SSN@LT-EDI-ACL2022: Transfer Learning using BERT for Detecting Signs of Depression from Social Media Texts

Adarsh S and **Betina Antony**
Department of Computer Science and Engineering
SSN College of Engineering
Kalavakkam, Chennai, India
adarsh19008@cse.ssn.edu.in, betinaantonyj@ssn.edu.in

Abstract

Depression is one of the most common mental issues faced by people. Detecting signs of depression early on can help in the treatment and prevention of extreme outcomes like suicide. Since the advent of the internet, people have felt more comfortable discussing topics like depression online due to the anonymity it provides. This shared task has used data scraped from various social media sites and aims to develop models that detect signs and the severity of depression effectively. In this paper, we employ transfer learning by applying enhanced BERT model trained for Wikipedia dataset to the social media text and perform text classification. The model gives a F1-score of 63.8% which was reasonably better than the other competing models.

1 Introduction

One of the crucial modern world problem that needs attention today is mental health and its wellness. According to GHDX, around 5% of all young adults have depressive disorders (Vieta et al., 2021). About half of them never get it diagnosed or treated. Since the advent of the internet, people have felt more comfortable discussing topics like depression and stress online due to the anonymity it provides (William and Suhartono, 2021). People have come forward to share their mental struggles with others and are ready to seek help. This sharedtask has used data scraped from various social media sites like twitter, reddit to detect signs and the severity of depression symptoms.

The main target of this task was to identify Deep Learning models that performed well in classifying tweets based on the level of severity of depression. The term severity here is based on the presence of certain words and their inclining in the word spectrum for mental health. The textual data contains many hidden patterns and styles that distinctly identifies signs of depression (un Nisa and Muhammad, 2021). In this paper, we designed a transformer model with significant fine-tuning to predict if the context shows moderate, severe or no signs of depression.

2 BERT based Transfer Learning

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained deep learning model that functions on the sequence to sequence learning of text. BERT models have found to be performing well in understanding textual data on depression identification (Martínez-Castaño et al., 2021). The types of BERT model differs based on the number of transformer layers, self-attention layers, number of parameters, types of fine tuning, masking and word embedding and so on. The classifier uses Google's BERT-small model[1] from TFHub pretrained for seqtoseq task and adapts it for text classification with both pooled and sequence type outputs. The model used in this classification system is very similar to García-Pablos et al. (2020). The difference occurs in fine-tuning of the model, where in addition to the dropout layer, a dense classifier layer is added to obtain the final label. The BERT model for Depression Detection is shown in Figure 1.

The Training phase consists of the following steps:

- Pre-Processing: To prepare the input sentence for the BERT encoder, the words are converted to tokens with input ids and tags using standard tokenizer. The labels are also encoded and assigned weights based on the input data distributed.
- BERT Encoding: The next step is to apply the pre-trained model to the current input data. This step tries to map the vector to words in the context with high precision. The BERT

[1] https://tfhub.dev/google/small_bert/

Proceedings of the Second Workshop on Language Technology for Equality, Diversity and Inclusion, pages 326 - 330
May 27, 2022 ©2022 Association for Computational Linguistics

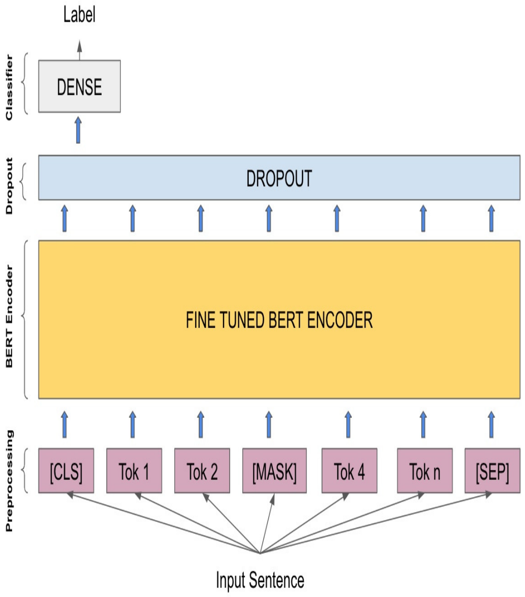


Figure 1: BERT Model for Depression Detection

layer is fine-tuned by adjusting the learning rate and optimization.

- Dropout: The dropout function tends to adjust the weights assigned in each layer so as to normalizing the weights among the words. This layer is significant as it differentiates the words related to depression vs the rest of the words. This step also distributes the weight to avoid overfitting.

