Fine Grained Fake News Classification using Liar Dataset

Anonymous ACL submission

1 Problem definition

Task description: Given a statement (tweet, political debate, ad etc), predict the truthfulness rating of the statement – pants-fire, false, barely true, half-true, mostly-true, and true. This is a multi-label classification task.

The threat of false news propagating through media outlets to the veracity of information is significant, and efforts to identify fake news have gained more attention in recent years. Because false news is frequently prepared with the goal of misleading readers, it is extremely difficult to identify fake news only based on news content. Fake news, on the other hand, can offer varying degrees of fakeness and even use real proof to imitate real news, making it harder to spot. On the other hand, the dissemination of false information results in a variety of data from numerous angles.

The major task that was given to us was to classify text like tweet, political debate, ad etc. based on our model analysis under these 6 classifiers pants-fire, false, barelytrue, half-true, mostly-true, and true .

2 Related works

In the paper 1 we used LIAR: dataset for fake news detection consists of 12,836 short statements taken from POLITIFACT and labeled by humans for truthfulness, subject, context/venue, speaker, state, party, and prior history. For truthfulness, the LIAR dataset has six labels: pants-fire, false, mostly-false, half-true, mostly-true, and true. These six label sets are relatively balanced in size.

This is a 6-way multiclass text classification problem.How effective are machine learning algorithms at putting a brief remark into a fine-grained category of fakeness based simply on surface-level language realizations?.Can we create an architecture for a deep neural network that incorporates speaker-related meta-data with text to improve the effectiveness of fake news detection?

CNN has given good results on text data so they used hybrid convolutional neural networks for integrating text and meta-data.To encode the metadata embeddings, we initialize a matrix of embedding vectors at random. To capture the relationship between the meta-data vector, we employ a convolutional layer (s). Then,On the latent space, a bi-directional LSTM layer is applied after the usual max-pooling process. In order to create the final prediction, we concatenate the maximum-pooled text representations with the bidirectional LSTM's meta-data representation and feed them to a fully connected layer with a softmax activation function.

We can see that, for the validation and test sets, respectively, the majority baseline on this dataset provides approximately 0.204 and 0.208 accuracy. Standard SVMs and LR models, two types of text classifier, significantly improved. The performance of the Bi-LSTMs was poor because of overfitting. The accuracy on the heldout test set was 0.270 thanks to the superior performance of the CNNs over all other models.

This paper 2 presented the details of the proposed framework for fake news detection named UPFD (User Preference-aware Fake News Detection).The framework is made up of three main parts. First, given a news item, we search through previous user posts to determine user endogenous preference. By encoding old posts with text representation learning methods, we implicitly derive the preferences of active users. The textual news data is encoded in the same way. Second, we construct the news propagation

graph in accordance with its engagement data on social media sites in order to take advantage of user exogenous context.To combine the user's endogenous desire and exogenous context, we thirdly design a hierarchical information fusion process. To be more specific, we employ a graph encoder called a GNN to obtain the user engagement embedding, using the text encoder's news and user embeddings as the relevant nodes.The news dissemination graph features The final news embeddings are made up of the concatenation of the news textual embedding and user engagement embedding.

Real and fake news detection is the main topic of this paper 3. The goal of the study is to use the transformer model to determine if news is authentic or fraudulent. The analysis of the Literature demonstrates that numerous ML-based detection techniques are used to identify fake news. However, these models don't perform well. Utilizing a transformer model with an attention mechanism, we attempt to enhance performance.

As a multiclass text classification framework, we use a dataset. Due to the attention mechanism architecture's best performance in many NLP-based tasks and its accuracy in NLP tasks like text summarization and language translation. There hasn't been any additional study on using the attention mechanism for NLP tasks, though. Consequently, in order to enhance the effectiveness of false news detection we used an attention based architecture called the transformer model.

