

Unified Fake News Detection using Transfer Learning of BERT Model

Vijay Srinivas Tida

Department of Computer Engineering
School of Computing and Informatics
University of Louisiana at Lafayette
vijaysrinivas.tida1@louisiana.edu

Dr. Sonya Hsu

Department of Informatics
School of Computing and Informatics
University of Louisiana at Lafayette
sonya.hsu@louisiana.edu

Dr. Xiali Hei

Department of Computer Science
School of Computing and Informatics
University of Louisiana at Lafayette
xiali.hei@louisiana.edu

Abstract— Automatic detection of fake news is needed for the public as the accessibility of social media platforms has been increasing rapidly. Most of the prior models were designed and validated on individual datasets separately. But the lack of generalization in models might lead to poor performance when deployed in real-world applications since the individual datasets only cover limited subjects and sequence length over the samples. This paper attempts to develop a unified model by combining publicly available datasets to detect fake news samples effectively. Our experiments are conducted on three publicly available datasets, namely ISOT and others from the Kaggle website. The model is designed with the help of Google's Bidirectional Encoder Representation from Transformers (BERT) base uncased model using a transfer learning approach by using pre-trained weights without changing them during training with preprocessing steps like removing the words whose lengths are less than three. To develop the final model, we fine-tune the pre-trained Google BERT base uncased model on each dataset and select the model with better performance on all three datasets. Our final model is constructed from the hyperparameters obtained from the individual models, which show better performance and are trained on the combined dataset. The results indicate that our proposed finalized model works better than existing machine learning and deep learning models like Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), etc., with an F1-score of 0.97 and accuracy of 97% when combined with all three datasets.

Keywords— Transformers, BERT, pre-trained model, Finetuning, hyperparameters, fake news

I. INTRODUCTION

Fake news can spread wrong information to the public to create problems that can cause wrong opinions, change real perspectives, spread violence, etc. Fake news from various social media platforms has dramatically affected the quality of life in terms of security. Because of the easy accessibility of the internet to the public, there is a higher possibility of sharing fake information, which can cause severe problems [1]. For instance, exposure to fake news will cause political instability that can cause a security threat to the country [2]. Fake news also causes problems to public safety [3]. During the 2016 US elections [4] and the COVID pandemic [5], there was a lot of misinformation spread across the internet. 62% of US individuals will access news from social media [6]. There is a dire need for detecting fake news because of these reasons.

Many sources can help to detect fake news based on the information available in it [7]. Unfortunately, most of these require human involvement to see whether the data sample is fake or not [8]. Fake news detection through Natural Language Processing is necessary to avoid human intervention problems for extracting features. Several proposed machine learning and deep learning models are used

for efficient fake news detection. Long Short Term Memory (LSTM) models showed improved accuracy, but vanishing gradient problems will arise with the more extensive data sequences. Training time will also be longer due to sequential dependency. However, transformers in deep learning solved these problems and showed promising results. The positional encoding and a simpler network model consisting of attention and fully connected layers yielded high performance.

We developed a fake news detector model using a pre-trained Google's BERT base uncased model. It consists of three fully connected layers, dropout layers, activation functions, etc., added at one of the transformer outputs. The transformer model was designed accordingly for an efficient fake news detection task. There are mainly two outputs from the pre-trained BERT model. The first output is considered for the sequence to sequence transformation tasks like sentence translation, tagging, etc. The second output consists of a vector size of 765 used for classifying tasks like fake news detection, sentiment analysis, spam detection, etc. Our model is implemented using the Hugging Face Transformers library [9] and preprocessed input samples before feeding into the pre-trained model. The significant contribution of this work is to show that the pre-trained BERT model does not need the entire input sequences for classification purposes. Therefore, the proposed modified model can also classify tasks like fake review detection, spam classification, etc.

The rest of the paper is organized as follows: Section 2 explains the background and literature reviews, whereas Section 3 explains the methodology, which explains the design process for fake news detection. Section 4 discusses the results, and finally, Section 5 concludes the research work.

II. BACKGROUND AND LITERATURE REVIEW

Fake news detection has become a crucial thing because the internet is becoming an essential thing for all people worldwide. Social media websites like Facebook, Twitter, Whatsapp, and so on are predominantly spreading fake news, presented by Vosoughi *et al.* and Gentzkov *et al.* [10], [11]. Generally, fake news detection is made by classifying the sample that belongs to various domains like politics, entertainment, etc. Then knowledge is imparted to deploy generalized machine learning models. Ahmed *et al.* used the process mentioned above with multiple machine learning models, i.e., K-Nearest Neighbor (KNN), Support Vector Machines (SVM), Logistic Regression (LR), Linear Support Vector Machine (LSVM), Decision Trees (DT) obtained the best accuracy of 92% by combining Term Frequency-Inverted Document Frequency (TF-IDF) as feature extractor and Linear Support Vector Machine as the classifier [12]. Similarly, Shu *et al.* did an extensive survey on different