- Classifier: The final step is to map the previous layer with 512 words to 3 words ('Severe', ' Moderate' and 'Not depressed') which forms the labels for this task. The system uses a simple dense layer to do that with sigmoid activation function.

3 Experimental Setup

In this section we will see the experimental arrangement of the model, their preprocessing steps and their results and discussion.

3.1 Dataset

The dataset for this test comprised of a training, development and testing data. The training and development data, each of them have 3 columns: PID (post_id), text_data and label. The main content lies in the text_data field. The number of instances for each label is listed in Table 1.

Label	Train	Dev	Test
Not depression	1,971	1,830	NA
Moderate	6,019	2,306	NA
Severe	901	360	NA
Total	8,891	4,496	3,245

Table 1: Class-wise distribution of Dataset instances

Label	Train	Dev	Test
Not depression	1,971	1,830	NA
Moderate	6,019	2,306	NA
Severe	901	360	NA
Total	8,891	4,496	3,245

Table 2: Redistribution of Dataset instances

3.2 Pre-Processing

The dataset provided has a mixture raw text obtained directly from the social media platforms. This data used as such slackened the working of the models as well as the prediction rate. hence a series of refining works were done to the dataset before actual compilation and building.

3.2.1 Removing duplicates and redistribution of dataset

The train and dev sets contain many duplicates. After removing these, the distribution results in just 2720 train cases, 4481 development cases. We chose to combine the train and development sets and perform an 80:20 split for the training and development set. The new class_wise distribution is given in Table 2 (approximate values because of the random split).

3.2.2 Removing stop words

The length of texts on the dataset is quite long. To reduce the model size and improve accuracy, we remove the stop words. The list of stop words is obtained from the Python Natural Language Toolkit and stop_words packages. The length of the text_data is given in Table 3.

This results in a considerable reduction in our text data. We set the maximum length of our models to 300. Shorter sentences are post padded and the longer ones are post truncated. In the case of our BERT Model, the maximum length is 512 words.

3.2.3 Tokenization

The words are tokenized using the TensorFlow tokenizer with a vocabulary size of 1024 words. Since

	Avg. Length of words	Median Length of words
Before	845	572
After	509	349

Table 3: Redistribution of Dataset instances

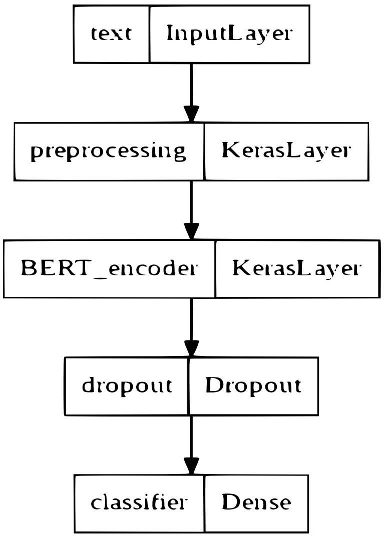


Figure 2: Layers of the Transfer Learning based BERT Model

small_BERT model is used for model building, tokenization is also performed by the pretrained model. In addition, the transfer learning also includes a preprocessing layer which takes care of tokenization. Tokenization is a crucial step for BERT based models as they prepare the data to be processed by segmenting them between the [CLS] and [SEP] tags.

3.2.4 One-hot encoding labels

Label binarizer's encoder function is used to encode our labels as one-hot vectors to perform multi-class classification. In the case of our BERT model, tf.one_hot function is used to encode the labels into tensors with a depth of 3 (Jie et al., 2019).

3.3 Fine-tuned BERT Model

The pre-trained small_Bert model for classification is retrieved from TF Hub. However, to adapt the model to the given dataset, we perform a layer of fine-tuning adding a Dense layer as output layer. The model is trained for 3 epochs with a learning rate of 3e-5. The number of epochs and learning rate can be increased but this will add overhead to the processing time. The details of the different layers of the model is shown in Figure 2 The details of the parameters used in each layer of the BERT model is shown in Figure 3

4 Results and Discussion

The model's functioning for the given dataset was better understood by comparing it's results with other DL models. The details of the other models are

- 3-layer embedded model: This is the first and least complex model. This model has an Embedding layer as input layer. This is passed on to a pooling layer, followed by a dense layer. Finally, another dense layer is used as the output layer. The pooling is done by GlobalAveragePooling for 1D. Since word embedding formed the primitive for many classification algorithms, this model was chosen (Ge and Moh, 2017). Further, the efficiency of this model is often overlooked due to its simplicity.

