Sentiment Analysis of Bangla Language Using Deep Learning Approaches

Conference Paper · February 2021

DOI: 10.1007/978-3-030-76776-1_10

3 authors:

Muntasir Hoq
North Carolina State University
13 PUBLICATIONS **105** CITATIONS
SEE PROFILE

Promila Haque
East Delta University
9 PUBLICATIONS **57** CITATIONS
SEE PROFILE

Mohammed nazim uddin
East Delta University
38 PUBLICATIONS **403** CITATIONS
SEE PROFILE

Sentiment Analysis of Bangla Language Using Deep Learning Approaches

Muntasir Hoq(✉), Promila Haque, and Mohammed Nazim Uddin

East Delta University, Chattaogram, Bangladesh
{muntasir.h,promila,nazim}@eastdelta.edu.bd

Abstract. Emotion is the most important gear for human textual communication with each other via social media. Nowadays, people use text for reviewing or recommending things, sharing opinions, rating their choices or unlikeness, providing feedback for different services, and so on. Bangladeshi people use Bangla to express their emotions. Current research based on sentiment analysis has got low-performance output by using several approaches on detecting sentiment polarity and emotion from Bangla texts. In this study, we have developed four models with the hybrid of Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) with various Word Embeddings including Embedding Layer, Word2Vec, Global Vectors (Glove), and Continuous Bag of Words (CBOW) to detect emotion from Bangla texts (words, sentences). Our models can define the basic three emotions; happiness, anger, and sadness. It will make interaction lively and interesting. Our comparisons are bestowed against CNN, LSTM with different Word Embeddings, and also against some previous researches with the same dataset based on classical Machine Learning techniques such as Support Vector Machine (SVM), Naïve Bayes, and K-Nearest Neighbors (K-NN). In our proposed study, we have used Facebook Bangla comments for a suitable dataset. In our study, we have tried to detect the exact emotion from the text. And in result, the best model integrating Word2Vec embedding layer with a hybrid of CNN-LSTM detected emotions from raw textual data with an accuracy of 90.49% and F1 score of 92.83%.

Keywords: CNN · LSTM · NLP · Sentiment analysis · Bangla language

1 Introduction

Social networking is one of the most popular means of communication. The foremost well-known one is Facebook where individuals of all ages from distinctive communities over the world are associated. People express their emotions by providing posts on social media, such as their likings, disliking, favorite brands, and product reviews. On Facebook, most Bangla users use Bangla in their posts and comments because they can express their exact emotions and feel free to write texts in Bangla. In this paper, feelings in post and comments on Facebook written in Bangla language will be analyzed in such a way as to reflect and explain it in a meaningful manner.

N. Chaubey et al. (Eds.): COMS2 2021, CCIS 1416, pp. 140–151, 2021.
https://doi.org/10.1007/978-3-030-76776-1_10

Multiple techniques can be applied to analyze the data. Sentiment analysis has proved to be profitable in this case as it can detect the emotion, attitude, and polarity that are hidden in words. Using this sentiment, a company can understand their consumer what they want and what more can be added. Buyers can know about their client satisfaction.

Most of the works on sentiment analysis in NLP have been focused on English language. But Sentiment analysis in Bangla language and topic modeling both are new ideas. Sentiment Analysis (SA) in Bangla has an extensive province of posterior possibilities with deep learning. So in this study, we have worked on Bangla textual data obtained from Facebook comments on various topics. The proposed method preprocesses the imbalanced dataset to get a balanced one by removing punctuations, alphanumeric and duplicate words. Our proposed model is implemented with a hybrid of CNN-LSTM framework which can use any Word Embedding as an external layer to identify three basic emotions including: happy, sad, angry. Different Word embedding's used with the hybrid model including Embedding layer, Word2Vec, CBOW, GloVe. Due to an imbalanced dataset, we have been capable of defining only three different types of emotions. No previous work has been done on sentiment analysis using Deep Learning approach with hybridization of CNN-LSTM model in Bangla language as of our knowledge. So our proposed method is the first of its kind.

