Claim Check Worthiness using FineTuning of BERTweet

Elapanti Sri Sai Chaithanya
Amuru Hareesh

Munagala Krishna Sai
Bhavana Nellore

1 Problem Statement

Given a tweet, predict whether it is worth fact-checking by professional fact-checkers or not.

2 Introduction

With the fast rise of social media platforms like Twitter, Facebook, etc., many false and unconfirmed claims/news have surfaced and spread, affecting both online social media users and the offline population.

3 Related Works

In this section, we will discuss previous work on check-worthiness estimation for tweets. Check-worthiness can be defined as checking/validating the claim/news floated on various social media platforms. So to reduce the spreading of fake news, we can find the worthiness of the claim.

Recently, many methods have been proposed to check a claim's worthiness. Check that! The lab has done various tasks on Check-worthiness(Shaar et al.). where they have crawled and annotated the tweets manually. They have only considered a few hashtags such as #covid19, #Corona,#CoronavirusOutbreak, #Coronavirus, #CoronaAlert, #CoronaOutbreak, Corona, and covid-19 and also included tweet popularity as a measure to build the dataset from the tweeter. Twelve teams have performed various models on this dataset and found that Transformers or a combination of embeddings have given the best results. Top performing team Accenture has used a model based on RoBERTa, with an additional mean pooling and dropout layer on top of the primary RoBERTa network. To avoid overfitting, the mean pooling layer averages the outputs from the last two RoBERTa layers, after which the result is passed to a classification head and a dropout layer. As the official evaluation metric, mean average precision (MAP) has been considered.

CheckThat! The lab(Hasanain et al.) has also conducted experiments with languages like Arabic and made the dataset with 15 different topics. Then annotated the tweets manually with a set of keywords, hashtags, and usernames to build the Arabic dataset, limited the dataset to the original Arabic tweets and crawled these tweets into the dataset. Afterward, they ranked the tweet's popularity (defined by the sum of their retweets and likes) using parameters like likes and retweets. They took the top 500 tweets, 27.5 of the entire dataset. Out of 15 topics, three are used for training the models and the rest for evaluating the models. They have achieved the best result from finetuning the pre-trained models like AraBERT and multilingual BERT. They have also used other pre-trained models like Glove, Word2vec, and Language-Agnostic sentence Representations (LASER) to obtain embeddings for the tweets and applied pos tagging, etc., to it was fed into a neural network or other machine learning models, such as SVM. They have evaluated the models using precision at k and Mean Average Precision and have considered P@30 as the official evaluation measure.

In 2021 the CheckThat! The lab(Nakov et al., 2021) has performed tasks on Arabic, Bulgarian, English, and Spanish datasets to check claim worthiness. Task 1 predicts fact-checking the worthiness of tweets in a Twitter stream(focusing on the COVID-19 data set). The datasets have been built based on the different languages chosen. They also added a few more tasks in multi-class fake news detection for news articles and domain classification focusing on COVID-19 in other languages. The authors experimented by adding 200 more labels to 2020's dataset to test the English dataset and various labels to different languages to make it multi-lingual. For evaluation, mean average precision or precision at rank k is used for ranking tasks, and F1 is used for the classification tasks.

4 Methodology

Data preprocessing, word embedding, models, and model fine-tuning are the four steps that determine whether a claim is worth investigating.

4.1 Dataset

The Dataset used is all about COVID-19. Train data has three columns, tweet, id, and label, with 2122 entries, whereas test data has two columns, id, and tweet, with 195 entries. Dataset is a combination of truthful news/claims and unreliable claims/news.

Figure 1: Word Cloud of Trained Data

The above word cloud describes the most frequent words in the dataset. The big words are the most repeated in the covid-19 dataset; here, the size represents the frequency of the words.

4.2 Data Preprocessing

In this section, we have used various NLP techniques for preprocessing the given input data to be used to extract the features. Below are the different preprocessing steps which we used on the given dataset.

1. Eliminated “@” as we have tweeter data as @ indicates the username
2. Removal of newline characters at the end of every sentence
3. Removal of distracting single quotes
4. Removal of punctuation and numbers
5. Removal of single characters
6. Removal of accented words
7. Removal of multiple spaces
8. After observing the data cloud, we saw that some unnecessary words were frequently repeated, and we removed those words, also.

The figure 2 shows the count of tweets based on the word length after preprocessing steps.

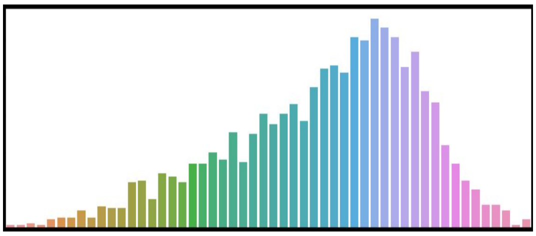

Figure 2: Count of Tweets with respect to Word Length

4.3 Word Embeddings

Word Embedding is very important because we must transform the dataset into a format that models can understand. We can’t use the same word embeddings for every model, so we employed different word embeddings for the models like TF-IDF, Word2vec, embedding layer, and Bert embeddings(pre-trained). Word Embeddings represent the words in the form of real-valued vectors.

