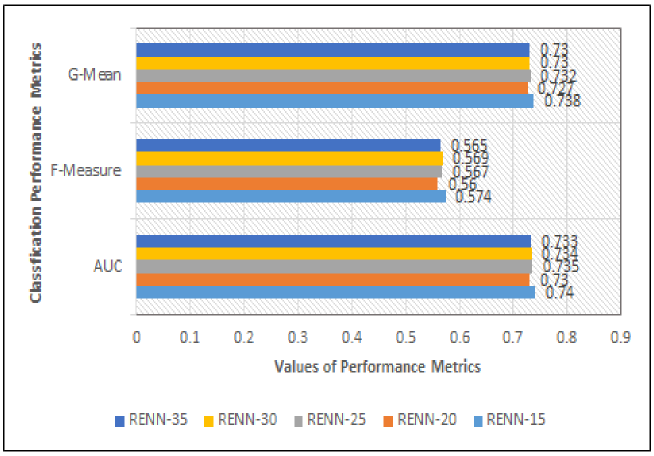

Figure 3: Comparison of best classification results based on activation function tanh for different epoch values

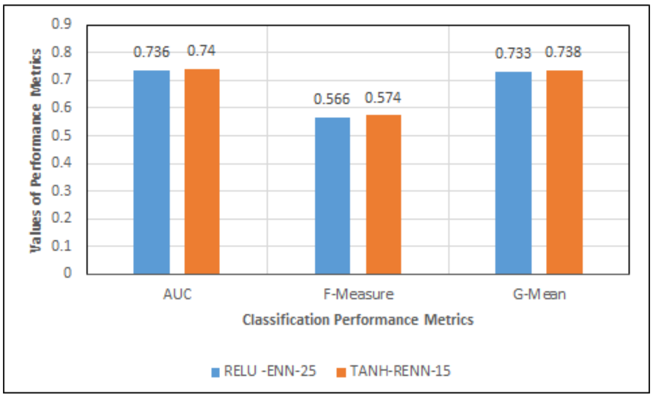

Figure 4: Comparison of best classification result obtained by tanh and rectifier optimizer

CONCLUSION

In the proposed system classification based approach is used for summarizing text. Deep learning based MLP classifier is used for classification. Corresponding to each sentence 22 feature values are computed. On account of imbalance data problem training data is resampled prior to classification. Experiments were conducted by using activation function such as rectifier or tanh in hidden layer and input layer of MLP and the number of epochs were varied from 15 to 35. Resampling data led to improvement in classification performance. The best classification results achieved using rectifier and tanh activation function showed only slight difference. Tanh based MLP outperformed rectifier based MLP. In future other MLP based techniques may be used to create better summarizer. Further studies can be conducted by combining other imbalance data handling methods along with MLP to achieve better results.

REFERENCES

[1] Hahn U, Mani I. Challenges of automatic summarization. Computer. 2000;33(11):29–36

[2] Ozsoy MG, Alpaslan FN, Cicekli I. Text summarization using Latent Semantic Analysis. J Inf Sci. 2011;37(4):405–17.

[3] Hirst G, Hovy E, Johnson M. Natural language processing of Semitic languages. 2014.

[4] Luhn HP. The Automatic Creation of Literature Abstracts. IBM J Res Dev. 1958;2(2):159–165.

[5] Edmundson HP. New methods in automatic extracting. J Assoc Comput Mach. 1969;16(2):264–85.

[6] Kupiec J. A Trainable Document Summarizer. Int ACM SIGIR Conf Res Dev Inf Retr. 1995;68–73.

[7] Conroy J, O'Leary DP. Text Summarization via Hidden Markov Models and Pivoted QR Matrix Decomposition. Proc 24th Annu Int ACM SIGIR Conf Res Dev Inf retrieval. 2001.

[8] Osborne M. Using maximum entropy for sentence extraction. Proc ACL-02 Work Autom. Summ. 2002:1–8.

[9] Yong SP, Abidin AI. Z, Chen YY. A neural-based text summarization system. WIT Trans Inf Commun Technol. 2006;37:185–192.

[10] Fattah MA, Fuji R. Automatic text summarization. World Acad Sci Eng Technol. 2008;37(17):192–195.

[11] PadmaPriya G, Duraiswamy K. An approach for text summarization using deep learning algorithm. J Comput Sci. 2014;10(1):1–9.

[12] Mani KB. Text Summarization using Deep Learning and Ridge Regression. CoRR. 2016.

[13] John M, Jayasudha JS. Pre-processing and Feature Extraction for Text Summarization. Fronteiras. 2017;6:508–515.

[14] Lawler J, and Dry HA, Using Computers in Linguistics: A Practical Guide, Routledge, 2001.

[15] Zhang J, Mani I. kNN Approach to Unbalanced Data Distributions: A Case Study involving Information Extraction. Workshop on Learn from Imbalanced Datasets II ICML Washingt DC 2003.

[16] Hart PE. The condensed nearest neighbor rule. IEEE Trans Inf Theory. 1968 May 14(3):515–516.

[17] Wilson DL. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Trans Syst Man Cybern. 1972 Jul;SMC-2(3):408–421.

[18] Tomek I. An Experiment with the Edited Nearest-Neighbor Rule. IEEE Trans Syst Man Cybern. 1976 Jun SMC-6(6):448–452.

[19] Tomek I. Two Modification of CNN. IEEE Trans Syst Man Cybern. 1976;769–772.

[20] Galar M, Fernandez A, Barrenechea E, Bustince H. A Review of Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. IEEE Trans Syst Man Cybern C Appl Rev. 2011;42(4):463–484.

[21] Chawla N V, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE : Synthetic Minority Over-sampling Technique. J Artif Intell Res. 2002;16:321–357.

[22] Batista GE, Prati RC, Monard MC. A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. SIGKDD Explor Newsl .2004;6(1):20–29.

[23] Hermann, Karl M, Kocisky T, Grefenstette E, Espeholt L, Kay W, Suleyman M, et al. Teaching Machines to Read and Comprehend. In: 28th International Conference on Neural Information Processing Systems. Montreal, Canada; 2015.