The remainder of the paper is arranged as follows. In the Sect. 2, we give a summary of the related work conducted in this domain. Section 3 describes how the data was collected, stored, and describes the methodologies that are used in this study, Sect. 4 described the experimental results and comparison with different traditional and state-of-the-art disciplines, and Sect. 5 briefly draws the findings and future directions of research.

2 Related Work

Because of its socio-commercial value, Sentimental analysis has become a hot research topic these days. The number of works on sentiment analysis has been rising by a significant amount in recent years. In [1], the study showed an approach for detecting sentiments from tweet data using deep convolutional neural networks. Several Machine Learning and Deep Learning techniques have been employed in [2] and detected Bangla text which are multi-type abusive. Besides, there have worked with new stemming rules for Bangla language which resulted in better. In [3] the dataset has three categories: positive, negative, and ambiguous and they have worked on LSTM using two types of loss functions. In this work, their dataset is substantial and textual for both Bangla and Romanized Bangla. In [4], they have worked for three categories of opinion: positive, negative, and neutral for Bangla textual movie reviews. In [5] neural network approach is used for emotion detection. They [6] have followed others and adopted distance supervision. Using Bangla WordNet, they have defined six different types of emotions. They have planned to reduce the noisy data from the web. By using LSTM on the noisy nature of Bangla toxic texts, it is possible to get better accuracy [7]. Toxic comment means detecting threats, spam, and vulgar words. In [8], they collect data from Bangla, English, and Romanized Bangla from YouTube comments and they have defined both classifying (positive, negative, neutral) a Bangla sentence and basic six different emotions detection from text. But they have skipped the comments that have more than 30

words. Many sentences in Bangla language can express their final emotion at the end of the sentence. So, if it is wanted to understand the exact emotion, needs to take a long sentence that contains more than 30 words. In this paper [9] they have used Word2vec for detecting top-ranked words or sentences. But for embedding layer it should have good vocabulary. For creating a vocabulary, will need a large dataset that are not available for Bangla language. For creating a dataset, one needs to create data manually and collect automatic data [10]. In [12], a Bangla text corpus was used that contains textual data from Facebook groups on different topics, mainly user comments. They used the same dataset and the same number of emotions as the proposed work, but they have used Naïve Bayes Classifier. In this study [13], attention-based CNN is used for the first time in Bangla sentiment analysis with an accuracy of 72.06%. Regarding LSTM networks with advanced layers, [14] this study achieves an accuracy of 94%. In [15], the authors have worked with several DL techniques and with two different datasets and shown some promising comparisons among all the models. In [8], they collect data from Bangla, English and Romanized Bangla from YouTube comments and they have defined both classifying (positive, negative, neutral) a Bangla sentence and basic six different emotions detection from text. But they have skipped the comments that have more than 30 words. Many sentences in Bangla language can express their final emotion at the end of the sentence. So, if it is wanted to understand the exact emotion, needs to take long sentence that contains more than 30 words.

In sentiment analysis, capturing local features is also important along with capturing the context of words in a sentence with long term dependency. Researches done in this field have not thoroughly explored this region. We have tried to overcome this problem with our proposed hybrid model of CNN-LSTM.

3 Methodology

Many algorithms have been used to solve sentiment analysis and polarity. Each proposed work has used different types of algorithms to solve problems. Most of the approaches have been given high accuracy while classifying English words. We have used different embedding layers, word2vec, CBOW, and Glove for developing Bangla CNN, LSTM, and our proposed CNN-LSTM models to detect emotions from Bangla language texts and have tried to improve accuracy than the existing accuracy.

Sentiment detection can be done in three ways: Sub Sentence Level, Sentence Level, and Document Level Analysis. In our proposed method we have followed the document level analysis because we have seen many people express their exact emotion at the end of the text. And few people show their emotion by only using a single sentence and very few people express their emotion by using only one sentence. By using document-level analysis, we have detected exact emotion from our Facebook comment dataset. We have continued annotation based on this document-level analysis. In the current section, an overview of our study is described. Figure 1 illustrates the step-wise architecture of the proposed model.

Fig. 1. Proposed architecture.

