Fake News Challenge
Comparing Models on Stance Detection

KANNAV DHAWAN
University of Waterloo
Electrical and Computer Engineering

Abstract

The fake news challenge is an NLP task focused on resolving the problem of fake news using headline and body text. Stance detection task can be considered as the core task in detecting fake news. Considering the problem as a classification problem, Pretrained Bert classifier (Simple Transformers) was considered as the baseline. Various classical machine learning models viz. Gradient Boosting, Random Forest, Logistic Regression, XGBoost with hand features and additional features were employed. Neural Networks viz. CNN, LSTM and Bidirectional LSTM with varying feature engineering techniques using different pretrained Glove and CBOW embeddings trained using Word2Vec were employed at different truncation lengths for the sequences. Bidirectional LSTM's outperformed in the Neural Network pipeline, thus an architecture consisting of separate Embedding and Bidirectional LSTM layers for both headline and body was constituted, a robust feature engineering with baseline features, TFIDF were used in the layers which finally provided us the state-of-the art results with a score of 9460 and an accuracy of 81.19% on the competition set.

1 Introduction

To combat the problem of Fake News by its automatic detection, (Pomerleau and Rao, 2017) organized the Fake News Challenge (FNC-1, 2017) to explore the performance of AI technologies to perform the task of stance detection at the first stage in the fact checking pipeline. The focus of FNC-1 is to classify the stance of a News Article with respect to a given headline which may or may not be for the same article and vice-versa. The stance of the article can either 'Agree', 'Disagree', 'Discuss' or be 'Unrelated' to a given headline. The classical machine learning models, CNN's for better feature extraction, RNN's for sequential processing of the input with additional feature engineering techniques can be employed to solve the problem to an extent.

In this paper, a comparison between the classical machine learning models viz. Gradient Boosting, RandomForest, Logistic Regression, XGBOOST with feature engineering techniques and Neural Networks is made. The effect of Glove and trained CBOW embedding at embedding layer of the Neural Networks is analyzed. Feature extraction techniques like count vectorization, Tfidf Vectorization and cosine similarity were employed and compared. The effect of truncations of each input was considered to determine its effect on time and accuracy.

2 Related Work

Baseline model for stance detection used hand features having n-gram co-occurrence counts, polarity, word overlapping and refuting features to train a Gradient Boosting classifier which achieved a score of 8748.75(75.08%) at the competition set. Deep learning approaches are often used in the tasks of text classification. These techniques involve the implementation of embedding layer as the input layer where the positive integers from the word index dictionary are converted into dense vectors of fixed size. Such dense vectors are created using the pretrained word embeddings viz. Glove, Fasttext etc. or trained using CBOW, Skip-gram which were given by (Corrado et al, 2013) etc. These embeddings use the cosine similarity and other mechanisms to make the words having similar context appear closer to each other when generating vectors.(Riedel et al., 2018) proposed a method using BOW representations for the text input viz. TF and TFIDF. They made the feature vector using the TF vector for both the headline

and body and the *l*2 normalized cosine similarity between them. On feeding the inputs in an MLP model, a relative score of 81.72 was achieved. (Baird et al., 2017) performed stance detection by combining the models. An ensemble based on 50-50 weighted average of gradient- boosting decision trees and a deep CNN for feature extraction where 2 different CNN's were made for headline and body and combined together for feature extraction whereas the "count features", "SVD features" etc. were also fed into the XGBOOST. Much faster computations were recorded because of no Recurrent mechanism in the architecture giving a score of 9556.50 and a relative score of 82.02 on competition set. Long short-term memory, (Schmidhuber et al., 1997) networks are the special type of RNN's designed specifically to learn long-term dependencies. These networks can add/remove the information to the cell state which is regulated by gates. The cell state of LSTM runs through the entire chain and information flows through it. Further the bi-directional LSTM with forward and backward states was introduced to retain the contextual information from both the directions which was not a case with LSTM(Paliwal and Schuster., 1997).