TF-IDF(Term frequency-inverse document frequency) is the word embedding used for text which considers the two metrics called inverse document frequency and term frequency. This helps extract keywords and stop word removal and mainly focuses on rarely used words.

For some models, we used an embedding layer, which acts as the hidden layer in the network. This considers integer-encoded vocabulary and checks the embedding vector for every word index.

In 2013 Google developed Word2Vec, which is based on the distributional hypothesis. Word2Vec considers the order of the words in the past and future. This helps reconstruct the linguistic meaning of words. Word2Vec maps semantically similar words to geometrically close vectors. The metric used in this embedding is cosine similarity between two words or documents.Word2Vec is of two types Skip-gram and Continuous Bag of Words (CBOW). For our data, we used CBOW word2Vec.

For Transformers, we use BERT(Bidirectional encoder representations from transformers), which mainly relies on the attention mechanism. This is a pre-trained method that, when passed through each BERT layer, captures the word associations based on their right and left words.

4.4 Models

Following a review of related publications on check claim worthiness, we discovered that numerous models could be used for this problem. The models we used are Non-Contextual Embeddings with SVM,CNN,LSTM and BiLSTM,SBERt with

SVM,FineTuning BERT and our proposed model.

4.4.1 Non-Contextual Embeddings with SVM

From the non-contextual Embeddings, we started with the TF-IDF embeddings, and we ran the model with the classical machine learning model, support vector machine. For SVM, we have taken the kernel as RBF. Then, we ran the same support vector machine for the word2vec embeddings. Since the context is not considered in this case, the results are also unsatisfactory. The F1 score on the test dataset is around 0.45.

4.4.2 CNN

For CNN, we have taken tokens from the TensorFlow tokenizer and converted them using texts_to_sequences() with a max_sequence length of 55. We have added the embeddings layer for the CNN model with 55 as input and the output dimensions as 128. We have added 32 * 5 convolutions twice with relu as an activation function. We have added max-pooling to that. Then, we added Bi-Directional LSTM with size 100 and a dropout of 0.3. Finally, the dense layer of size one is added as output with sigmoid activation and adam optimizer.

4.4.3 LSTM and BiLSTM

For LSTM and BiLSTM, we have taken tokens from the TensorFlow tokenizer and converted them using texts_to_sequences() with a max_sequence length of 55. We have added the embeddings layer for the LSTM model with 55 as input and the output dimensions as 128. Then, we added Bi-Directional LSTM or LSTM with size 100 and a dropout of 0.3. Finally, the dense layer of size one is added as output with sigmoid activation and adam optimizer.

The results for the CNN and RNN models also need to be more satisfactory for the test data. So, we have tried the contextual embeddings from the pre-trained Transformer. (Hochreiter and Schmidhuber, 1997)

4.4.4 SBERT+SVM

We have taken the embeddings from the SBERT of the version “sentence-transformers/stsb-mpnet-base-v2" and the output dimensions as 768. We have added a support vector machine on top of this. The results are better than the previous non-contextual embeddings.

4.4.5 FineTuning BERT

Then, for the Finetuning of BERT, we have taken the BERT embeddings of the base version. The dimensions of the BERT embeddings are 1024. Then we added the linear layer of dimensions 1023 to 512, with relu as activation and 0.1 as a dropout. Then we added a similar structure of linear layers from 512 to 128 and then to 128 to 2. Finally, we added the softmax to the last layer. The training is done with Adam optimizer, batch size of 16, and a learning rate of 5e-5. (Devlin et al., 2019)

4.4.6 FineTuning BERTweet

BERTweet has a similar Architecture to the BERT Base. This BERTweet is trained on 850 M tweets in English, with the pre-trained procedure being the same as RoBERTa. In the other versions, the BERTweet is pre-trained on 25M tweets based on covid. For the present proposed model, we have tried all the versions but finally, the version with "vinai/bertweetcovid19-base-uncased" seems relevant because the dataset is similar. (Nguyen et al., 2020)

For the proposed model, we have taken the BERTweet embeddings of the covid version. The dimensions of the BERTweet embeddings are 784. We freeze the last layer to this. Then we added the linear layer of dimensions 784 to 512, with relu as activation and 0.1 as a dropout. Then we added a similar structure of linear layers from 512 to 128 and then to 128 to 2. Finally, we added the softmax to the last layer. The training is done with Adam optimizer, batch size of 16, and a learning rate of 5e-5.