3.1 Database Formation

This is the initial phase of the analysis process. The dataset we used for this work consists of a large number of textual comments of different users collected from three Facebook groups by using Facebook graph API. We have collected the dataset developed in this study paper [11]. From six emotions and 5640 annotated data from 32000 comments, due to imbalanced data we chose to detect three basic emotions excluding other emotions. The selected emotions are: happy, sad, and angry emotions.

Table 1 provides the class labels, and the amount of data we have used.

In the preprocessing stage, the dataset was processed and cleaned as follows:

- Class labels were removed from the dataset.
- Texts have been split into tokens by white space
- Punctuations have been removed from each token.
- Bangla stop words were removed that are available in our file. The difference in token length can be seen as before removing stop words token length was 48892

Table 1. Emotion Class level and amount of data

Class level	Amount of data
Happy	1000
Sad	1000
Angry	1000

- After removing Bangla stop words token length is 48889. So, stop words are those words that are repeated mostly. Stop words do not play any role. These are like a preposition, conjunction, etc. The list of Bangla stop words were obtained from an open-source project[1].
- Vocab file was created using our clean textual data.

Our processed data contains comments where one comment contains a minimum of 1 word and a maximum of 148 words. In this study [8], the comments were skipped that have 30 words but many comments express emotions at the end of the text. For this reason, we have taken both small and large comments.

3.2 Word Embedding

Word embedding is a feature learning and language modeling technique in NLP. It is required for representing words and documents using dense vector representation. We have used several popular words embedding for sentiment analysis from the text. These are Embedding layer, CBOW, Glove, and Word2vec. In this study, using these word embedding layers, features are extracted and fed into the proposed model to detect emotions from textual data.

Embedded Layer. Firstly, we have used an embedding layer with defined library functions. For developing the model, we have loaded the vocab file created at the database formation phase and the training comments (happy, angry, and sad). Then the training document for integers sequence has been loaded by using Keras API. Then we have defined pad sequence adding empty words to the text for the same length and the X and Y training-testing, that is the transformation of actual string to a number. We have categorized levels 0, 1 & 2 for happy, sad, angry emotions respectively. 80% of data has been used for training and the remaining 20% is for testing. In our embedding layer, we have used 100-dimensional vector space and vocab_size for the number of words in our vocabulary and max_length for input length.

Word2vec. In Word2Vec embedding, words are represented with features which are vector based to capture the similarity. Hence similar words are likely to have similar vectors. The word similarities are calculated with the help of Cosine Similarity Function from the word vector values. The Cosine Similarity function is given below:

$$similarity = cos\theta = \frac{\bar{a} \cdot \bar{b}}{|\bar{a}| \cdot |\bar{b}|} \tag{1}$$

Here, $\bar{a}$, and $\bar{b}$ are two word vectors.

These similarity values range from a lowest of −1 and a highest of 1. In our word2vec model, we have passed the clean sentence and defined embedding vector space 100 and the number of neighbor word 5 as the window size. We have used workers 4 for the number of threads when fitting the model. After that, we have defined min_count 5 that means the number of counted words occurrence. Finally, a word2vec file has been created where we have found the data as ASCII code.

CBOW. In this proposed method, CBOW model has been created by using gnsim library that is special for word2vec. The working procedure has been followed the same as word2vec except min_count is considered 1 in this case. As this method works based on the probability of words so the length of the vocabulary is different than word2vec which works based on finding similar word to word. The result has been saved in ASCII code format to be used for the CNN, LSTM and CNN-LSTM model.

Pre-trained GloVe (Global Vectors). We have used a pre-trained Bangla glove file that has 100-dimensional space. It generates an ASCII code text file. We have used this file with our model. Glove works based on word co-occurrence probability. Let, co-occurrence count is X whose entries is X_{ij}. We have tabulated the number of times where word j occurs in the context of the word i. Suppose, $X_i = \sum kX_{ik}$ be the number of times and any word can occur in context word i. Now, in the end, we have found Pij = P(jli) and we can write it XijlXi.