models using textual and visual features combinations to improve accuracy [12]. Later, Wang *et al.* proposed a new machine learning model by using textual features and metadata with the technique of Convolutional Neural Networks (CNN's) [13]. The final proposed model, after many modifications, used a Bidirectional LSTM model with a fully connected layer, and softmax was used at the output to predict the outcomes of the given samples. The authors focused on the data from the political domain with enhanced input features and achieved the best accuracy of 27.7%. Further, Riedel *et al.* presented a better solution in regards to stance detection labeled four levels: agree, disagree, discuss, and unrelated [14]. The results showed an accuracy of 88.46%. The author's proposed model, which uses linguistic properties for text using Term Frequency (TF) and Term Frequency Inverse Document Frequency (TF-IDF), was used for the feature set and Multilayer Perceptron (MLP) for classification purposes. Ghanem *et al.* proposed a new model with different word embeddings along with n-gram features for stance detection in fake articles and achieved an accuracy of about 48.80% [29]. Ruchansky *et al.* proposed a hybrid deep learning model that uses Capture, Score, and Integrate modules with the help of recurrent neural networks for fake news classification, which achieved an accuracy of about 89.20% [16]. Singh *et al.* fed Linguistic Analysis, and Word Count (LIWC) features to the traditional machine learning methods, Support Vector Machines (SVM's), and showed an accuracy of 87.00% [18]. Jwa *et al.* used a transformer model of BERT and showed an increase at 0.14 of F1-score over the existing models by using Stack LSTM and featMLP [17]. Szczepański *et al.* used BERT and Bidirectional LSTM model used only one dataset [18] from Kaggle and showed 0.98 F1-score [19]. Kaliyar *et al.* proposed the FakeBERT model using only one dataset but showed an accuracy of about 98.9% [20]. Ahmad *et al.* used the ensemble methods by combining three publicly available datasets and showed an accuracy of 91% for the combined dataset [21]. Vijay *et al.* proposed the universal model for spam detection in which four datasets were fed to the model for training by finetuning the BERT uncased pre-trained classifier [22]. The obtained hyperparameters from individual models with acceptable performance helped train the final model by combining all the samples. The proposed finalized model showed an accuracy of about 97%.

Transformers played a crucial role in Natural Language Processing (NLP) applications because it has less training time and is working very well for the more extended sequence of data without any vanishing gradient descent problem. The shorter training time is due to the incorporation of position encoding in the architecture. The attention model, along with fully connected layers, helps solve the vanishing gradient problem. To understand the BERT architecture clearly, we need to know the background related to this topic. The sections from A to F will explain the required knowledge to understand the trained model better.

A. Transformers

Computation load from sequential models has become a serious problem for many NLP tasks over the past decade [23].

Transformers showed the solution to this problem with a more straightforward structure, and there is no need for convolutional and recurrent layers. This significant change aids in reducing the delay to more extent with the help of position embeddings [24]. The multi-headed attention mechanism helps improve the efficiency of the model. Transformer models revolutionized the NLP area with less training time. Several Transformer models were released every year [25], [26], [27], [28]. Among these models, BERT and GPT-2 (Generative Pre-Trained Transformer) showed a more significant impact for NLP applications [29], [30].

B. Self-attention

The self-attention mechanism helps identify the interdependence of tokens for the given sentence. It encodes the input sequence into the token with the help of the tokenizer. The three weight matrices query, key, and value are learned during the training process. In Google's BERT, multiheaded self-attention was used, which is an extension of the self-attention mechanism. The user can set the number of heads according to the requirement of the model [24]. This entire procedure will be taken care of Hugging Face Transformers library.

C. Transfer learning

Pre-trained models help users train their model for their downgrading tasks. Usually, a huge corpus of data like BookCorpus helps train these pre-training models [31]. In addition, the weights obtained from these models contain information about the context or words used in the input samples. By finetuning these models by adding layers to the vector obtained from the pre-trained model, users can get their desired results [32]. This paper used pre-trained weights from the original Google's BERT base uncased model without changing them during the training phase.

D. Parameters and hyper-parameters

Parameters are the trainable weights where the model learns during the training process, whereas hyper-parameters are fixed values that the user sets at the beginning of the training process such that it improves model performance [33].

E. Activation Functions

Every neuron in each layer is associated with a corresponding activation function. Furthermore, these functions are non-linear and differentiable. These properties will help the model learn complex features and make the backpropagation possible for updating the weights for better models [34]. Here in this model, we used two activation functions, namely ReLU and log softmax. The mathematical equation for ReLU is stated in Equation 1 as below:

$$f(x) = \max(0, x), \quad (1)$$

where $f(x)$ is ReLU activation and x is considered input for the function.

The softmax activation function used for multi-class classification problems [38]. The equation for the softmax function is stated in Equation 2 as below:

$$f(x_i) = \frac{\exp(x_i)}{\sum_j(\exp(x_j))}, \quad (2)$$

where $f(x_i)$ is softmax activation, x_i is considered input of the function, and j is the number of classes.

The log softmax function extends the softmax that will apply log to the existing softmax function [39]. The log softmax function is stated in Equation 3 as below:

$$f(x_i) = \log \left(\frac{\exp(x_i)}{\sum_j (\exp(x_j))} \right), \quad (3)$$

where $f(x_i)$ is log softmax activation, x_i is the input to the function, and j is the number of classes.

Log softmax function showed better performance when compared to the sigmoid function because of probability outputs from each node.

F. Loss Function

After the forward pass of the model, the loss function was used from the output after the log softmax function for error calculation, which is considered a loss and based on the derivative value concerning model weights, the backpropagation process is done [40]. Here in this model, the cross-entropy loss is used and stated in Equation 4 as below:

$$\text{Cross entropy loss} = - \sum_j (y_j) \log((p_j)), \quad (4)$$

where y_j is the actual class label and p_j is the predicted probability of the j^{th} class.

III. METHODOLOGY

A. Datasets description

We used three publicly available datasets for this project: ISOT and the other two from the Kaggle website. For training the model, the samples needed to have features labeled. The datasets contain both real and fake news articles from various domains. The real world signifies the real samples, where rumors or fake news spread through the internet represent fake samples. These samples are labeled manually with fact-checking websites like politifact.com and snopes.com.

The first dataset used for our analysis is the “ISOT Fake News Dataset,” containing real and fake sample articles derived from the internet. The real samples got from reuters.com, while fake samples were obtained by flagging the fake articles by politifact.com. The dataset consists of 21,417 real articles while 23,481 fake articles, which gives 44,898 articles. The data samples in this dataset are mainly related to political news, but some are from other domains [35].

