Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 235 (2024) 1353–1363

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering (ICMLDE 2023)

Stratification of Depressed and Non-Depressed Texts from Social Media using LSTM and its Variants

Keerthan Kumar T G[1], Anoop R[2], Shashidhar G Koolagudi[3], Trupthi Rao[2], Ashwini Kodipalli[2].

[1]*Department of Information Science and Engineering, Siddaganga Institute of Technology, Tumkur, India.*

[2]*Department of artificial intelligence and Data Science, Global academy of Technology, Bengaluru, India.*

[3]*Department of Computer science and Engineering National Institute of Technology, Karnataka, Surathkal, India.*

Abstract

This work examines the performance of various LSTM (long short-term memory) variants on social media text data. This study evaluates the performance of LSTM models with different architectures, namely, classic LSTM, Bidirectional LSTM, Stacked LSTM, gated recurrent unit (GRU), and bidirectional GRU, on a social network dataset comprising texts extracted from multiple social media platforms. We aim to identify the most effective LSTM variant of the five considered LSTM models for text analysis through a comparative study of the models' precision, recall, F1-score, and accuracy. The research findings show that the Classic LSTM and the GRU model perform better than the other models in accuracy. In contrast, the bidirectional models (Bidirectional LSTM and Bidirectional GRU) provide better precision scores than their respective primitive models. This research has significant implications for developing more efficient models for natural language processing applications. It offers beneficial insights into the implications involving the scrutiny of depression on social media platforms through text data analysis.

Keywords: Depression; Deep-Learning; Gated Recurrent Unit; Long Short Term memory; Performance analysis;

1. Introduction

Depression has been a prime mental health issue affecting a huge part of the human population over the years across the globe. A study states that every fifth person has experienced depression [1]. 4.5 percent of the world's population suffers from this condition. In many nations, depression remains underdiagnosed and undertreated, resulting in a negative view of oneself and, in extreme cases, suicide. Early diagnosis and efficient treatment of depression is imperative in managing the condition and providing improved quality of life for the individuals suffering from it. A growing amount of attention is being focused on the identification of signs of depression using the text data generated on social media platforms. Long Short-Term Memory (LSTM) models have drastically revolutionized the domain of Natural Language Processing (NLP), enabling effective prediction and analysis of sequential data such as text [18]. Integrating Deep learning models like LSTM in clinical practice can enable early depression detection, intervention, and referrals. Network administrators can use these models to monitor social media for signs of depression, offering proactive support through relevant content and recommendations. The collaboration between technologists and mental health experts benefits individuals by leveraging technology for early detection and clinical expertise for support. Online peer groups and social media serve as digital platforms for individuals suffering from depression, providing them with knowledge, experiential guidance, and support [2]. This research paper presents a comparative study of a set of the selected LSTM variants applied on a labelled classification dataset comprising text data from multiple social media platforms classified into depressive and non-depressive texts. The objective here is to evaluate the performance of the selected LSTM variants in identifying depression in social media text data.

2. Literature Survey

This literature survey attempts to investigate existing research on the application of various LSTM models in NLP applications, focused primarily on text data analysis. Multiple studies have been previously performed in an attempt to study LSTM models for NLP tasks like machine translation, sentiment analysis, and text classification, emphasizing their effectiveness in achieving good performance and high accuracy. In [3], the authors have chosen and trained the model with the twitter data. The data is classified as balanced and imbalanced data, where the former is further classified into positive tweets and negative tweets, and in later data, the data without missing values is oversampled. The data obtained is processed using various ML algorithms such as support vector machine (SVM), decision tree, K-nearest neighbour (KNN), and LSTM, out of which LSTM outperforms all other algorithms on both balanced and imbalanced data with an accuracy of 83%. In [4], the authors have developed their own dataset with the raw data obtained from Twitter. The obtained unlabelled data is further transformed into labelled data by applying data preprocessing techniques. Some of the deep learning (DL) models, such as gated recurrent unit (GRU), recurrent neural network (RNN), and convolutional neural network (CNN), are used to make predictions on the data. The word-based GRU is the best-performing model with 98% of accuracy. Further, in [5], the authors extracted the Reddit data. The main goal is to detect the depressed attitude in the posts of Reddit users with the help of NLP and text classification approaches. The feature extraction is done, and features such as N-grams, linguistic inquiry and word count (LIWC), and linear discriminant analysis (LDA) are extracted. Later, different ML classifiers such as SVM, logistic regression, random forest, and MLP are used to model the features. The individual features are processed by each classifier, where LIWC + LDA + N-grams modelled by MLP give the maximum accuracy of 91%.

