Group 10: Early Depression Detection using NLP

Dipan Shah
1171547

Krishna Gandhi
1170559

Sachin Singh
1141124

Dhruva Shah
1165998

Palak Patel
1166610

Abstract

Social networks have been developed as a great platform for its users to communicate with their friends and share their opinions, photos, and videos reflecting their feelings and sentiments. This creates an opportunity to analyze social network data for users' sentiments to investigate their moods and attitudes when communicating via these online tools. Hence, the proposed study aims to exploit deep learning techniques for detecting a probable depressed Twitter user based on his/her tweets. For this purpose, we have trained and tested our dataset on two deep learning models: LSTM and CNN to distinguish whether a user is depressed or not. The results show that both the models perform well with our dataset and the best performing configuration gave us the accuracy of 99.28%.

1 Introduction

Depression and anxiety have become a serious issue for the current generation affecting a number of people everyday. Due to the stigma around mental health, people avoid seeking professional consultation and thus different channels of social media become their diary to share their state-of-mind. Recent studies indicate a correlation between high use of social media and increased depression. Since more number of people rely on social media usage, analysing these posts will benefit in depression detection.

The content or points created by the user is the data that is valuable for the researchers to analyze the state of mind M. R. Islam and Ulhaq (2018). Twitter is a social media platform that has a limit of 140 characters. Twitter's API enables complex query pulling and extraction of values on large scale. This permits collection of huge amount of text data essentially required for a text classification task.

Romero (2019) have curated a dataset comprising of many such Tweets that have been labelled as integers 0 or 1 representing "Non-Depressed" or "Depressed" respectively. The deep learning models are trained to classify the input texts assigning them a label as an output in the same format: 0 for Non-Depressed and 1 for Depressed. A simple representation of the proposed working is shown in Fig. 1

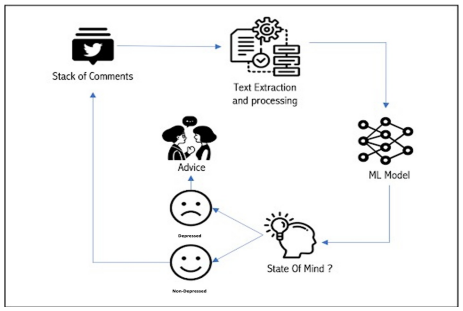


Figure 1: Figure that explicitly shows the input and the output of the task

In this paper, two Deep Learning Models: Convolutional Neural Network and Long-Short Term Memory have been implemented to detect depression in user tweets using text classification, whose performance will be evaluated based on the accuracy of the DL models and a front-end application that will enable qualitative human-evaluated assessment of the models, for completely new data.

An effective real-time implementation of this idea, social media platforms may suggest online professional help to the users on detecting depressive posts. This may prove to be a valuable contribution to not only the technical community but also to the social one.

2 Related Work

In recent years, the research is inclined towards development of behaviour analysis - ML models for detecting depressed users that have an extensive level of self-disclosure on social media.

Govindasamy and Palanichamy (2021) implemented two different types of classifiers: Naive Bayes and a hybrid model called NBTREE on the twitter dataset of size 3000, to compare the results in terms of accuracy. Both the classifier performed equally well yielding an accuracy of 97.31. However, the size of the dataset is very small which helped to achieve the good accuracy. Further, in another article, a linear SVM model outperformed Naive Bayes and Decision Tree along with tfidf vectorization technique on a twitter dataset, that achieved 82.5% accuracy(Alsagri and Ykhlef, 2020).

Ensembling machine learning technique was used along with tfidf and bigram model's vectors, the results of which were fitted into 6 different classifiers (Jagtap et al., 2021). Results from 3 best performing classifiers are ensembled by blending technique using KNN strategy that achieved 96.4% accuracy.

In the work of Owen et al. (2020), the transformer based language models such as BERT and ALBERT are compared with traditional baseline model SVM on manually labelled dataset with size 4500. These models perform better on the trained data but do not produce accurate results when tested against new data. Imbalanced and insufficient data might lead to such poor performances of the language models.

The previous work done suggests that the traditional Deep Learning models when trained with sufficient and balanced data, and efficient preprocessing technique, perform better. Therefore, techniques like TfidfVectorizer and Keras Tokenizer are selected as they align with the requirements of CNN and LSTM.

3 Approach/Methodology

This study aims to analyze the performance results of aforementioned DL models on the basis of two approaches as shown in Fig.2. Topic modeling is implemented to label the dataset and test unsupervised learning approach. This technique may be applied to real-time tweets for generating a well-labelled dataset with minimum error and time-consuming manual labelling can be avoided. Deep Learning models can be trained on this self-labeled dataset only if the labelling is very near to the ground truth. Results of this approach are discussed further in Section 4.