The second dataset used here is from the Kaggle website. This dataset contains 20,386 samples that include information from various domains like entertainment and sports, not only related to politics [19].

The third dataset used here is also from the Kaggle website. It contains a total of 3,352 samples. Various authenticated online sources like CNN, the New York Times, Reuters, etc., are used to label real samples. Similarly, untrusted websites are used to label fake samples. In this dataset, these samples cover sports, entertainment, and politics [36].

The samples from all three datasets are combined to get the larger dataset termed the fourth dataset. This dataset will cover a wide range of domains. The combined dataset will

have different sample distributions such that the model has to be designed accordingly by considering this condition without biased.

The model is a supervised algorithm in which we labeled '0' for real news, while '1' for fake news: the content and the encoded labels from the samples trained and evaluated the model's performance.

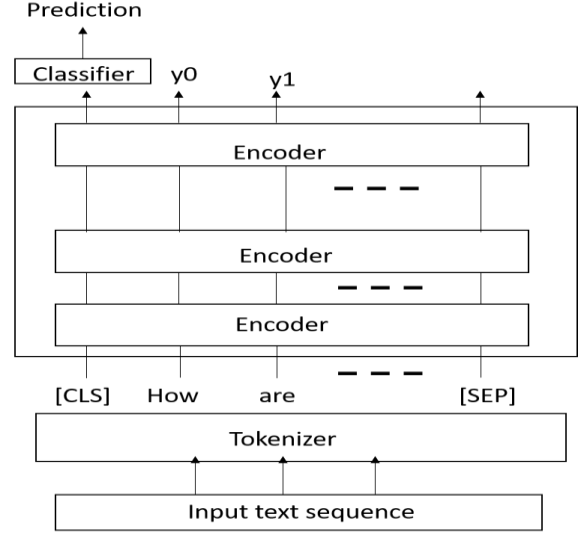


Figure. 1. Architecture of Google's BERT

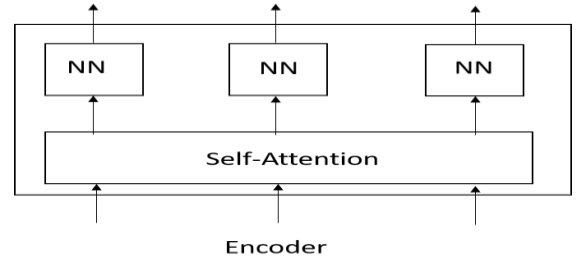


Figure. 2. Encoder internal structure

B. Model Selection and architecture description

1. Model Selection

Pre-trained model selection is considered an essential task for any classification task. However, data preprocessing is necessary for NLP applications for any model designed from scratch. The main advantage of using pre-trained models is that many preprocessing tasks are inbuilt. For example, Google's BERT model states that there is no need to preprocess input data. The authors suggested this because they designed a tokenizer with good results. Unfortunately, this tokenizer will not reduce the sequence length. Google's BERT with many versions is very popular for applications like sequence-to-sequence modeling and classification tasks. As the fake news detection task is considered a classification task, we use the output corresponding to the classifier part. Therefore, the base model is best suited for our task, which contains 12 encoders with 110 million pre-trained weights since it contains fewer parameters [29].

2. BERT model architecture

Transformers generally contain both encoder and decoder units, whereas BERT has only the encoder part. The overview of the encoder's BERT architecture and internal structure of

the encoder can be seen in Fig.1 and Fig.2, respectively. The input samples are fed into the tokenizer. The tokenizer preprocesses the input samples according to the model requirement. In the end, the tokenizer adds the [CLS] and [SEP] tokens to indicate the start and end of the sequence. After this, the data is fed into the encoder block stacked together. The internal structure of the encoder contains attention and feed-forward neural networks, as shown in Fig. 2. The outcome after the last encoder consists of a sequence and a vector of length 768. We use vector output for the classifying task and designed a neural network after this vector. The sequence outputs are used for named entity recognition, parts of speech tagging, etc. We ignore the outcomes related to sequence to sequence modeling for classifying applications.

C. Workflow

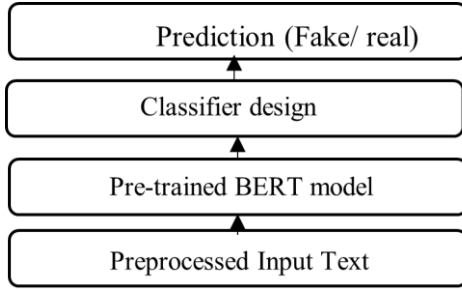


Figure 3. Workflow structure for the modeling process

The workflow for building the model is shown in Fig 3. and the detailed process is explained below:

1. Preprocessed input data is prepared by removing the words less than 3.
2. These processed data samples are fed into the pre-trained BERT model to produce two output vectors.
3. The output vector related to classification is then fed to the designed classifier.
4. Finally, the model evaluation is made with the help of metrics like accuracy, precision, recall, and F1-score by varying the classifier model.
5. Repeat this process for all three individuals such that the model should be the same for all the three datasets with similar hyperparameters.
6. After obtaining the hyperparameters from the individual models the same process is followed for designing the unified model with the combined dataset.

D. Improvement techniques

Model performance is improved with the help of Xavier initialization [37] and an Adam optimizer with a learning rate of 0.003 [38]. The addition of these two methods help the model to converge faster. Finally, to avoid an exploding gradient problem, we use the gradient clipping technique with a clip value set to 1. The model weights with the best validation accuracy are used for subsequent epochs during the training phase.