Further in [6], the authors extend the research to audio classification as well. Both text and audio are used as datasets. A separate model is chosen to model each data. The text is analyzed using one unimodal, and the audio is analyzed with another unimodal. In both cases, CNN and LSTM are used. The data is first fed into the CNN, and the CNN layer's output is summarized and fed as input to the LSTM, which gives an F1-score of 0.8 on the non-depressed class of test data for the text analysis, while an F1-score 0f 0.75 was achieved of the voice quality model. Further, in [7], the authors attempt to attain the utmost accuracy by modelling the data with the help of LSTM and RNN. The features are visualized with the help of PCA, a dimensionality reduction technique. The model is implemented such that LSTM contains two hidden layers, and RNN is implemented with two dense layers. The LSTM-RNN model is used to differentiate between depressed and non-depressed text, wherein this framework gives 99% accuracy. In [8], the authors investigate the feasibility of predicting a user's mental state using Twitter data by distinguishing between depressed and non-depressive tweets. The authors propose a hybrid model that combines CNN and bi-directional Long Short-Term Memory (biLSTM) architectures on a benchmark depression dataset incorporating tweets, attaining an accuracy of 94.28%. They compare their model's performance to other models and baseline techniques, proving its efficacy in forecasting depression. The research looks at how depressed and non-depressive material is represented linguistically on social media sites. In recent research, [9] utilizes an LSTM deep recurrent network to analyze Bangla social media data, focusing on hyper-parameter tuning to enhance the accuracy of depression detection. This study provides valuable insights, particularly beneficial for psychologists and researchers seeking to identify signs of depression in individuals through their virtual social interactions. Additionally, [10] introduces a novel approach with their Deep-Knowledge-aware Depression Detection system, underlining the significance of incorporating domain knowledge for more effective depression detection. Empirical findings from this work emphasize the substantial performance improvement achieved by integrating domain knowledge, highlighting its potential to advance knowledge-aware machine learning and support large-scale mental health assessments by offering early detection and factor explanation capabilities [19][20].

3. Dataset

The dataset for the study was obtained from Kaggle, a renowned online platform for data science projects. Our dataset combines tweets [11] and Reddit messages [12] obtained from two distinct datasets, respectively, and classified as depressive and non-depressive. The dataset comprises 12,570 rows where each of the messages classified as depressive is labelled with the value 1, and those classified as non-depressive are labelled with the value 0. As this study looks to explore a path in the analysis of depression in social media users based on the kind of content the users interact with rather than the kind of content they share on these platforms, unlike the previous studies on similar grounds, the selection of the messages to be labelled as depressive was based on the accounts sharing depressive content, for example, Twitter accounts having the user-names like "Depression Quotes," "Depressive Quotes" etc. The selection of the messages labelled as non-depressive were obtained from media accounts sharing feeds over a wide range of topics such as 'Sports,' 'News', and other users known to the dataset providers.

4. Proposed Methodology

The research was conducted using the Python programming language on the Google Colab platform with T4 GPU, which provides access to cloud-based computing resources. The proposed methodology provides a systematic and rigorous approach to conducting a comparative study of multiple LSTM variants on a dataset comprising social media text data.

4.1 Data Pre-processing

The data was handled to remove extraneous information like URLs, special characters, and numeric values. The messages were tokenized, and the stop words were omitted at the end. The data was lemmatized to decrease the count of unique words and improve the model's performance. In an attempt to negate the slight imbalance encountered in the dataset, with approximately 53% of the data labelled as "not depressed" and 47% as "depressed.", we employed the Synthetic Minority Over-sampling Technique (SMOTE). In this resampling technique, we oversampled the "depressed" class of data to create synthetic samples to match the size of the majority class ("not depressed"). The pre-processed text data was then converted to numerical representation using one-hot encoding. Finally, the data was split in the ratio 60:20:20 into the training set, testing, and validation set, respectively. The model was trained using the training set, and its performance was gauged using the testing set. The validation set is used to assess and improve the model's performance during training.