3.3 Convolutional Neural Network (CNN)

At first, we have added a method for sequence. Every Keras model using a build or customized sequential class is for a linear stack of layers. Then we have added our embedding layer where we define vocab size, dimensional vector space, and input length. Then, we have defined Convolution layer1D where we have defined filter size and kernel size and activated RELU (rectified linear activation function). Here, filter size has been used for parallel field word processing and relu use for transferring and summing the weight from the input node to output for the input. After that, we have added pool size, flatten method where the pool size variable has been used for a hyperparameter for our CNN model and have flattened the input that does not affect by batch-size. Then we have added a dense function where we have activated relu and sigmoid with a different value. We have seen the output in a 2D vector where features extracted by CNN in back-end multilayer perceptron interpret the CNN feature. We have used the sigmoid function for output values where 0 for happy emotion, 1 for sad emotion, and 2 for angry emotion. Then, we have added the compile method in our model where we define loss, optimizer, and metrics. We have used the binary cross-entropy loss function to classify binary classification problems. We had to keep track of the loss that is produced in training time. For this reason, we have implemented the "adam" optimizer function. After this procession, our training model has got fitness. So, we have defined testing wherein training period testing data were not seen. We have used 200-dimensional vector space with 32 filters (parallel fields for processing words) and a kernel size of 8 with a rectified linear (relu) activation function. We have used a 12582-word size vocab file. Dimensional space can be used 50, 100, 150 it depends on data length.

3.4 Long Short-Term Memory (LSTM)

LSTM is suitable for analyzing time-series data and can remember several states that we needed. It has got a record for natural language text compression and can be trained as supervised training. As LSTM is a feedback network, it can reduce error by backtracking and changing weight. In LSTM, memory units handle a period dependency. It has four units: Cell, Input gate, Output gate & Forget gate.

According to Fig. 2, we can see there is three input where two inputs have come from the previous state. One is h_{t-1}, that has come from previous LSTM, and another is C_{t-1}, which is a memory and it has come from the previous time step. One of the important parts is the gate, where one is the forget gate and another is the memory gate.

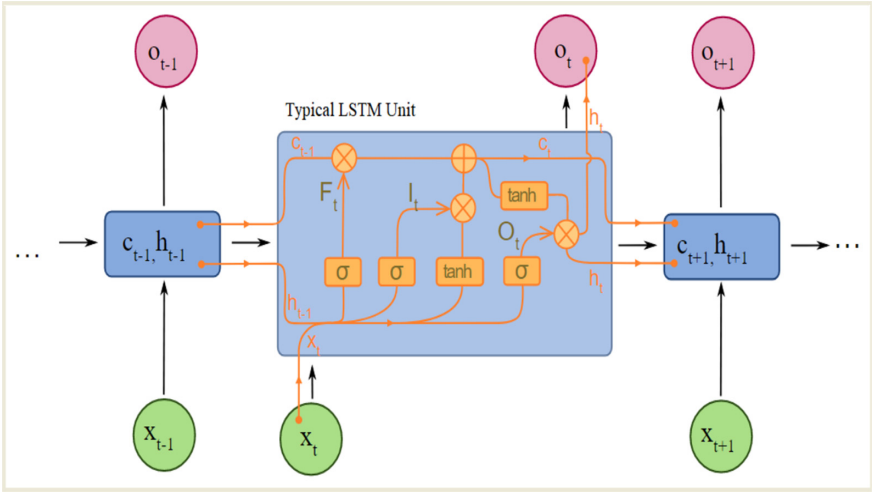

Fig. 2. A typical LSTM

The generic equation for LSTM can be described as follows:

$$i_t = \delta\left(W^i x_t + U^i h_{t-1}\right) \tag{2}$$

$$f_t = \delta\left(W^f x_t + U^f h_{t-1}\right) \tag{3}$$

$$o_t = \delta\left(W^o x_t + U^o h_{t-1}\right) \tag{4}$$

$$c_t = \tan h\left(W^c x_t + U^c h_{t-1}\right) \tag{5}$$

$$c_t = i_t \odot \tilde{c}_t + f_t \odot \tilde{c}_{t-1} \tag{6}$$

$$h_t = o_t \odot \tan h(c_t) \tag{7}$$

Here, i_t represents the input gate, f_t represents forget gate, o_t represents the output gate, δ represents the sigmoid function.W^i, W^f, W^o, W^c, U^i, U^f, U^o, U^c represent the weight matrices and x_t is the vector input to timestep t, h_t current is exposed hidden state and memory cell state is c_t, $\odot$ this is the element-wise multiplication.