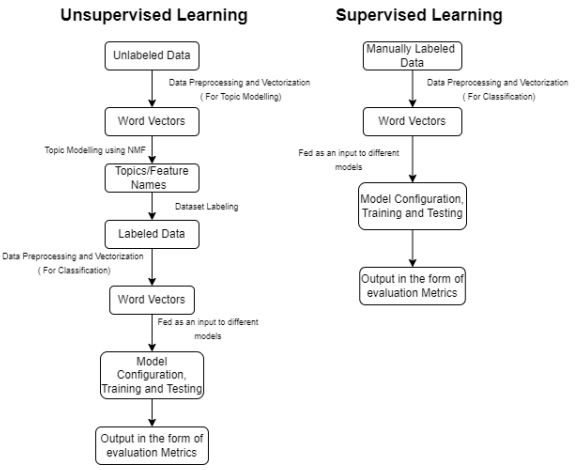


Figure 2: Methodology diagram

3.1 Data Preprocessing

The text data loaded from social media was not accurate as it consisted of spelling and grammatical errors, abbreviations and irrelevant use of punctuation marks. Therefore, basic text preprocessing tasks like URL, email, punctuation and stopword removal, lowercase conversion and lemmatization were performed.

Preprocessed data is in text format, however DL models require numerical data for complete understanding. This is done using the Word-embedding models. Here, we propose the two different word-embedding models.

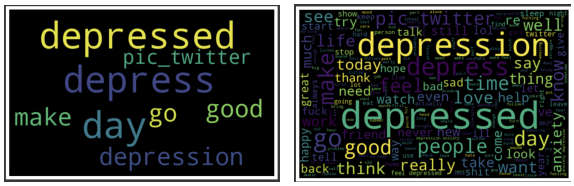
1. Tokenizer
2. TF-IDF vectorizer

Keras Tokenizer's text_to_sequences() function transforms each string to a sequence of integers which will output different numbers of features for each text, based on the availability of number of tokens present in the text. This problem can be solved by using padding where zeros will be added to each text vector such that all the texts will have same number of features, fullfilling the requirement for both deep learning models. Another technique used is TfidfVectorizer that dominates in tasks of visualizing important words in document and topic modeling by using the importance score of words(Yun-tao et al., 2005).

3.2 Topic Modeling

For the Unsupervised Learning approach shown in Fig.2, topic modelling is used to obtain similar words, and then train the models that will create labels for the dataset. It is statistical technique that is used to obtain insights of the corpus by finding

a set of similar words. Similar approach is needed to find a set of words from the dataset relevant to the topic of "depression". Latent Dirichlet Association (LDA) and Non-negative Matrix Factorization (NMF) are extensively used in for topic extraction. After testing both the above mentioned algorithms, NMF produced better results with these short social media posts showed in Fig 3.



(a) LDA Features Name (b) NMF Features Name

Figure 3: Word cloud for topic modeling

3.3 Training Models

CNN architecture when used with word embeddings are effective for classification problems as they are able to pick salient features irrespective of their position in the sentence(Goldberg, 2015). The size of the fixed windows used to feed input to the network restricts the model's ability to understand context of the text after the window size.

Unlike CNN, RNNs can process any length of text data, preserving information about the previous words to build and understand the context. Although, this slows down the computation and introduces the vanishing gradient problem.

LSTM models overcome the issue of vanishing gradient and information loss faced by RNN, making it a suitable choice for text classification(Zhang et al., 2018).

4 Experiments

4.1 Data Description

Dataset used for this experiment has been compiled from multiple sources.[1][2]. It consists of total 19713 tweets with each entry is labeled as 0 or 1, where 0 represents non-depressed tweets and 1 represents the depressed tweets (Dataset Shape: (19713, 2), size: 2.09 MB csv file). In this dataset, 11,286 (57.25%) tweets are depressed tweets and 8247 (42.74%) are non-depressed tweets. Train-Test Split of 70:30 is used, where 70% of total data is used for training and 30% is used for testing.

4.2 Experimental Details

Kaggle Environment is used to run the experiments mentioned further, whose details have been discussed in detail in Appendix A. All the datasets used here are open source and publicly available on Kaggle.

Initial phase of the experiments consisted of training the NMF and LDA models on the clean and preprocessed data, to generate labels. NMF() and LatentDirichletAllocation() functions from the scikit python library were used to generate two topics for each model: 0 for "Non-depressed" and 1 for "Depressed". 20000 features were used for each tokenizer, which is an optimal value with regard to computational resources available. The output for this part of the experiment was qualitatively evaluated by the team, analysing which model gave a better word cloud for "depression".