4.2 Model Selection

Five popular LSTM variants were used in the study; they are: (i). Classic LSTM (ii). Stacked LSTM (iii). Bidirectional LSTM (iv). GRU (v). Bidirectional GRU. Each of these models was implemented using the Keras deep learning library. The following sections comprise a detailed description of the above-listed models.

4.2.1 Classic LSTM

LSTM is an RNN (recurrent neural network) that is programmed to interpret long-term dependencies from sequential information. This is accomplished through the introduction of a memory cell that stores information for a long period of time. The core principle behind an LSTM is to keep a cell state that can selectively forget or recall information based on the input and the prior concealed state at each time step. Fig. 1 depicts the surface architecture of a classic LSTM cell, and Fig. 2 depicts the single-cell architecture of LSTM.

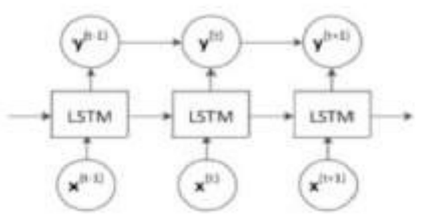

Fig. 1. Surface architecture of LSTM [14]

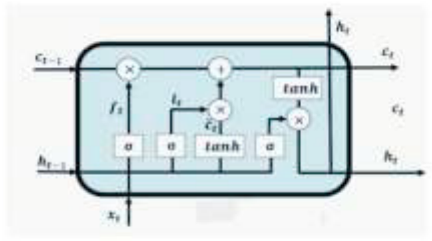

Fig. 2. Single cell architecture of LSTM [16]

The architecture of a classic LSTM model comprises a sequence of LSTM cells that process input sequentially. Each cell has three gating mechanisms. The forget gate determines whatever information from the previous time step is to be ignored. It takes the previous hidden state and the current input as input and returns a number between 0 and 1 for each hidden state element. The closer the value is to one, the more likely it is that the information should be kept, while the closer value is to zero means that the information should be forgotten. Eq. (1) denotes the workflow of the forget gate of a classic LSTM model.

$$f_t = sigmoid\left(W_f * [H_{t-1}, X_t] + b_f\right) \qquad (1)$$

where f_t is the forget gate's output information. H_{t-1} represents the previous hidden state and X_t represents the current input. b_f and W_f are the learnable biases and weights of the forget gate.

The input gate determines the new information that needs to be added to the memory cell. It accepts the previous concealed state and the current input as input and returns a value ranging from 0 to 1 for each memory cell element. A number close to 1 indicates that the information should be added, whilst a value close to 0 indicates that the information should be ignored. Eq. (2) denotes the workflow of the input gate of a classic LSTM model.

$$i_t = sigmoid(W_i * [H_{t-1}, X_t] + b_i) \qquad (2)$$

where i_t is the input gate's output information. H_{t-1} represents the previous hidden state and X_t represents the current input. b_i and W_i and are the learnable biases and weights of the input gate.

The output gate determines which information from the memory cell should be given out as the output. The previous concealed state and the current input are used as input to generate a number between 0 and 1 for each memory cell element. A number close to 1 indicates that the data should be given out as the output, while a value nearer to 0 means the information should be suppressed. Eq. (3) denotes the workflow of the output gate of a classic LSTM model.

$$o_t = sigmoid(W_o * [H_{t-1}, X_t] + b_o) \qquad (3)$$

where o_t is the output gate's output information. H_{t-1} represents the previous hidden state and X_t represents the current input. b_o and W_o and are the learnable biases and weights of the output gate.

The LSTM cell, in addition to the three gating mechanisms, features a memory cell that may store information over time, based on the information from the input and the forget gate, the cell state (C_t) is updated.
Eq. (4) denotes the updation of the cell state.

$$C_t = f_t * C_{t-1} + i_t * \tanh(W_c * [H_{t-1}, X_t] + b_c \qquad (4)$$

where $C_{t-1}$is the previous cell state, W_c and b_c are the learnable weights and biases. Tanh is the hyperbolic tangent activation function.

Finally, the hidden state (H_t) is processed based on the output gate and updated cell state as denoted in Eq. (5).

$$H_t = o_t * \tanh(C_t) \qquad (5)$$

To acquire the final prediction, the output of the last LSTM cell is normally transmitted through a fully linked layer.

4.2.2 Stacked LSTM

The Stacked LSTM is a RNN with an architecture designed to interpret long-term dependencies from sequential information. This is accomplished by stacking numerous LSTM layers on top of one another to acquire progressively sophisticated representations of the input sequence. Each LSTM layer processes the input sequence and forwards the results to the next layer in the stack. The last LSTM layer generates the final output of the network.