In our study, we have built an LSTM model where we have defined embedding size 100 and the embedding method which contains vocab size, embedding size, and input length. We have added a dropout of 0.4 where dropout has reduced the over fitting. We have added a compilation method that we also have used on CNN. After that, we have fitted the training data and used batch size 32 and epoch 15 with validation data.

3.5 CNN-LSTM

In general, LSTM is better for text classification. But in text classification probably a small dataset can vanish the power of LSTM. CNN has its efficacy at learning the local features and spatial structure of textual data and adding a CNN after embedding layers which fed into an LSTM network helps to acquire the global and sequenced characteristics of the input textual data. For this reason, we have created a hybrid model by using CNN and LSTM that has been shown in our proposed methodology. Basically, in our hybrid model (CNN-LSTM) we have pooled a CNN and feed that into LSTM. We have used an embedding layer and used a single Conv1D pooling layer that has helped LSTM to run faster.

In this study, an extensive experiment is done to show that our proposed model outperforms other models in terms of detecting emotions. In this work, for mathematical analysis, for programming Python is used. We have used Google Colab for the simulation.

4 Result and Analysis

In this section, we have discussed the achievement of our proposed method. Some tables and figures are given here, where different kinds of models and their accuracy and F1 scores are shown. We have done our work by defining 80% training and 20% testing from our dataset that have real-time data. Our dataset contains 3000 Facebook comments that have been collected from three different Facebook groups and we have tried to predict three basic emotions: happiness, anger, and sadness from the data. We have implemented CNN for Bangla language and our proposed algorithm is LSTM and we also have made a hybrid model that contains CNN-LSTM.

In Table 2, we can see the result of CNN model where we have used different embedding layers for developing a good Bangla CNN. For our 80% training and 20% testing splitting data, we have got the highest accuracy by using pre-trained Bangla glove embedding. On the other hand, we have got a higher F1-score for CBOW embedding.

In Table 3, we have provided the result of LSTM model where we have got the same accuracy as well as F1-score for Word2vec, CBOW, and Glove embedding.

The result of our CNN-LSTM hybrid model by using different embedding layer has been shown in Table 4.

In CNN-LSTM hybrid model, we have got the best accuracy and F1-score by using Word2vec embedding. We did not get a good result by using only the embedding layer in our three models.

Table 2. Accuracy and F1-score for different embedding layer in CNN model

Model	Accuracy	F1 score
Embedding layer	56.67%	46.74%
Word2vec	82.50%	85.49%
CBOW	83.33%	87.14%
Glove	84.00%	84.46%

Table 3. Accuracy and F1-score for different embedding layer in LSTM model

Model	Accuracy	F1 score
Embedding layer	33.33%	16.67%
Word2vec	66.67%	80.00%
CBOW	66.67%	80.00%
Glove	66.67%	80.00%

Table 4. Accuracy and F1-score for different embedding layer in CNN-LSTM model

Model	Accuracy	F1 score
Embedding layer	33.33%	16.67%
Word2vec	90.49%	92.83%
CBOW	86.00%	88.88%
Glove	80.16%	83.16%

The output of our classifier can be evaluated in terms of accuracy. Therefore, the results of the classifier are measured in terms of accuracy, F1-scores, and AUC curve. Accuracy can be defined as:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions\ made} \tag{8}$$

To calculate F1-score, a confusion matrix is expressed as in Table 5 to understand the binary classifier predictions with tp, fp, tn, fn as true positive, false positive, true negative, and false negative respectively.

From the confusion matrix, F1-score can be defined as:

$$Precision = \frac{tp}{tp + fp} \tag{9}$$

$$Recall = \frac{tp}{tp + fn} \tag{10}$$

Table 5. Confusion Matrix for different embedding layer in CNN-LSTM model

	Emotion 1 predicted	Emotion 2 predicted
Emotion 1 actual	*tp*	*fn*
Emotion 2 actual	*fp*	*tn*

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{11}$$

In Fig. 3 the differences in AUC score among our models have been shown. As the result of the embedding layer is less than the baseline, it has not been shown in the Figure.