Model Architecture used for the final experiments are displayed in Table 1.

Table 1: Model Implementation Details

WITHOUT TOPIC MODELLING			
	Hyperparameters	Model Configuration	Size
LSTM	Learning Rate: 0.001 Optimizer: Adam Loss function: binary_crossentropy Training Time: 13 min 24s	InputLayer Embedding LSTM MaxPooling Dense OutputLayer: Dense	(None,74) (74,20) (None,74,15) (None,15) (None,32) (None,1)
CNN	Learning Rate: 0.001 Optimizer: Adam Loss function: binary_crossentropy Training Time: 1 min 32 s	InputLayer Conv1D GlobalMaxPooling Dense OutputLayer: Dense	(None,74,200) (None,74,128) (None,128) (None,10) (None, 1)

The results of the experiments were quantitatively evaluated in terms of training and testing accuracy and Binary Cross Entropy Loss function. It is a maximum Likelihood estimator that has asymptotical properties, are consistent and statistically efficient. The logarithm in the cross-entropy will undo any exponential behavior given through output activation units like the sigmoid function. Moreover, a GUI was developed using Streamlit library that could be used to qualitatively test new random data for both the models. As the user will be unaware of the actual data that was used in training, this technique is like a black-box testing for the models. User has the option to select between both the models. A text output will be generated stating whether the test tweet implies depression or not as shown in the Fig. 5. It has been designed

[1]https://bitbucket.org/nlpcardiff/preemptive-depression-anxiety-twitter/src/master/DATD/DATD$_training.csv$

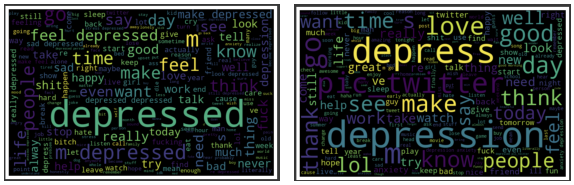
[2]https://github.com/viritaromero/Detecting-Depression-in-Tweets/blob/master/sentiment$_tweets3.csv$

keeping in mind the real-time application of the project and has a scope of further development.

5 Results

5.1 Topic Modeling

As mentioned earlier the results of the topic modeling were analysed using the word clouds formed out of the set of words. Fig 4 represents the word cloud for NMF model.



(a) NMF depressed tweets (b) NMF non-depressed tweets

Figure 4: Word cloud for NMF

It can be noticed that both the topics contain similar words that directly imply depression, like "depression" and "anxiety" as opposed to the word clouds of the original dataset that consisted distinctive words in both the topics. This means that the topic modeling technique performed poorly resulting in incorrect labelling for the dataset. The reason for such a poor performance might be insufficient data. Any topic modeling technique requires quite huge data to actually learn all the context that can be inferred, and then form a set of words for a given topic. In absence of such an important resource, the model failed to learn context and gave the same word sets for both the topics.

5.2 Without Topic Modeling

The results obtained from both the models in terms of accuracy have been tabulated in Table 2.

Table 2: Results

Model	Training Accuracy	Testing Accuracy
CNN	99.28%	84.61%
LSTM	96.79%	87.10%

Both the models have performed equally well, however CNN gave slightly better results than LSTM. CNN was able to capture the context of the data better. To test the models further, actual tweets from Twitter were tested using the GUI application as shown in Fig. 5. It was observed that CNN was able to predict the class of the tweet irrespective of the position of the contextual words like "depressed" and "sad", whereas the output of LSTM was dependent on such positions. This might be due to CNN's capability of extracting local and position-invariant features from textual data.

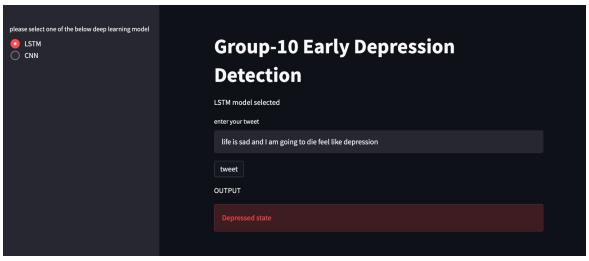


Figure 5: GUI Application with a test tweet and its result using CNN Model

Additionally, when a large input text was tested the classification by both the models was not efficient. The models were still able to recognize a few different forms of the same stem word, which is remarkable when such a limited data is used. For an instance, when the tweet is large in size and contains few unseen words such as synonyms of depression "depressed", "sad", etc. models could not predict it as depressed state. To overcome this problem, different model configurations were tested and the best configurations were selected. This improved overall performance of the model. However, we did not achieve the expected results.