The stacked LSTM equations are identical to those of the standard LSTM, but with the addition of additional layers. Each layer takes the previous layer's output as input and produces an output for the next layer. Let's suppose a model of the stacked LSTM comprises of n layers. The equation for each of the gates at the k-th layer of the stacked LSTM, where k = 1,2,3, 4…. n is as follows:
The input gate's workflow at the k-th layer of a stacked LSTM model is represented in Eq. (6).

$$i\_k_t = sigmoid(W_{i_k} * \left[H_{k_{t-1}}, H_{\{k-1\}_t}\right] + b_{i\_k}) \qquad (6)$$

where i_k_t is the output information of the input gate, $H_{k_{t-1}}$ represents the previous hidden state, $H_{\{k-1\}_t}$ represents the hidden state from the previous LSTM layer.

Eq. (7) denotes the workflow of forget gate at the k-th layer of a stacked LSTM model:

$$f\_k_t = sigmoid(W_{f_k} * \left[H_{k_{t-1}}, H_{\{k-1\}_t}\right] + b_{f\_k}) \qquad (7)$$

where f_k_t is the output information of the forget gate, $H_{k_{t-1}}$ represents the previous hidden state, $H_{\{k-1\}_t}$ represents the hidden state from the previous LSTM layer.

Eq. (8) is a representation of the workflow of the output gate at the k-th layer of a stacked LSTM model:

$$o\_k_t = sigmoid(W_{o_k} * \left[H_{k_{t-1}}, H_{\{k-1\}_t}\right] + b_{o\_k}) \qquad (8)$$

where o_k_t is the output information of the output gate, $H_{k_{t-1}}$ represents the previous hidden state, $H_{\{k-1\}_t}$ represents the hidden state from the previous LSTM layer.
The updating of the cell state (C_k_t) can be denoted using Eq. (9):

$$C\_k_t = f\_k_t * C\_k_{t-1} + i\_k_t * \tanh(W_{c\_k} * [H\_k_{t-1}, H\_\{k-1\}_t] + b_{c\_k}) \qquad (9)$$

Based on the output gate and the updated cell state, the hidden state (H_k_t) is processed as denoted in Eq. (10)

$$.H\_k_t = o\_k_t * \tanh\left(C_{k_t}\right) \qquad (10)$$

In the above mentioned equations W_{i_k}, W_{f_k}, W_{o_k}, W_{c_k}, b_{i_k}, b_{f_k}, b_{o_k}, b_{c_k}, are all learnable weights and biases of the respective gates specific to the k-th layer. Fig. 3 represents the surface architecture of stacked LSTM.

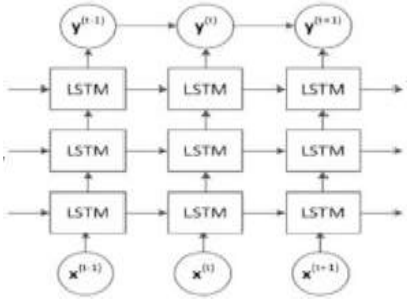

Fig. 3. Surface architecture of Stacked LSTM [14]

4.2.3 Bidirectional LSTM

Bidirectional LSTM [13] is a type of RNN with an architecture to capture both the past and future contexts of an input sequence. The model achieves this with the help of two distinct LSTM networks, one that processes the sequence of inputs in the forward direction and the other processes the sequence of inputs in backward direction. The Bidirectional LSTM model's architecture is comprised of two parallel LSTM networks, both reading the input sequence in opposite directions. The output of each LSTM network is concatenated at each time step to obtain the final output.

The Eqs. (11-21) denoting the workflow of bidirectional LSTM at each of the two layers (forward and backward) is as follows:

Forward LSTM

The workflow of the Input Gate (i_t) is represented in Eq. 11

$$i_t = sigmoid(W_i[H_{t-1}, X_t] + b_i \quad (11)$$

The workflow of the forget Gate (f_t) is represented in Eq. 12:

$$f_t = sigmoid(W_f[H_{t-1}, X_t] + b_f \quad (12)$$

The workflow of the output Gate (o_t) is represented in Eq. 13:

$$o_t = sigmoid(W_o[H_{t-1}, X_t] + b_o \quad (13)$$

The Cell State (c_t) is updated as denoted in Eq. (14) :

$$c_t = f_t * c_{t-1} + i_t * \tanh(W_c[H_{t-1}, X_t] + b_c \quad (14)$$

Where c_{t-1} is the previous cell state.