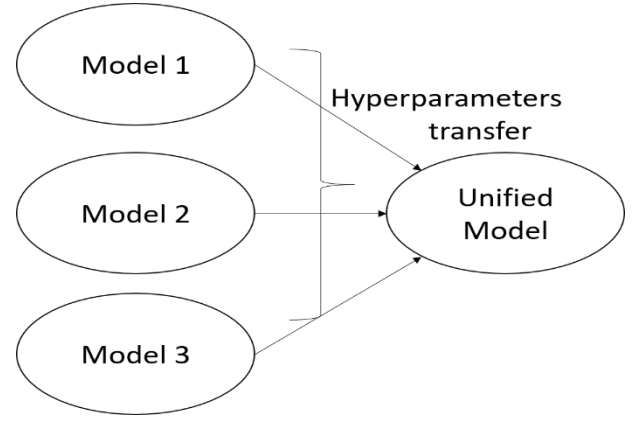


Figure 4. Final modeling selection process

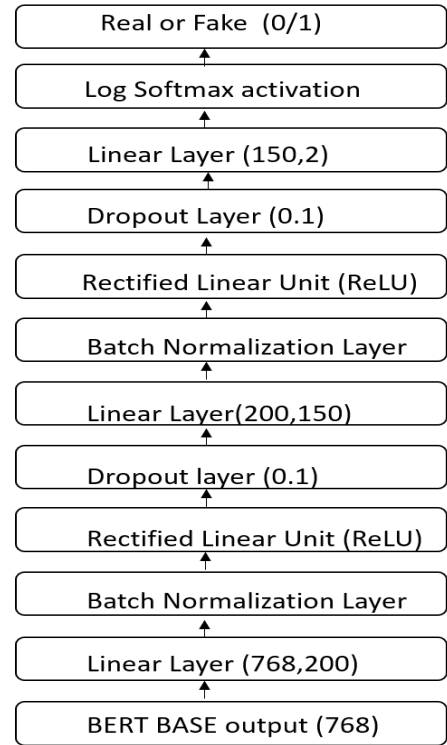


Figure 5. Final model architecture for classifier

E. Final Modeling process

Designing the model is divided into two phases, as shown in Fig 4. The models are developed on individual datasets during the first phase, and the performance is evaluated. Then, individual models are trained such that hyperparameters are the same with acceptable performance. While designing individual models, minibatch size is varied for better performance due to different dataset sizes. Except for the minibatch size, all other parameters are the same for three individual models. The unified model is obtained during the second phase by transferring hyperparameters obtained from the individual models. Again, minibatch size is varied accordingly to have a better performance. Note that the individual models are designed for specific datasets, and then with the information obtained from individual models, the unified model is created accordingly.

Hyperparameter tuning is essential to obtain better performance for the given model. In this model, sequence length, selection of linear layers, the number of neurons in each layer, optimizer selection, learning rate, minibatch size, epochs are considered hyperparameters. The process keeps the pre-trained model with the same weights throughout the training. The finalized obtained model with all hyperparameters is the same except for minibatch size. This is due to varying sizes among the three datasets.

F. Final classifier architecture

Fig 5 depicts the finalized model architecture for the classifier obtained after analyzing the model with different hyperparameters. First, the output vector of 768 from the BERT pre-trained model is fed into a linear layer with 200 neurons. Batch Normalization with Rectified Linear Unit (ReLU) and the dropout layer with a dropout rate of 0.1 is added over the output. A similar structure is repeated once but changed to 150 neurons in the linear layer. At the final stage of the finetuned classifier, a linear layer is added, which contains 2 neurons, and activation function log softmax is applied to the output. In the end, a linear layer with 2 neurons and a log SoftMax function are added to classify whether the given input sample is fake or real. Finally, the unified model is trained using the combined dataset with the same hyperparameters obtained from individual models.

Google's BERT paper didn't recommend using preprocessing steps to the input data [29]. The words less than 3 will not signify any valuable information for classification purposes. Removing these words helped the model converge faster. The model's training time is reduced to almost half by eliminating the input samples' words whose length is less than 3. This advantage is obtained by lowering the sequence length from 300 to 175.

IV. RESULTS

This section will explain the performance metrics and results. Different batch sizes analyze the results from 16, 32, 64, 128, 256, 512, 1024 with 50 epochs. The performance metrics to evaluate the proposed model are explained below:

Table1. Confusion Matrix

| Real Result | Test Result Predicted | |
|--------------|-----------------------|-----------|
| | Real samples | Fake news |
| Real samples | TN | FP |
| Fake news | FN | TP |

A. Confusion Matrix

From table1, we can state the four terms in the confusion matrix as below:

- a) TN: True Negative has considered the total number of real news samples predicted correctly.
- b) TP: True Positive has considered the total number of fake news samples predicted correctly.
- c) FP: False Positive is regarded as the total number of fake news samples predicted as real samples.

- d) FN: False Negative is the total number of real news samples predicted as fake news samples.

B. Accuracy

The accuracy metric analyzes the trained model based on the test samples. Accuracy is defined as the number of samples classified correctly from the total number of samples. Accuracy can be expressed mathematically in the equation below [46]:

$$Accuracy = \frac{(TN+TP)}{(TP+FN+FP+TN)} \quad (5)$$

C. Recall

The recall metric can be defined as the number of fake news samples correctly predicted among all the truly fake news samples. Recall can be expressed mathematically in the equation below [39]:

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

D. Precision

The precision metric can be defined as the number of fake news samples predicted among the total samples predicted as fake news samples. Precision can be expressed mathematically in the equation below [39]:

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

E. F1-Score

F1-score can be defined as the harmonic mean of precision and recall value. F1- score can be expressed mathematically in the equation below [39]:

$$f1\text{-score} = \frac{2 \times (precision \times recall)}{(precision + recall)} \quad (8)$$

To measure the performance of the finalized model, we varied batch size, the number of epochs, and train-valid-test data distributions. However, for comparison purposes, we made the train-test distribution as 90:10 so that the designed model can be compared easily with the prior state of work. Table 2 from Appendix A, accuracy, F1-score, precision, and recall values are compared. The results are obtained from [21] for different datasets as the authors proposed the various machine learning algorithms and benchmark algorithms. The results show that the proposed model achieves superior performance compared to other implementations. The results are noted for different machine learning and deep learning models compared with our proposed model with corresponding minibatch size, which showed the best performance. From Table 2, the accuracy values obtained from the finetuned BERT model showed superior performance when compared to the existing benchmarks for all four datasets. However, Random Forest, Perez LSVM, boosting based on XGBoost, and bagging classifier based on decision trees show high performance on specific datasets.