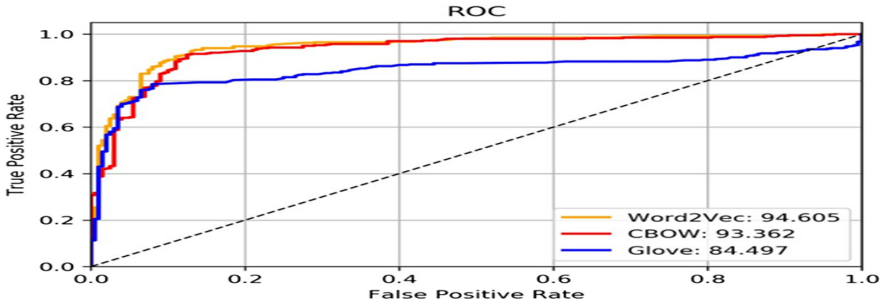

Fig. 3. AUC score CNN-LSTM model

From Table 2, 3, and 4 the best result has been found from the word2vec CNN-LSTM model. CBOW CNN, LSTM, and CNN- LSTM model also have almost equal accuracy and F1 score in the case of 80% training and 20% testing data split. We have provided both accuracy and F1 score. Because accuracy has been showing the ratio of correctly predictive all observations. We have used it because our all classes are equally important. We also have shown F1 score because it is a mean of Precision and Recall as well as it has given an excellent performance for incorrectly classified cases rather than accuracy.

We have developed four models and got the best result by using word2vec embedding. That model has reduced our time and space complexity and provided us the highest accuracy. Alongside 80-20 splitting for the training and testing data depicted in Table 4, we have split 70% training and 30% testing data in Table 6 then, 75% training and 25% testing data in Table 7. Basically, we have got the best result in Word2vec CNN-LSTM model for splitting 80% training and 20% testing data with an accuracy of 90.49% and an F1 score of 92.83% for three basic emotions from 3000 Bangla comments by using Word2vec hybrid model (CNN-LSTM). They have [12] used the same dataset for the same three emotions, but their overall accuracy was 78.6% which is lower than our proposed method. In this paper [11], they have got 77.16% by using only 301 sentences and defined only two basic emotions. We have tried to implement the best model rather

than the existing model. In [12], they also have used the same dataset but we have got a better result using our hybrid model for three emotions.

Table 6. Word2vec CNN-LSTM model (70% training & 30% testing data)

Model	Accuracy	F1 score
Embedding layer	44.53%	36.51%
Word2vec	77.67%	84.94%
CBOW	77.44%	82.54%
Glove	82.40%	87.33%

Table 7. Word2vec CNN-LSTM model (75% training & 25% testing data)

Model	Accuracy	F1 score
Embedding layer	44.53%	36.51%
Word2vec	86.13%	89.12%
CBOW	83.20%	88.26%
Glove	82.40%	87.33%

5 Conclusions

Our effort was to give priority to identify the right way to define the emotion from text accurately and to create something that has not been done yet. In Bangladesh, people use social media for sharing their emotion, branding their products, sharing information. It is very tough to get exact emotion in Bangla language because people express the same emotion writing it by using different structures. It is difficult to retrieve emotion from texts; the area of English language consists most of the works. Our first contribution is to create a hybrid Bangla language CNN-LSTM model and the second contribution is to get a better result than the existing models. We have also created Bangla language CNN and LSTM. Now when people will work with the Bangla language, they can use it. It will reduce their time. We have created the hybrid model with different Word Embedding's to find the best model that will provide us higher accuracy and reduce loss. We have built a model can detect emotion with a good accuracy that has been shown in the result part. Due to imbalanced dataset, there aroused some problems. And hence we had to work with only three emotions, but in our future study, we would like to create our balanced dataset and would like to work with more emotions. Negations can be handled too for better performance.