Hidden State (H_t) is updated as denoted in Eq. (15):

$$H_t = o_t * \tanh(c_t) \quad (15)$$

Backward LSTM

The workflow of the Input Gate $(\grave{i_t})$ is represented in Eq. 16:

$\grave{i}_t = sigmoid(\grave{W}_i[\grave{H}_{t-1}, \grave{X}_t] + \grave{b}_i$ (16)

The workflow of the forget Gate ($\grave{f}_t$) is represented in Eq. 17:

$\grave{f}_t = sigmoid(\grave{W}_f[\grave{H}_{t-1}, \grave{X}_t] + \grave{b}_f$ (17)

The workflow of the output Gate ($\grave{o}_t$) is represented in Eq. 18:

$\grave{o}_t = sigmoid(\grave{W}_o[\grave{H}_{t-1}, \grave{X}_t] + \grave{b}_o$ (18)

The Cell State ($\grave{c}_t$) is updated as denoted in Eq. (19):

$\grave{c}_t = \grave{f}_t * \grave{c}_{t-1} + \grave{i}_t * \tanh(\grave{W}_c[\grave{H}_{t-1}, \grave{X}_t] + \grave{b}_c$ (19)

Hidden State (($\grave{H}_t$) is updated as denoted in Eq. (20):

$\grave{H}_t = \grave{o}_t * \tanh(\grave{c}_t)$ (20)

Final Output

The model determines the final output Y_t by concatenating the output of each layer as denoted in Eq. (21):

$Y_t = [H_t, \grave{H}_t]$ (21)

In the above equations W_c, $\grave{W}_c$, W_o, $\grave{W}_{o,}$, $W_{f,}\grave{W}_{f,}W_i$ and $\grave{W}_i$ are all the learnable weight metrices, b_c, $\grave{b}_c$, b_o, $\grave{b}_o$, b_f, $\grave{b}_f$, b_i and $\grave{b}_i$ are the biases, $\grave{X}_t$ and X_t are the input data and H_{t-1} and $\grave{H}_{t-1}$ are the previous hidden state, all respective to their GRU layer and gates.

Each LSTM network of the Bidirectional LSTM model has the same architecture as that of the classic LSTM, with three gating mechanisms (output gate, input gate and forget gate) along with a memory cell. During the training phase, gradients from both the forward and backward LSTM networks are combined to revise the model parameters. Fig. 4 depicts the surface architecture of a bidirectional LSTM model.
$f_t = sigmoid(W_f[H_{t-1}, X_t] + b_f$ (12)

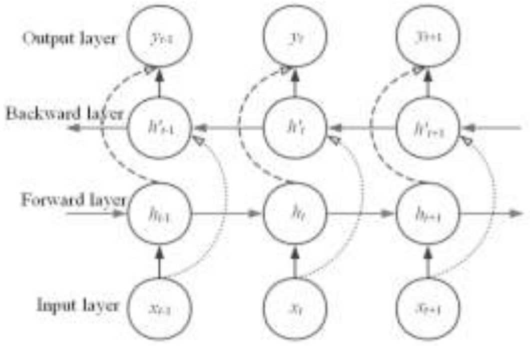

Fig. 4. Surface architecture of Bidirectional LSTM [15]

4.2.4 GRU

Gated Recurrent Unit (GRU) [20,19, 21] is a type of RNN used in sequence-to-sequence learning. GRU, like the LSTM model, employs gating mechanisms to regulate the flow of input across the network. The architecture of the GRU model comprises of a set of interconnected GRU cells that sequentially process the input one step at a time. Each cell has a memory cell and two gating mechanisms, the reset gate and the update gate. The reset gate chooses on what information of the previous hidden state should be neglected, and the update gate controls what proportion of the new information should be added on to the current hidden state. Depending on the needs of input sequence, these gates assist GRU model to selectively retain old information or update its memory cell with new information. Eq. (22) denotes the workflow of the Reset gate (r_t):

$$r_t = sigmoid(W_r[H_{t-1}, X_t] + b_r) \qquad (22)$$

The workflow of the Update gate (z_t) is denoted in Eq. (23):

$$z_t = sigmoid(W_z[H_{t-1}, X_t] + b_z) \qquad (23)$$

The updated Hidden state (H_t) is computed as represented in Eq. (24):

$$H_t = (1 - z_t) * H_{t-1} + z_t * \tanh(W_H[r_t * H_{t-1}, X_t] + b_H) \qquad (24)$$