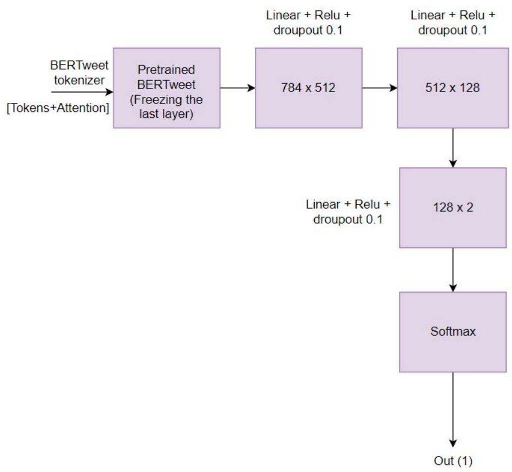

Figure 3: Proposed Model Architecture(FineTuning BERTweet)

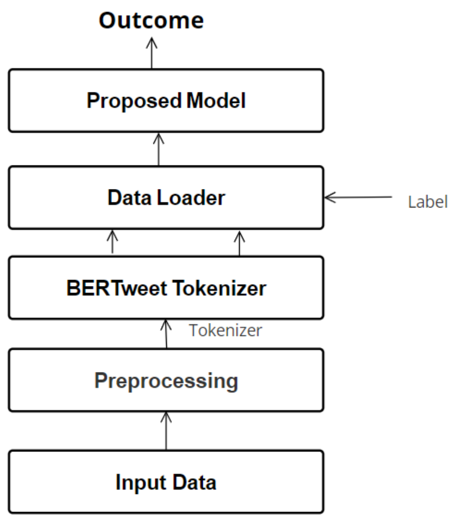

Figure 4: Data Flow Diagram

5 Experiment Results

For the tokenization of the data, we have taken the max sequence length as 55. From Fig2, it is clear that some of the tokens have crossed the length 55. We have used post-truncation and post-padding. Although we experimented with different max sequence lengths, 55 gives better results. We have tried with a different learning rate from 1e-5 to 5e-5 using the Adam optimizer, but with 5e-5, the proposed model gives better results. We have used the cross entropy loss for all the experiments.For the dataset, we have tried a batch size of 16. We experimented with different dropouts to the layers and finally fixed them to 0.1.

From the below results, we can observe that the best model is fine-tuning with BERTweet.

Models	F1 - Score
BERTweet	0.84761
Bert	0.78
SBERT+SVM	0.74
LSTM and BiLSTM	0.44
CNN	0.48
TF-IDF + SVM	0.44186
Word2Vec+SVM	0.44827

6 Analysis

The claim check worthiness of the tweet is predicted based on FineTuning BERTweet (Proposed model) giving better results, i.e., F1-Score. This pre-trained model is trained on similar data (tweets

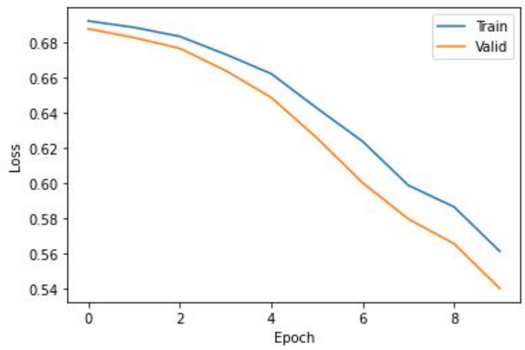

Figure 5: Epochs Vs Loss(BERTweet)

of covid). So the context is considered through embeddings. We can further improve the score by tuning the hyperparameters.

7 Contributions

Krishna and Sai Chaitanya worked on TF-Idf with SVM, Word2Vec with SVM, and LSTM. Hareesh and Bhavana worked on LSTM, BiLSTM, and CNN. We all contributed to SBERT with SVM, fine tunning BERT, and BERTweet.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Number: arXiv:1810.04805 arXiv:1810.04805 [cs].

Maram Hasanain, Fatima Haouari, Reem Suwaileh, Zien Sheikh Ali, Bayan Hamdan, Tamer Elsayed, and Alberto Barron-Cedeno. Overview of Check-That! 2020 Arabic: Automatic Identification and Verification of Claims in Social Media. page 13.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Preslav Nakov, Giovanni Da San Martino, Tamer Elsayed, Alberto Barrón-Cedeño, Rubén Míguez, Shaden Shaar, Firoj Alam, Fatima Haouari, Maram Hasanain, Nikolay Babulkov, Alex Nikolov, Gautam Kishore Shahi, Julia Maria Struß, and Thomas Mandl. 2021. The CLEF-2021 CheckThat! Lab on Detecting Check-Worthy Claims, Previously Fact-Checked Claims, and Fake News. In Djoerd Hiemstra, Marie-Francine Moens, Josiane Mothe, Raffaele Perego, Martin Potthast, and Fabrizio Sebastiani, editors, *Advances in Information Retrieval*, volume 12657, pages 639–649. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for

English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, Online. Association for Computational Linguistics.

Shaden Shaar, Alex Nikolov, Nikolay Babulkov, Firoj Alam, Alberto Barron-Cedeno, Tamer Elsayed, Maram Hasanain, Reem Suwaileh, and Fatima Haouari. Overview of CheckThat! 2020 English: Automatic Identification and Verification of Claims in Social Media. page 24.