References

1. Sarkar, K.: Sentiment polarity detection in bengali tweets using deep convolutional neural networks. J. Intell. Syst. **28**(3), 377–386 (2019). https://doi.org/10.1515/jisys-2017-0418
2. Emon, E.A., Rahman, S., Banarjee, J., Das, A.K., Mittra, T.: A deep learning approach to detect abusive Bengali text. In: 2019 7th International Conference on Smart Computing & Communications (ICSCC) (2019). https://doi.org/10.1109/icscc.2019.8843606
3. Hassan, A., Mohammed, N., Azad, A.: Sentiment Analysis on Bangla and Romanized Bangla Text (BRBT) using Deep Recurrent models, October 2016. https://arxiv.org/abs/1610.00369. https://www.researchgate.net/publication/308831680_Sentiment_Analysis_on_Bangla_and_Romanized_Bangla_Text_BRBT_using_Deep_Recurrent_models/citation/download
4. Banik, N., Rahman, M.H.H.: Evaluation of naïve bayes and support vector machines on bangla textual movie reviews. In: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1–6 (2018). https://doi.org/10.1109/ICBSLP.2018.8554497. https://ieeexplore.ieee.org/document/8554497
5. Abdul-Mageed, M., Ungar, L.: EmoNet: Fine-grained emotion detection with gated recurrent neural networks. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, vol. 1, Long Papers, Vancouver, Canada, pp. 718–728 (2017). https://doi.org/10.18653/v1/p17-1067
6. Das, D., Bandyopadhyay, S.: Developing Bengali WordNet Affect for Analyzing Emotion (2010). https://www.semanticscholar.org/paper/Developing-Bengali-WordNet-Affect-for-Analyzing-Das-Bandyopadhyay/cb8e29ffe894321a34aa50d302d022f207e5a73a
7. Banik, N., Rahman, Md.: Toxicity Detection on Bengali Social Media Comments using Supervised Models, November 2019. https://doi.org/10.13140/RG.2.2.22214.01608
8. Irtiza Tripto, N., Eunus Ali, M.: Detecting multilabel sentiment and emotions from Bangla youtube comments. In: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, pp. 1–6 (2018). https://doi.org/10.1109/icbslp.2018.8554875
9. Abujar, S., Masum, A.K.M., Mohibullah, M., Ohidujjaman, Hossain, S.A.: An approach for Bengali text summarization using Word2Vector. In: 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kanpur, India, pp. 1–5 (2019). https://doi.org/10.1109/icccnt45670.2019.8944536
10. Canales, L., Strapparava, C., Boldrini, E., Martinez-Barco, P.: Intensional learning to efficiently build up automatically annotated emotion corpora. IEEE Trans. Affect. Comput. 1–1 (2017). https://doi.org/10.1109/taffc.2017.2764470
11. Rahman, M.A., Seddiqui, M.H.: Comparison of classical machine learning approaches on Bangla textual emotion analysis. ArXiv:190707826 *Cs*. http://arxiv.org/abs/1907.07826 (2019). Accessed 19 Mar 2020
12. Azmin, S., Dhar, K.: Emotion detection from Bangla text corpus using Naïve Bayes classifier. In: 2019 International Conference on Electrical Information and Communication Technology (ECIT), Khulna, 1–5 Dec 2019. https://doi.org/10.1109/eict48899.2019.9068797
13. Sharmin, S., Chakma, D.: Attention-based convolutional neural network for Bangla sentiment analysis. AI Soc. (2020). https://doi.org/10.1007/s00146-020-01011-0
14. Ahmed, A., Yousuf, M.A.: Sentiment analysis on Bangla text using Long Short-Term Memory (LSTM) recurrent neural network. In: Kaiser, M.S., Bandyopadhyay, A., Mahmud, M., Ray, K. (eds.) Proceedings of International Conference on Trends in Computational and Cognitive Engineering. Advances in Intelligent Systems and Computing, vol. 1309. Springer, Singapore (2021). https://doi.org/10.1007/978-981-33-4673-4_16
15. Rahman, M., Haque, S., Saurav, Z.R.: Identifying and categorizing opinions expressed in Bangla sentences using deep learning technique. Int. J. Comput. Appl. **176**(17), 0975–8887 (2020)