In the above equations W_r, W_z and $W_H$are the learnable weight metrices, b_r, b_z and $b_H$are the biases, X_t is the input data and H_{t-1} is the previous hidden state. Each GRU cell calculates its output by combining its current memory cell and current input through a non-linear activation function. In turn, this output is sent as input to the next GRU cell. Fig. 5 depicts the surface architecture of a GRU model and Fig. 6 denotes the internal architecture of a single GRU cell.

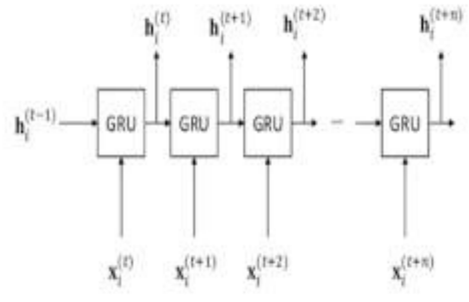

Fig. 5. Surface architecture of GRU [17]

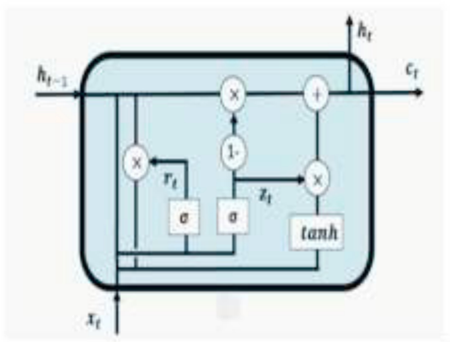

Fig. 6. Single cell architecture of simple GRU [16]

4.2.5 Bidirectional GRU

Bidirectional GRU [21] is a type of RNN architecture that is an extended version of standard GRU, used to process information from both past and future contexts of sequential data. It uses two separate GRU networks where one network is designated to process the inputs in a forward direction while the other network is designated to process the input sequence in a backward direction. Each network in the Bidirectional GRU model has the same architecture as that of the standard GRU, with two gating mechanisms (reset gate and update gate) along with a memory cell. The following equations denoting the workflow of bidirectional GRU at each of the two layers (forward and backward).

Forward GRU

Eq. (25) denotes the workflow of the Reset gate (r_t):

$$r_t = sigmoid(W_r[H_{t-1}, X_t] + b_r) \quad (25)$$

Eq. (26) denotes the workflow of the Update Gate(z_t) :

$z_t = sigmoid(W_z[H_{t-1}, X_t] + b_z)$ (26)

The updated Hidden state (H_t) is computed as represented in Eq. (27):

$$H_t = (1 - z_t) * H_{t-1} + z_t * \tanh(W_H[r_t * H_{t-1}, X_t] + b_H) \quad (27)$$

Backward GRU

Eq. (28) denotes the workflow of the Reset Gate($\grave{r_t}$) :

$\grave{r_t} = sigmoid(\grave{W_r}[\grave{H}_{t-1}, \grave{X_t}] + \grave{b_r})$ (28)

Eq. (29) denotes the workflow of the Update Gate ($\grave{z_t}$):

$\grave{z_t} = sigmoid(\grave{W_z}[\grave{H}_{t-1}, \grave{X_t}] + \grave{b_z})$ (29)

The updated Hidden State($\grave{H_t}$) is computed as represented in Eq. (30) :

$$\grave{H_t} = \left(1 - \grave{z_t}\right) * \grave{H}_{t-1} + \grave{z_t} * \tanh(\grave{W_H}[\grave{r_t} * \grave{H}_{t-1}, \grave{X_t}] + \grave{b_H}) \quad (30)$$

The model determines the final output G_t by concatenating the output of each layer as denoted in Eq. (31):

$G_t = [H_t, \grave{H_t}]$ (31)

In the above equations W_z, $\grave{W_z}$, W_r, $\grave{W}_{r,} W_H$ and $\grave{W_H}$ are all the learnable weight metrices, b_z, $\grave{b_z}$, b_r, $\grave{b_r}$, b_H and $\grave{b_H}$ are the biases, $\grave{X_t}$ and X_t are the input data and H_{t-1} and $\grave{H}_{t-1}$ are the previous hidden state, all respective to their GRU layer and gates. During training, the model parameters are updated by combining the gradients from both the forward and backward GRU networks.