The transformer model uses attention to boost speed, specifically self attention and uses best encoder decoder architecture.Each encoder layer further consists of two layers: self-attention layer and a feed-forward neural network (FFN) layer. The decoder also has multiple layers. Besides, self-attention and FFN layers, decoder contains the Encoder-Decoder attention layer. This Encoder-Decoder attention layer helps to concentrate on relevant components of the text.

The main contribution of this paper 4 is to provide a comprehensive review of automatic fake news detection and a summary of the current industry developments in the field of deep learning, encouraging subsequent scholars to make suggestions and improve the performance of contemporary technologies in the industry. It uses The Weibo dataset is a Chinese Named Entity Recognition dataset taken from Sina Weibo, a Chinese social media platform. The dataset includes user preference, region, organization, and user posting. UPFA dataset can evaluate binary graph classification, graph anomaly detection, and fake news detection tasks. To fill the research gap of the lack of DL models for fake news detection,, the author proposes a novel hybrid deep learning model that combines both RNN and CNN techniques that improve the accuracy of the classification task. Precisely, the model does binary classification between false information and legitimate news, and the result denotes either false or real. The hybrid CNN-RNN model utilizes CNN to capture local characteristics and make use of RNN to learn the long-term dependencies(sequential characteristics) of the dataset

3 Methodology

We preprocessed the data we needed, both training as well as testing, in the following manner: Lowercasing the text and any other metadata if we used and stemming using nltk library routine, PorterStemmer.

We used eight baselines to solve the multi-class text classification problem. We used linear regression classifier (LR), a supervised support vector machine classifier (SVM), a decision Tree classifier (DT), a k-nearest neighbor classifier (KNN), a bi-directional long short-term memory networks model (Bi-LSTMs), an artificial neural network, (ANN), and a distilBERT model. For LR, SVM, DT, RF, and KNN, we used the sklearn library. For ANN, and Bi-LSTMs, we used TensorFlow for the implementation. For distilBERT, we used PyTorch for the implementation.
We compared the results of the machine learning models, using two different settings. In the first setting, we used the TF-IDF vectorizer to vectorize the corpus. We used only the statements columns in it. All the statements were converted to a vector of dimension 500. In the second setting, we built a more robust model. We used the Sentence Transformer, to convert the corpus into vectors. We used all-mpnet-base-v2 to vectorize as it gave the

better results as compared to all-miniLM-L6-v2. This model maps sentences and paragraphs to a 768-dimensional dense vector space. In this setting, we used four attributes: statement, subject, speaker, and the party affiliation.

We strictly tuned all the hyperparameters on the validation dataset. We used the rbf kernel in SVM, 50 nearest neighbor in KNN, and 200 n-estimators in RF. In both ANN and Bi-LSTM, 4 attributes were used as described above. In ANN, we used 3 dense layers with 512, 64 and 6 nodes. We used the ReLU activation function for the first 2 layers and the last layer used a softmax activation function. 2 dropout layers were added in between with 0.2 probabilities. Adams optimizer was used with sparse categorical cross entropy, and it was trained for 20 epochs. In Bi-LSTMs we used an embedding layer, a bi-LSTM layer and a dense layer with 6 nodes. Softmax activation function was used in the dense layer. The batch size for Adam optimization was set to 64, and the learning process involves 10 passes over the training data for text model. The loss used was categorical crossentropy.

For SVM, we first used the contextual embedding model "all-MiniLM-L6-v2". We used the first column only from the dataset and got an f1-score of 0.23.When we used one more column namely "subject" the score increased to around 0.25. We decided to introduce a new contextual embedding model "all-mpnet-base-v2" and it performed better than the previous one with one column as well as two columns. In the final attempt, we took four columns namely, "statement", "speaker", "subject", "party-affiliation" and didn't use the rest of the columns because it had null value in the entries. The final highest score we get using the latter contextual embedding model is 0.30 which is higher than 0.27 for when we used only two columns.

For distilBERT model, we used the DistilBertTokenizer for tokenization and DistilBertModel from transformer module. We used the loss function BCELogitswithLoss from pytorch and Adam optimizer. The Linear layer is 6. We fine tuned the model with 3 epochs and we got a loss of around 0.5 from that. We chose F1-macro score as the evaluation metric.