(Pfohl et al., 2017), applied the attention mechanism and conditional encoding on LSTM networks using averaged bag of vectors constructed for headline and body vectors. On processing the headline and the body in two separate LSTM's, the final hidden state of LSTM(headline) is sent as an input to the hidden state of the LSTM Network processing body. AttentionMechanism works on the final hidden state of the LSTM network processing headline and on the hidden state of LSTM processing body. They reached a relative score of 80.8 on the competition set. When Attention mechanism coexists with encoder-decoder architecture of the Bi-directional LSTM, the most recent output of the input sequence is conserved enabling chosen model to starchily grasp the inputs and associating output with its sequences. They (Parmar et al. 2017) presented a transformer model which supersedes the recurrent layers in the LSTM encoder-decoder composition with the leverage of more swift training.

3 Approach

After the brief analysis of the methodologies discussed in the previous section, three pipelines were constituted consisting of different Neural Networks viz. CNN, LSTM, Bi-directional LSTM with pretrained Glove and trained Word2Vec embeddings with input sequence having different truncation lengths, classical machine learning models with baseline and similarity features and the Bert classifier(Thilina Rajapakse, 2020) build on top of the transformers implementation by HuggingFace as our baseline model. A more robust Bidirectional LSTM was implemented using all the features from different architectures with separate embedding layers and Bi-LSTM layers for headline and body. This model is a big improvement to the basic architecture (Proctor and Davis, 2018) with just separate embedding layer for headline and body.

3.1 Preprocessing

General preprocessing of removing insignificant information comprising of punctuations, stop words and Non-ASCII characters was carried out. In Neural Networks, Word index dictionary was formed using keras *fit_on_texts* followed by the text sequencing where the input text is converted into corresponding index from the word index dictionary and the sequences are truncated and padded given the fixed length.

3.2 Feature Engineering

The provided baseline consists of 4 features viz. Word overlap, Polarity, Refuting and hand features. Word overlap feature is the intersection count of head and body divided by the total number of words whereas polarity is the count of refuting words in the input body and headline. For each input tokenized headline where refuting word is a list of provided refuting words, new Refuting feature set comprises of:

$$\begin{cases} 1\ if, Token\ in\ Refuting\ word \\ 0\ if, Token\ !\ in\ Refuting\ word \end{cases}$$

This helps in improving the precision of 'unrelated' stances to a large extent. Feature consisting of separate token and n-gram count of headline appearing in body text for each input as a list was taken as hand features.

(Wang and Oates, 2015) used Gramian matrix to find the correlations and achieved SOTA classification results for temporal data. Taking inspiration from that, we used the gram matrix of TFIDF as one of our features for the classical models. This helps in getting a new feature

depicting the similarity between the headline and the body text. Figure 1 shows the approach.

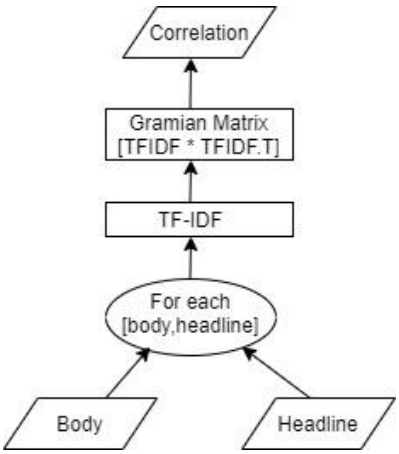

Figure 1: Gramian Features

3.3 Models

BERT being a bidirectional model based on the transformer architecture replaces the sequential nature of LSTM and GRU using optimized attention-based scheme. Bert's implementation of SimpleTransformer(Rajapakse,2020) build on top of HuggingFace was taken as the baseline .

3.3.1 Neural Networks

We setup a pipeline of CNN, LSTM and Bidirectional-LSTM with different word embeddings to analyze the behavior of CNN and RNN for stance detection. Embeddings being the relation-specific vector offsets having captured syntactic and semantic regularities plays an important role in the task and are used in the embedding layer. Pretrained 50 dimensional vectors of GloVe (Pennington et al., 2014) and the embeddings trained on our corpus using Word2Vec (CBOW) (Mikolov et al., 2013) with the same dimensions were taken for the embedding layer. Different truncations were used to explore the effect of entropy while feeding the data into embedding layer. For the following CNN, LSTM and Bidirectional-LSTM, inputs are tokenized, concatenated, preprocessed, sequenced, padded for given lengths and fed into the networks, where the classification is done.