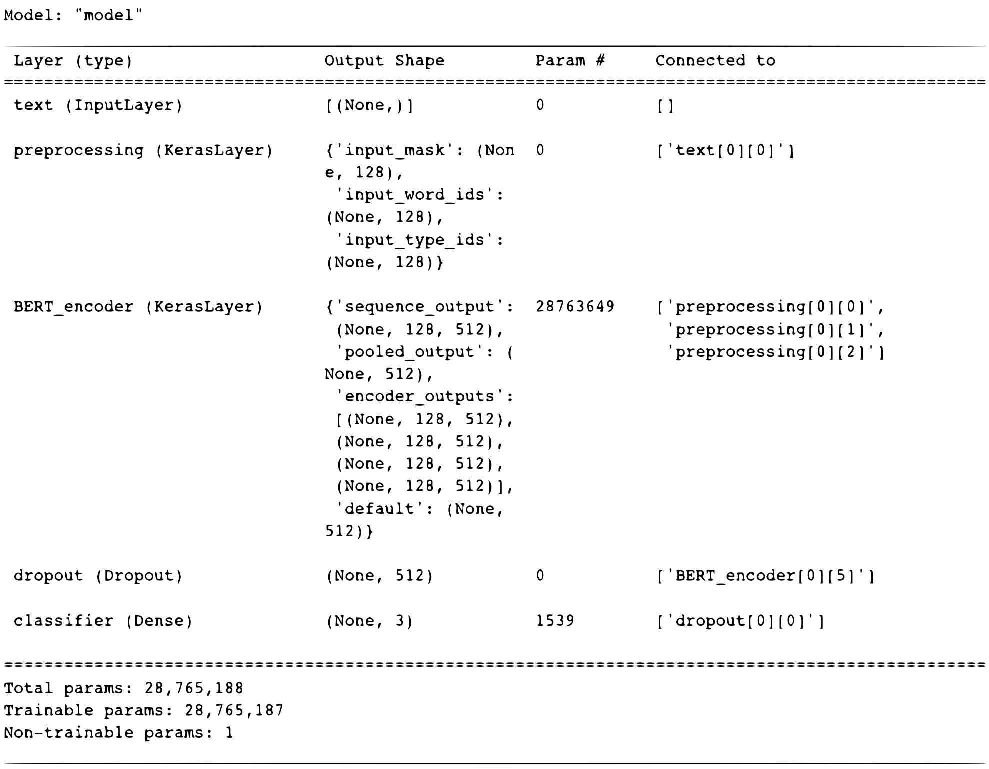
- RNN with Bidirectional LSTM: This model comprises an Embedding layer as input layer, passed on to 2 bidirectional LSTM layers. The output from the LSTMs are then passed on to a Dense layer and an output Dense layer. Dropout is used to counter overfitting. The reason for choosing this model is to deploy the hierarchical nature of LSTM that enhances the performance of contextual understanding and text classification (Yin et al., 2019).

- BERT based Transformer: This model contains the preprocessing layer, BERT based Keras layer and dropout layer. The model is post-processed by adding the final classifier layer. This model was found to perform best in case understanding contexts and sequential operation.

The summary of the three model parameters are shown in Table 4

4.1 Results

The performance scores against the test dataset for different models is listed in Table 5. The evaluation metrics used are Precision (clarity), Recall (coverage) and F1-Score. The metrics are calculated at macro as well as weighted levels. In addition, A measure of accuracy of the system is calculated based on their performance on development as well as test data.



```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 text (InputLayer)              [(None,)]            0           []

 preprocessing (KerasLayer)     {'input_mask': (Non  0           ['text[0][0]']
                                e, 128),
                                 'input_word_ids':
                                (None, 128),
                                 'input_type_ids':
                                (None, 128)}

 BERT_encoder (KerasLayer)      {'sequence_output':  28763649    ['preprocessing[0][0]',
                                 (None, 128, 512),                'preprocessing[0][1]',
                                 'pooled_output': (               'preprocessing[0][2]']
                                None, 512),
                                 'encoder_outputs':
                                 [(None, 128, 512),
                                 (None, 128, 512),
                                 (None, 128, 512),
                                 (None, 128, 512)],
                                 'default': (None,
                                512)}

 dropout (Dropout)              (None, 512)          0           ['BERT_encoder[0][5]']

 classifier (Dense)             (None, 3)            1539        ['dropout[0][0]']

==================================================================================================
Total params: 28,765,188
Trainable params: 28,765,187
Non-trainable params: 1
__________________________________________________________________________________________________
```