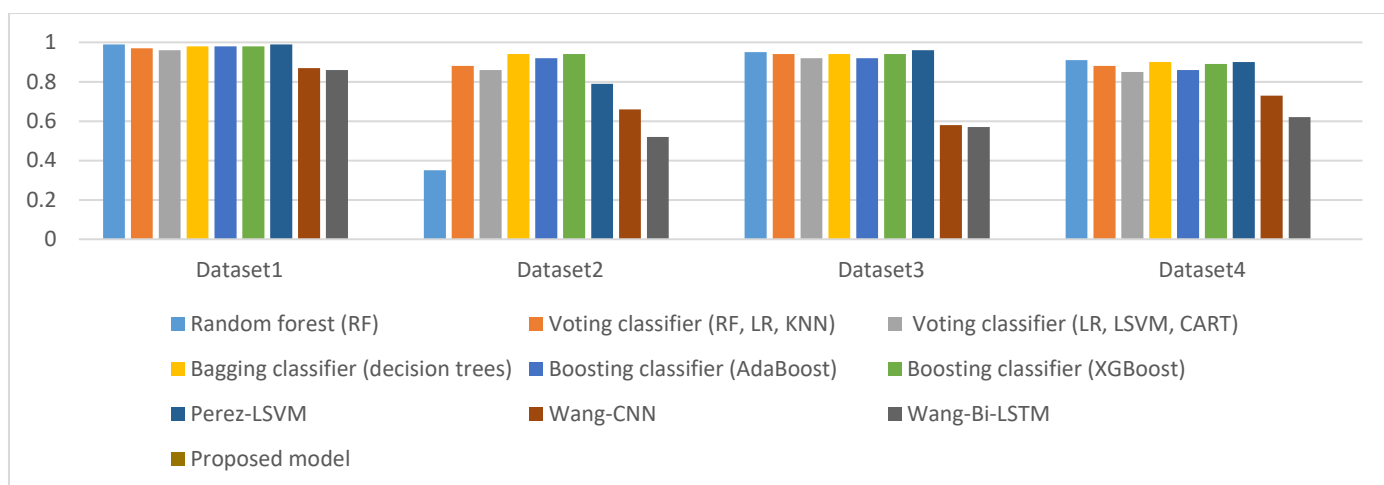


Figure. 6. Accuracy of four datasets with different models

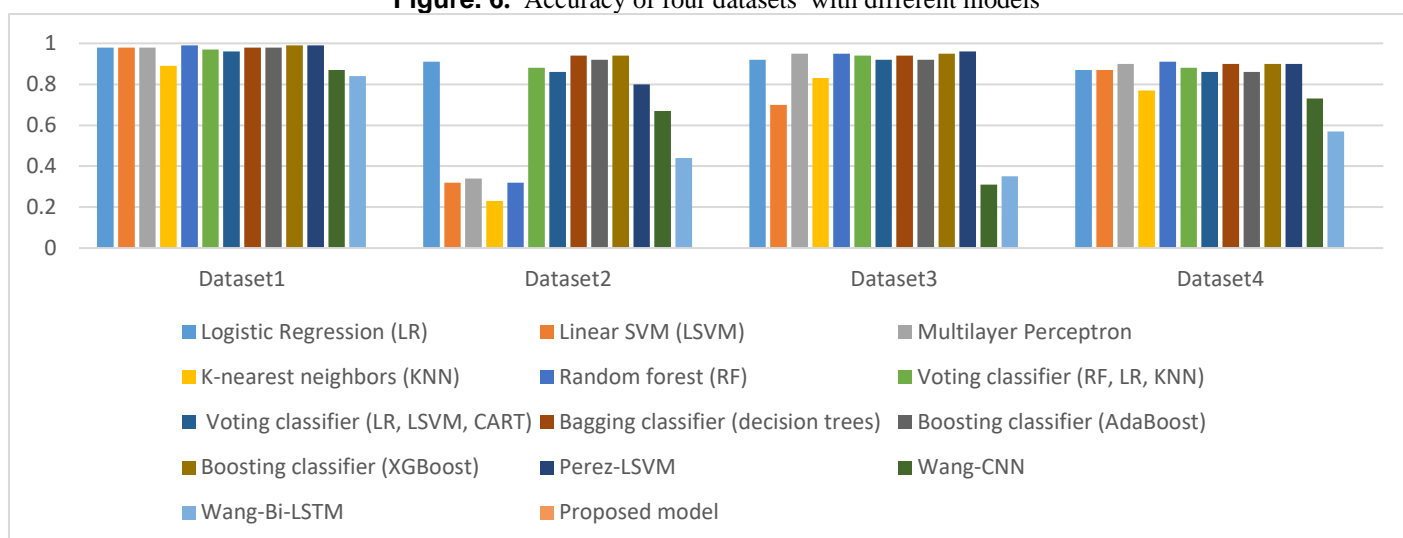


Figure. 7. F1-Score of four datasets with different models

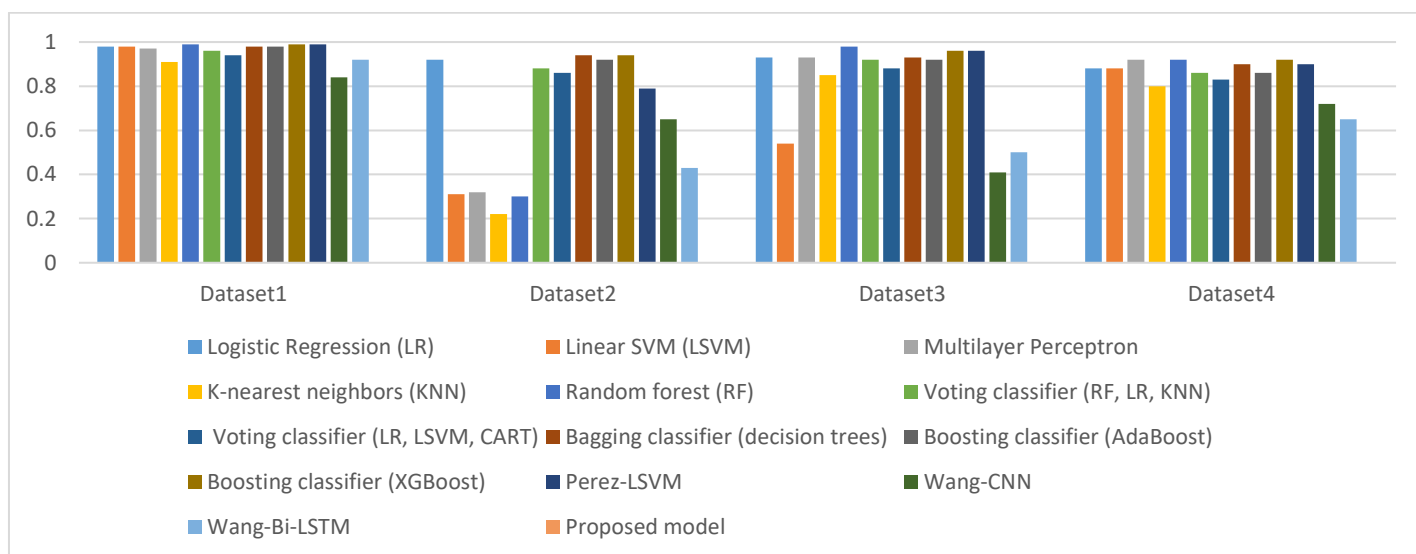


Figure. 8. Precision of four datasets with different models

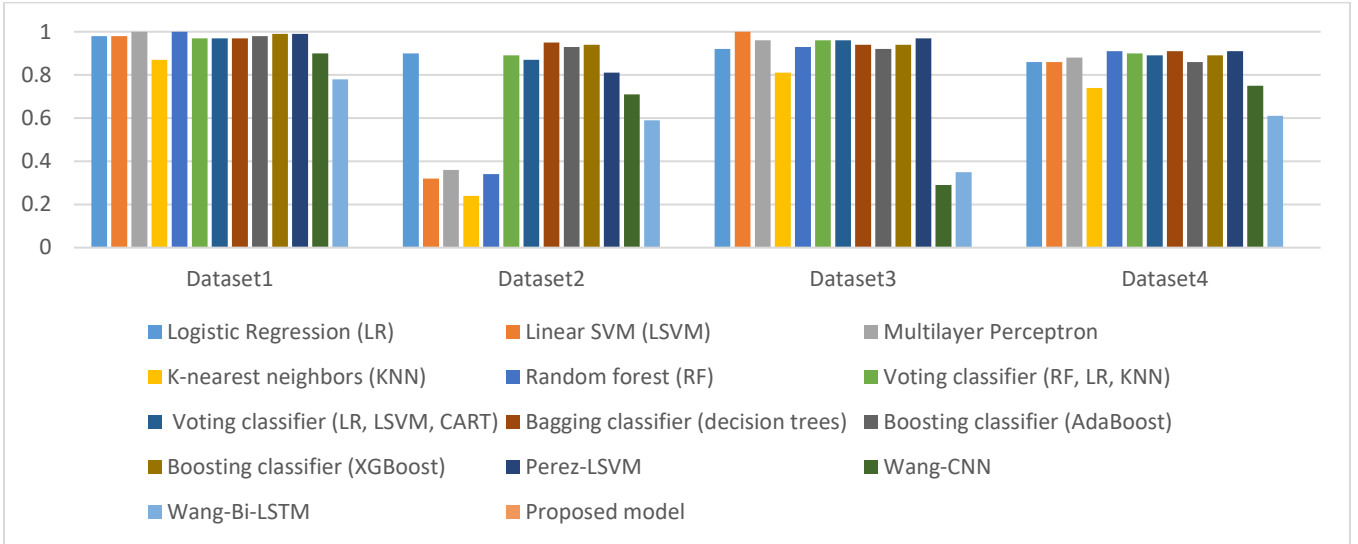


Figure 9. Recall of four datasets with different models

Similarly, the F1 score comparison show similar results as the accuracy metric. Precision metric shows boosting classifier with XGBoost competed with our model, but our overall model performance is better in all cases. For recall metric, our model perform slightly lower for individual datasets, but the combined dataset show better recall value than existing models.

Fig 6 depicts the accuracy of all algorithms for all four datasets. Our proposed model performed best with an accuracy of 97%, and the worst performed model was Wang-Bi-LSTM with an accuracy of 64.25%. Fig 7,8,9 shows different models' F1-score, precision, and recall values. The proposed model showed less recall value when compared to LSVM which showed 1.0 for the third dataset. Overall three metric values for the proposed model are best for all other cases.