Model	F1-Score
KNN (TFIDF + 1 col)	0.21
KNN (SentenceTransformer+4 cols)	0.26
DT (TFIDF + 1 col)	0.20
DT (SentenceTransformer+4 cols)	0.22
RF (TFIDF + 1 col)	0.20
RF (SentenceTransformer+4 cols)	0.26
SVM (TFIDF + 1 col)	0.21
SVM (SentenceTransformer+4 cols)	**0.30**
LR (TFIDF + 1 col)	0.16
LR (SentenceTransformer+4 cols)	0.23
ANN (4 cols)	0.28
Bi-LSTM (4 cols)	0.23
distilBERT	0.25
SVM(SentenceTransformer+3col)	0.254

Figure 1: Experimental results of the models we used

4 Experimental Results

The figure 1 shows the f1 score as submitted on kaggle as well. The first model we tried is K-nearest neighbor with Tfidfvectorizer and only the statement column as our dataset and the f1-score we got is 0.21 While for the same model with sentence transformers and 4 columns including the statement column the f1-score increases to 0.26. The three additional metadata columns are "subject", "speaker" and "party affiliation". We also tried the decision tree model where when using TfidfVectorizer we got a score of 0.20 while with contextual embedding and additional metadata we got an increased score of 0.22.

We can say that ml models with contextual embedding give an overall better score than with non-contextual embedding. The most significant difference we saw is in the SVM (support vector machine) model with a highest score of 0.30 with contextual embedding and only a score of 0.21 with non-contextual embedding. The linear regression model performed the lowest with only a score of 0.16 with non-contextual embeddings.

We also tried Bi-LSTM as well as distilBERT with only the "statement" column and got a score of 0.23 and 0.25 respectively. We got a score of 0.28 with ANN with the same 4 columns as above.

5 Analysis

We observed that the F1 macro score for the machine learning models were less when we used

only the attribute – ‘statements. However, when we added the meta-data: ‘subject’, ‘speaker’, and the ‘party affiliation’, the F1 score increased. Thus, the model performs better on addition of more information as expected.
Secondly, we observed that the F1 macro scores were less when we used the TF-IDF vectorizer for vectorizing the data. However, when we used the sentence transformer to convert the corpus to the vector space, the F1 score increased. This is due to the fact that TF-IDF is a non-contextual vectorizer. Though it converts all the text to vectors, the semantic meaning is completely lost. On the other hand, the sentence transformer is a much robust model, which is a contextual model. Thus, the meaning of the statements remain embedded in the vector, and thus the model performs much better using this model.

We observed that random forest model performed better than the decision tree model. This is because random forest is an ensemble learning method, which uses several decision trees to predict the class. Thus, it’s more robust and accurate, and give better results. We observed that the rbf kernel in SVM performed better than the linear kernel. This is because the rbf kernel was able to model the non-linear data well.

Our best model was SVM with a radial basis function kernel , which used 4 attributes as described above which was vectorized with sentence transformer with all-mpnet-base-v2 model. Our second best model was ANN, which also used the same 4 attributes, and was vectorized using the same model.

6 Contribution of each member

Saurabh - Bi-LSTM, ANN, RF, KNN
Aftab - DT, DistliBERT
Rayudu - Report, SVM, Sentence Transformers
Rahul - LR, Tfidf, Presentation, Report

References

[1] Wang, W. Y. (2017). " liar, liar pants on fire": A new benchmark dataset for fake news detection. arXiv preprint arXiv:1705.00648.

[2] Dou, Yingtong, et al. "User preference-aware fake news detection." Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021.

[3] Qazi, M., Khan, M. U., Ali, M. (2020, January). Detection of fake news using transformer model. In 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET) (pp. 1-6). IEEE.

[4] Feng, Y. (2022). Misreporting and Fake News Detection Techniques on the Social Media Platform. Highlights in Science, Engineering and Technology, 12, 142-152.