Figure 3: Summary of the BERT Model

Model	No of Params	Layer types
Embedding	68,803	Embedding, Pooling, Dense
Bidirectional LSTM	177,155	Embedding, Bidirectional, Dense, Dropout
Fine-tuned BERT	28,765,188	Preprocessing, Encoder, Dropout, Dense

Table 4: Comparison of models used

4.2 Discussion

Tasks like text classification, machine translation and language modeling rely greatly on the use of sequential modeling. Even as RNN and LSTM were perfect for these operations, the computation time taken due to processing of single input at a time led to the popularity of transformer models. Thus the use of BERT is justified in many classification problems due to their efficiency of being pre-trained in large dataset and being deeply bidirectional. BERT's transformer architecture and model size helped it learn the features better. One instance where it was able to predict the label correctly is given below

```
".., i always make the
wrong choices and i can't
get myself out of the
depressed state of mind
and i feel like my life is
over ..."
```

The above sentence was tagged as *moderate* as the context is which the words 'depressed' and 'failure' are used is also studied. The other two models had labelled it as *severe*. The BERT model works well for context with opposite terms as it works on parallel processing of layers.

Since contextual words form the key feature in this learning model, absence of words directly related to depression in the context had am impact on the performance of the model. For instance, the below sentence, though looks like a serious case of self harm, was tagged *moderate* due to the lack of words directly related to depression.

```
"... I'll finally get to
take a breather. Today I
think I'll die."
```

5 Conclusion

Detecting the signs of depression form just a collection of words is a huge accomplishment in the

Model	Accuracy	Macro F1 score	Macro Recall	Macro Precision	Weighted F1 score	Weighted Recall	Weighted Precision
Embedding	0.560	0.432	0.449	0.426	0.574	0.560	0.593
Bidirectional LSTM	0.610	0.466	0.491	0.468	0.610	0.610	0.621
Fine-tuned BERT	0.636	0.531	0.533	0.528	0.638	0.636	0.641

Table 5: Evaluation of the three DL models

field of Artificial Intelligence. This is because, we have come to a point where even machine identifies the emotions of a person from the words he or she speaks. This work produces one such model that is capable of detecting depression by exploiting the efficiency of BERT transformer model. Further for a model pretrained in a completely different context, the fine tuned BERT model performed reasonably well when compare to other Deep learning models such as LSTM and Embedded models. The model could be enhanced further to address superficial and unclear words by understanding the context better and by redistributing the weights among words in the encoder layer.

References

Aitor García-Pablos, Naiara Perez, and Montse Cuadros. 2020. Sensitive data detection and classification in spanish clinical text: Experiments with bert. *arXiv preprint arXiv:2003.03106*.

Lihao Ge and Teng-Sheng Moh. 2017. Improving text classification with word embedding. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1796–1805. IEEE.

Liang Jie, CHEN Jiahao, Zhang Xueqin, ZHOU Yue, and LIN Jiajun. 2019. One-hot encoding and convolutional neural network based anomaly detection. *Journal of Tsinghua University (Science and Technology)*, 59(7):523–529.

Rodrigo Martínez-Castaño, Amal Htait, Leif Azzopardi, and Yashar Moshfeghi. 2021. Bert-based transformers for early detection of mental health illnesses. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 189–200. Springer.

Qamar un Nisa and Rafi Muhammad. 2021. Towards transfer learning using bert for early detection of self-harm of social media users.

Eduard Vieta, Jordi Alonso, Víctor Pérez-Sola, Miquel Roca, Teresa Hernando, Antoni Sicras-Mainar, Aram Sicras-Navarro, Berta Herrera, and Andrea Gabilondo. 2021. Epidemiology and costs of depressive disorder in spain: the epico study. *European Neuropsychopharmacology*, 50:93–103.

David William and Derwin Suhartono. 2021. Text-based depression detection on social media posts: A systematic literature review. *Procedia Computer Science*, 179:582–589.

Shi Yin, Cong Liang, Heyan Ding, and Shangfei Wang. 2019. A multi-modal hierarchical recurrent neural network for depression detection. In *Proceedings of the 9th International on Audio/Visual Emotion Challenge and Workshop*, pages 65–71.