4.3 Model Training

Every very LSTM variant under study was trained on the training set for ten epochs and a batch size of 32. The model is trained with the help of Adam optimizer (a popular optimization algorithm that uses adaptive learning rates to update the model parameters). On the basis of the binary cross-entropy loss function, the error between the actual labels and the predicted labels are calculated.

4.4 Model Evaluation

The conduct of the model was evaluated on the testing set of data using various metrics, including precision, recall, accuracy, and F1-score. In an attempt to visualize the model's performance, the confusion matrix was generated. The results were interpreted to evaluate each LSTM variant and to determine the most effective model for the classification of social media text messages into depressive and non-depressive texts.

5. Results

The results obtained by each of the LSTM variant under study have been tabulated in Table 1. From the this, we can infer that Classic LSTM and GRU provide the same accuracy, but when other parameters such as f1-score, recall and precision are considered, GRU appears more efficient compared to the other procedures.

Table 1. Performance metrics of each model.

Methods/Parameters	Accuracy	Precision	Recall	F1-score
Classic LSTM	95.00%	95.00%	94.00%	95.00%
Stacked LSTM	94.00%	96.00%	96.00%	94.00%
Bidirectional LSTM	94.00%	97.00%	94.00%	94.00%
GRU	95.00%	96.00%	95.00%	95.00%
Bidirectional GRU	94.00%	97.00%	97.00%	94.00%

This output can be owing to certain factors. Even though Classic LSTM and GRU work similarly to each other, GRU has a predominant reputation of being more efficient while computing on a large set of data, which is the case in the acquired dataset for this problem statement. Also, GRU is more productive and accurate when the data is lengthy, and a lot of keywords are used in a single sentence when compared to the other models. This is due to the fact that in GRU, the long-term and short-term memory is combined together, which is then processed through the Update and Forget gate, which keeps track of the amount of memory that should be retained and should be forgotten, respectively. Fig. 7, comprising the confusion matrix, depicts the number of correctly and incorrectly classified data for all the methods used for the classification of data.

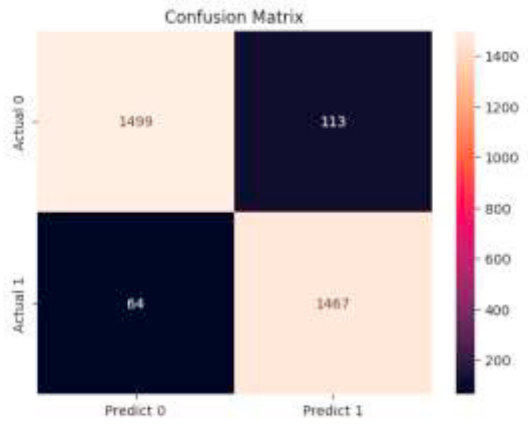

Fig. 7. Confusion matrix depicting the number of classified and misclassified Labels while using the GRU method

6. Limitations

Different social media platforms may have unique linguistic patterns and user behaviours. The research still holds scope for investigation of platform-specific models to account for these variations. Social media text data is inherently diverse and dynamic. Incorporation of a broader range of linguistic and contextual features to enhance model robustness could be considered.

7. Conclusion and Future Scope

This work focuses on developing accurate models for early depression detection using NLP techniques. We conducted a comparative study of five LSTM variants with social media text data, finding that they all performed similarly in accuracy. However, the GRU variant showed slight advantages in handling large datasets and complex phrases. Notably, classic LSTM and GRU models slightly outperformed their bidirectional counterparts, indicating that not all NLP tasks require bidirectional analysis. From this work, we can safely conclude that this work contributes to more reliable and accurate depression diagnosis models through NLP techniques, with practical applications for mental health support. In future, we plan to integrate multimodal data, such as images and voice, to create more comprehensive and accurate models for early depression detection.