6 Conclusion

This study represents an experimental evaluation for detecting early depression in twitter posts. Using the twitter dataset along with word embedding techniques for data preprocessing, we ran a comparative analysis of CNN and LSTM as a result of which, CNN outperformed LSTM by achieving the highest training accuracy of 99.28%. Front-end developed enables qualitative real-time, black-box testing of the models, that has ample scope of further development and real-time application. The models have certain limitation with respect to long sequence of input data and more accurate prediction on real-time data, owing to the small dataset and limited computational resources. The performance of the models could be improved when trained on a much larger dataset, as an extended study of this article.

References

Hatoon Alsagri and Mourad Ykhlef. 2020. Machine learning-based approach for depression detection in twitter using content and activity features.

Yoav Goldberg. 2015. A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726.

Kuhaneswaran AL Govindasamy and Naveen Palanichamy. 2021. Depression detection using machine learning techniques on twitter data. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 960–966.

Nakshatra Jagtap, Hrushikesh Shukla, Vaibhavi Shinde, Sharmishta Desai, and Vrushali Kulkarni. 2021. Use of ensemble machine learning to detect depression in social media posts. In *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, pages 1396–1400.

A. Ahmed A. R M. Kamal H. Wang M. R. Islam, M. A. Kabir and A. Ulhaq. 2018. Depression detection from social network data using machine learning techniques. *Health Inf Sci Syst*, 6.

Francis Bach Matthew D. Hoffman, David M. Blei. 2010. Online learning for latent dirichlet allocation.

David Owen, José Camacho-Collados, and Luis Espinosa Anke. 2020. Towards preemptive detection of depression and anxiety in twitter. *CoRR*, abs/2011.05249.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Viridiana Romero. 2019. Detecting depression in tweets.

Zhang Yun-tao, Gong Ling, and Wang Yong-cheng. 2005. An improved tf-idf approach for text classification. *Journal of Zhejiang University - Science A: Applied Physics Engineering*, 6:49–55.

Jiarui Zhang, Yingxiang Li, Juan Tian, and Tongyan Li. 2018. Lstm-cnn hybrid model for text classification. In *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 1675–1680.

A Appendix A: Kaggle Environment

Kaggle is an online cloud platform for Python based environments. It provides faster GPU for Deep Learning Tasks. It provides 40 hours of free GPU a week, which is needed for experiments such as these. Moreover, many useful python libraries are by default installed that makes development of the projects easier. Kaggle also enables sharing of standard and custom datasets, with access to every user if made public.

B Appendix B: Python Libraries

Scikit-learn: Machine Learning in Python, by Pedregosa et al. (2011) is a python library that provides a number of functions used to perform various Natural Language Processing Algorithms.

1. pad_sequences()
   This function transforms a list of sequences (lists of integers) into a 2D Numpy array of the given shape. It either takes the maxlen argument if provided, or the length of the longest sequence in the list, to perform the padding operation. Sequences are either padded or truncated to fit the desired length given to the function. The position where padding or truncation happens is determined by the arguments padding and truncating, respectively. Pre-padding or removing values from the beginning of the sequence is the default.

2. NMF()
   NMF stands for Non-Negative Matrix Factorization. Scikit-learn library in python provides an in-built method, NMF() for the implementation of this technique that is used for example for dimensionality reduction, source separation or topic extraction. It typically finds two non-negative matrices whose product approximates the non- negative matrix X.

3. LatentDirichletAllocation()
   This method of scikit-learn library is implemented based on the works of Matthew D. Hoffman (2010). Latent Dirichlet Allocation is a generative probabilistic model for collections of discrete dataset such as text collection. It is also a topic model that is used for discovering abstract topics from a collection of documents Matthew D. Hoffman (2010).

C Appendix C: Team Contribution

1. Dipan Shah
   - Studied several related research papers and discussed with group
   - Gave comparison of models

- Specifically studied over Machine Learning models for text classification
- Final report editing

2. Dhruva Shah
    - Worked on previous work
    - Dataset collecting and preprocessing
    - Mainly saw analysis of SVM how vectorization works
    - Perform LSTM model with topic modeling
    - Final report editing

3. Krishna Gandhi
    - Studied different word- embedding techniques
    - Studied LSTM and CNN
    - Worked on developing UI
    - Perform LSTM and CNN without topic modeling
    - Final Report editing

4. Palak Patel
    - Studied topic modeling
    - Studied LSTM and CNN
    - Studied different word- embedding techniques
    - Implemented CNN model with topic modeling
    - Final report editing

5. Sachin Singh
    - Dataset collection and preprocessing
    - Studied topic modeling
    - Perform LSTM and CNN without topic modeling
    - Worked on developing UI
    - Final report editing