V. CONCLUSION AND FUTURE WORK

Transformers showed significant improvement over the past decade in natural language processing applications. Different present deep learning models for false news detection are trained with different datasets. But the lack of generalization in models might lead to poor performance when deployed in real-world applications. Training a unified model instead of a specific model is necessary to work better in real-world applications. This manuscript showed that finetuning of the pre-trained BERT transformer model helped improve the performance of existing machine learning and deep learning models like CNN and LSTM. We combined three datasets to train the model and achieved a promising accuracy of 97% with a 0.97 F1-score. We argue that a combined dataset might give you better generalization as it includes samples from different domains like political, sports, entertainment, etc. This work provides a benchmark to train the models individually and uses the information obtained from the individual models to train the model with a larger dataset by combining all the samples. Furthermore, this model can be improved by reducing the number of layers without comprising the performance of the trained model. Our model can be adopted in applications like fake review

detection, spam classification, sentiment analysis detection for multiclass classification.

REFERENCES

- [1] "Fake News and Alternative Facts: Finding Accurate News: Why is Fake News Harmful?" [Online]. Available: <https://researchguides.austincc.edu/c.php?g=612891&p=4258046>
- [2] M. Balmas, "When fake news becomes real: Combined exposure to multiple news sources and political attitudes of inefficacy, alienation, and cynicism," *Commun. Res.*, vol. 41(3):430-454, 2014.
- [3] C. Kang and A. Goldman, "In washington pizzeria attack, fake news brought real guns," *New York Times*, 2016.
- [4] A. D. Holan, "2016 Lie of the Year: Fake News, Politifact, Washington, DC, USA, 2016."
- [5] "Fighting misinformation in the time of COVID-19, one click at a time." [Online]. Available: <https://www.who.int/news-room/feature-stories/detail/fighting-misinformation-in-the-time-of-covid-19-one-click-at-a-time>.
- [6] Gottfried Jeffrey and Shearer Elisa, "News use across social media platforms 2016. In Pew Research Center Reports." [Online]. Available: <http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/>.
- [7] N. J. Conroy, V. L. Rubin, and Y. Chen, "Automatic deception detection: Methods for finding fake news," *Proc. Assoc. Inf. Sci. Technol.*, 2015.
- [8] F. T. Asr and M. Taboada, "Misinfotext: a collection of news articles, with false and true labels," 2019.
- [9] "HuggingFace Transformers Library." [Online]. Available: <https://huggingface.co/transformers/quicktour.html>.
- [10] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science (80-.)*, 2018.
- [11] H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of Economic Perspectives*. 2017.
- [12] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake News Detection on Social Media," *ACM SIGKDD Explor. Newsl.*, 2017.

- [13] W. Y. Wang, “‘Liar, liar pants on fire’: A new benchmark dataset for fake news detection,” in *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 2017.
- [14] S. R. Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, “A simple but tough-to-beat baseline for the Fake News Challenge stance detection task,” 2017.
- [15] B. Ghanem, P. Rosso, and F. Rangel, “Stance Detection in Fake News A Combined Feature Representation,” 2019.
- [16] N. Ruchansky, S. Seo, and Y. Liu, “CSI: A hybrid deep model for fake news detection,” in *International Conference on Information and Knowledge Management, Proceedings*, 2017.
- [17] H. Jwa, D. Oh, K. Park, J. M. Kang, and H. Lim, “exBAKE: Automatic fake news detection model based on Bidirectional Encoder Representations from Transformers (BERT),” *Appl. Sci.*, 2019.
- [18] “Fake News.” [Online]. Available: <https://www.kaggle.com/c/fake-news>.
- [19] M. Szczepański, M. Pawlicki, R. Kozik, and M. Choraś, “New explainability method for BERT-based model in fake news detection,” *Sci. Rep.*, vol. 11, no. 1, p. 23705, 2021.
- [20] R. K. Kaliyar, A. Goswami, and P. Narang, “FakeBERT: Fake news detection in social media with a BERT-based deep learning approach,” *Multimed. Tools Appl.*, 2021.
- [21] I. Ahmad, M. Yousaf, S. Yousaf, and M. O. Ahmad, “Fake News Detection Using Machine Learning Ensemble Methods,” *Complexity*, 2020.
- [22] S. Tida, Vijay Srinivas; Hsu, “Universal Spam Detection using Transfer Learning of BERT Model,” in *Hawaii International Conference on System Sciences*, 2022.
- [23] J. F. Kolen and S. C. Kremer, “Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies,” in *A Field Guide to Dynamical Recurrent Networks*, 2010.
- [24] A. Vaswani *et al.*, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, 2017.
- [25] W. Liu *et al.*, “K-BERT: Enabling language representation with knowledge graph,” *arXiv*. 2019.
- [26] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized autoregressive pretraining for language understanding,” *arXiv*. 2019.
- [27] P. He, X. Liu, J. Gao, and W. Chen, “DeBERTa: Decoding-enhanced BERT with Disentangled Attention,” *arXiv*. 2020.
- [28] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, “Transformer-XL: Attentive language models beyond a fixed-length context,” in *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2020.
- [29] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019.
- [30] Radford Alec, Wu Jeffrey, Child Rewon, Luan David, Amodei Dario, and Sutskever Ilya, “Language Models are Unsupervised Multitask Learners | Enhanced Reader,” *OpenAI Blog*, 2019.
- [31] “Dataset Card for BookCorpus.” [Online]. Available: <https://huggingface.co/datasets/bookcorpus>.
- [32] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A survey on deep transfer learning,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.
- [33] J. Brownlee, “What is the Difference Between a Parameter and a Hyperparameter?,” 2017. [Online]. Available: <https://machinelearningmastery.com/difference-between-a-parameter-and-a-hyperparameter/>.
- [34] A. F. M. Agarap, “Deep Learning using Rectified Linear Units (ReLU),” *arXiv*. 2018.
- [35] “ISOT Fake News Dataset.” [Online]. Available: https://www.impactcybertrust.org/dataset_view?idDataset=952.
- [36] “Fake News Detection.” [Online]. Available: <https://www.kaggle.com/jruvika/fake-news-detection>.
- [37] X. Glorot and Y. Bengio, “Xavier Initialization,” *J. Mach. Learn. Res.*, 2010.
- [38] O. Konur, “Adam Optimizer,” *Energy Education Science and Technology Part B: Social and Educational Studies*. 2013.
- [39] “Metrics and scoring: quantifying the quality of predictions.” [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html.
- [40] T. M. Mitchell, “The Discipline of Machine Learning,” *Mach. Learn.*, 2006.
- [41] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. 2000.
- [42] S. Akhtar *et al.*, “Improving mispronunciation detection of arabic words for non-native learners using deep convolutional neural network features,” *Electron.*, 2020.
- [43] B. Gregorutti, B. Michel, and P. Saint-Pierre, “Correlation and variable importance in random forests,” *Stat. Comput.*, 2017.
- [44] L. Lam and C. Y. Suen, “Application of majority voting to pattern recognition: An analysis of its behavior and performance,” *IEEE Trans. Syst. Man, Cybern. Part A Systems Humans.*, 1997.
- [45] P. Bühlmann, “Bagging, boosting and ensemble methods,” in *Handbook of Computational Statistics: Concepts and Methods: Second Edition*, 2012.
- [46] R. E. Schapire, “A brief introduction to boosting,” in *IJCAI International Joint Conference on Artificial Intelligence*, 1999.
- [47] A. K. Agarwal, S. Wadhwa, and S. Chandra, “XGBoost: A Scalable Tree Boosting System Tianqi,” *J. Assoc. Physicians India*, 2010.
- [48] V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, “Automatic detection of fake news,” in *COLING 2018 - 27th International Conference on Computational Linguistics, Proceedings*, 2018.