References

[1] Naslund, J. A., Tugnawat, D., Anand, A., Cooper, Z., Dimidjian, S., Fairburn, C. G., ... & Patel, V. (2021). Digital training for non-specialist health workers to deliver a brief psychological treatment for depression in India: Protocol for a three-arm randomized controlled trial. Contemporary clinical trials, 102, 106267.
[2] Vydiswaran, V. V., & Reddy, M. (2019). Identifying peer experts in online health forums. BMC medical informatics and decision making, 19, 41-49.
[3] Gupta, S., Goel, L., Singh, A., Prasad, A., & Ullah, M. A. (2022). Psychological analysis for depression detection from social networking sites. Computational Intelligence and Neuroscience, 2022.
[4] Singh, D., & Wang, A. (2016). Detecting depression through tweets. Standford University CA, 9430, 1-9.
[5] Tadesse, M. M., Lin, H., Xu, B., & Yang, L. (2019). Detection of depression-related posts in reddit social media forum. IEEE Access, 7, 44883-44893.
[6] Solieman, H., & Pustozerov, E. A. (2021, January). The detection of depression using multimodal models based on text and voice quality features. In 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus) (pp. 1843-1848). IEEE.
[7] Amanat, A., Rizwan, M., Javed, A. R., Abdelhaq, M., Alsaqour, R., Pandya, S., & Uddin, M. (2022). Deep learning for depression detection from textual data. Electronics, 11(5), 676.
[8] Kour, H., & Gupta, M. K. (2022). An hybrid deep learning approach for depression prediction from user tweets using feature-rich CNN and bi-directional LSTM. Multimedia Tools and Applications, 81(17), 23649-23685.
[9] Uddin, A. H., Bapery, D., & Arif, A. S. M. (2019). Depression Analysis from Social Media Data in Bangla Language Applying Deep Recurrent Neural Networks.
[10] Zhang, W., Xie, J., Liu, X., & Zhang, Z. (2023). Depression Detection Using Digital Traces on Social Media: A Knowledge-aware Deep Learning Approach. arXiv e-prints, arXiv-2303.
[11] Mowery, D. L., Bryan, C., & Conway, M. (2015). Towards developing an annotation scheme for depressive disorder symptoms: A preliminary study using twitter data. In Proceedings of the 2nd workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality (pp. 89-98).
[12] Arumae, K., Qi, G. J., & Liu, F. (2018). A Study of Question Effectiveness Using Reddit" Ask Me Anything" Threads. arXiv preprint arXiv:1805.10389.
[13] Brahma, S. (2018). Improved sentence modeling using suffix bidirectional lstm. arXiv preprint arXiv:1805.07340.
[14] Sahar, A., & Han, D. (2018, August). An LSTM-based indoor positioning method using Wi-Fi signals. In Proceedings of the 2nd International Conference on Vision, Image and Signal Processing (pp. 1-5).
[15] Cheng, H., Xie, Z., Shi, Y., & Xiong, N. (2019). Multi-step data prediction in wireless sensor networks based on one-dimensional CNN and bidirectional LSTM. IEEE Access, 7, 117883-117896.
[16] ArunKumar, K. E., Kalaga, D. V., Kumar, C. M. S., Kawaji, M., & Brenza, T. M. (2022). Comparative analysis of Gated Recurrent Units (GRU), long Short-Term memory (LSTM) cells, autoregressive Integrated moving average (ARIMA), seasonal autoregressive Integrated moving average (SARIMA) for forecasting COVID-19 trends. *Alexandria engineering journal*, *61*(10), 7585-7603.
[17] Alfarraj, M., & AlRegib, G. (2018). Petrophysical property estimation from seismic data using recurrent neural networks. In *SEG Technical Program Expanded Abstracts 2018* (pp. 2141-2146). Society of Exploration Geophysicists.
[18]. Keerthan Kumar, T. G., Himanshu Dhakate, and Shashidhar G. Koolagudi. "IIMH: Intention Identification In Multimodal Human *Utterances." In Proceedings of the 2023 Fifteenth International Conference on Contemporary Computing, pp. 337-344. 2023.*
[19]. Sushma, S. A., and Keerthan Kumar TG. "Comparative Study of Naive Bayes, Gaussian Naive Bayes Classifier and Decision Tree Algorithms for Prediction of Heart Diseases." (2021).
[20]. Varun, J., E. S. Vishnu Tejas, and T. G. Keerthan Kumar. "Performance Analysis of Machine Learning Algorithms Over a Network Traffic." In International Conference on Intelligent and Smart Computing in Data Analytics: ISCDA 2020, pp. 1-10. Springer Singapore, 2021.