Appendix: A

Table 2. Accuracy/F1-score/Precision/Recall for the different datasets

| Model | Dataset1 | Dataset2 | Dataset3 | Dataset4 |
|--|--------------------------------|--------------------------------|---------------------------------|--------------------------------|
| Logistic Regression (LR) [21],[40] | 0.97/0.98/0.98/0.98 | 0.91/0.91/0.92/0.90 | 0.91/0.92/0.93/0.92 | 0.87/0.87/0.88/0.86 |
| Linear SVM (LSVM) [21], [41] | 0.98/0.98/0.98/0.98 | 0.37/0.32/0.31/0.32 | 0.53/0.7/0.54/1 | 0.86/0.87/0.88/0.86 |
| Multilayer Perceptron [21], [42] | 0.98/0.98/0.97/1 | 0.35/0.34/0.32/0.36 | 0.94/0.95/0.93/0.96 | 0.90/0.90/0.92/0.88 |
| K-nearest neighbors (KNN) [21], [42] | 0.88/0.89/0.91/0.87 | 0.28/0.23/0.22/0.24 | 0.82/0.83/0.85/0.81 | 0.77/0.77/0.80/0.74 |
| Ensemble learners | | | | |
| Random forest (RF) [21], [43] | 0.99/0.99/0.99/1 | 0.35/0.32/0.30/0.34 | 0.95/0.95/0.98/0.93 | 0.91/0.91/0.92/0.91 |
| Voting classifier (RF, LR, KNN) [21], [44] | 0.97/0.97/0.96/0.97 | 0.88/0.88/0.88/0.89 | 0.94/0.94/0.92/0.96 | 0.88/0.88/0.86/0.90 |
| Voting classifier (LR, LSVM, CART) [21] | 0.96/0.96/0.94/0.97 | 0.86/0.86/0.86/0.87 | 0.92/0.92/0.88/0.96 | 0.85/0.86/0.83/0.89 |
| Bagging classifier (decision trees) [21], [45] | 0.98/0.98/0.98/0.97 | 0.94/0.94/0.92/0.95 | 0.94/0.94/0.93/0.94 | 0.90/0.90/0.90/0.91 |
| Boosting classifier (AdaBoost) [21], [46] | 0.98/0.98/0.98/0.98 | 0.92/0.92/0.92/0.93 | 0.92/0.92/0.92/0.92 | 0.86/0.86/0.86/0.86 |
| Boosting classifier (XGBoost) [21], [47] | 0.98/0.99/0.99/0.99 | 0.94/0.94/0.94/0.94 | 0.94/0.95/0.96/0.94 | 0.89/0.90/0.92/0.89 |
| Benchmark Algorithms | | | | |
| Perez-LSVM [21], [48] | 0.99/0.99/0.99/0.99 | 0.79/0.80/0.79/0.81 | 0.96/0.96/0.96/0.97 | 0.90/0.90/0.90/0.91 |
| Wang-CNN [21], [13] | 0.87/0.87/0.84/0.90 | 0.67/0.67/0.65/0.29 | 0.58/0.31/0.48/0.29 | 0.73/0.73/0.72/0.75 |
| Wang-Bi-LSTM [21], [13] | 0.86/0.84/0.92/0.78 | 0.52/0.44/0.43/0.59 | 0.57/0.35/0.50/0.35 | 0.62/0.57/0.65/0.61 |
| Proposed model (minibatch size) | 0.99/0.99/0.99/0.99(32) | 0.97/0.97/0.96/0.99(64) | 0.99/0.99/0.99/0.98(128) | 0.97/0.97/0.98/0.95(64) |