Base Convolutional Neural Network

The CNN networks employ the kernels to extract the local features which are passed through different layers and the classifications can be made. (Kim, 2014) used CNN on top of embeddings for the text classification and reached state-of-the-art results. Filter in CNN being of size (Filter size * Embed size) moves vertically down convoluting the n-grams (Considering the filter size). When the long-range semantic dependency corresponds to the stance, CNN's do not work very well due to the loss of contextual information leading to the class. So, we have employed CNN as baseline for this pipeline with varied truncations and different embeddings to analyze the behavior for classification.

Base LSTM and Bidirectional LSTM

(Schmidhuber et al., 1997) introduced the LSTM networks which process the information which has passed through the network using the hidden states whereas the bi-directional LSTM has the access to the information from both the directions (Forward and backward) thus keeping a better context of the words.We have used these Networks with different setups which are presented in the Figure 2.

Robust Bidirectional LSTM

(Davis and proctor, 2018) implemented the concatenated MLP, Bi-LSTM, Dual GRU architectures having separate embeddings for headline and body using GloVe with different features and reached the score of 89% on test-set. We tried to improve the model with robust layer structure extending the same architecture with headlines and bodies passing separately through the Bi-LSTM layer and used all the features from our knowledge base of the previous models and the similarity features. (Riedel et al., 2018)

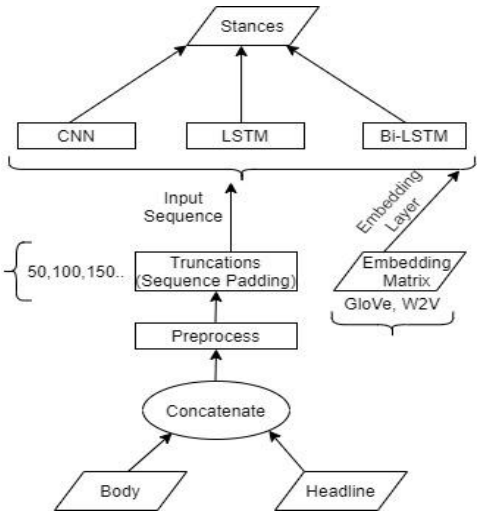

Figure 2: Neural Networks

In this architecture, five set of features were given as input to the layers. Pad sequenced body, headline(were fed to the embedding layer having GloVe embeddings separately) followed by the separate inputs to the Bi-LSTM, TF-IDF features for both and their cosine similarity were stacked, similarly all the stacked hand features from the baseline implementation, Text Blob sentiment polarity for both and their difference stacked together were fed into the network at different layers. These layers were concatenated, and the classification task was performed. Figure 3 shows the architecture.

3.3.2 Classical Models

To explore the performance of classical machine learning models, we implemented the Gradient Boosting, Logistic Regression, Random Forest and XGBoost algorithms using baseline feature extraction methodologies and the Gramian features tuned with different hyperparameter variations. With an assumption of independent inputs, logistic regression efficiently computes the MLE, thus it was chosen as the baseline for the pipeline and Random Forest was implemented being a suitable choice for high dimensional noisy data. XGBoost was chosen as an improvement over Gradient boosting as second order gradient computation of the loss function carried out, helps gather more entropy about the gradient direction which intends to minimize the loss. Relevant Features are extracted from Preprocessed headlines and bodies which are further stacked, normalized and fed into the classifier. Figure 4 depicts the architecture.

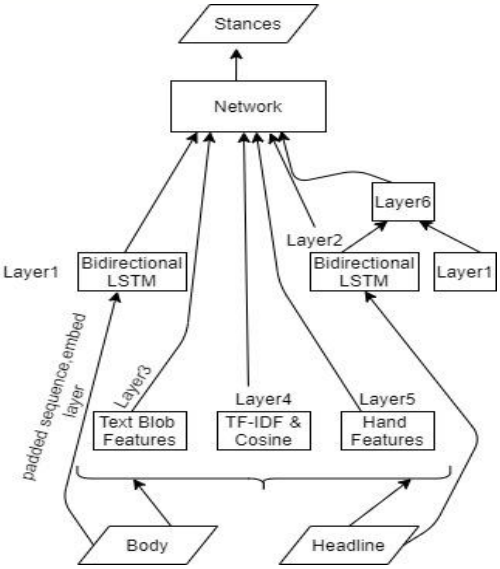

Figure 3: Robust Bidirectional LSTM

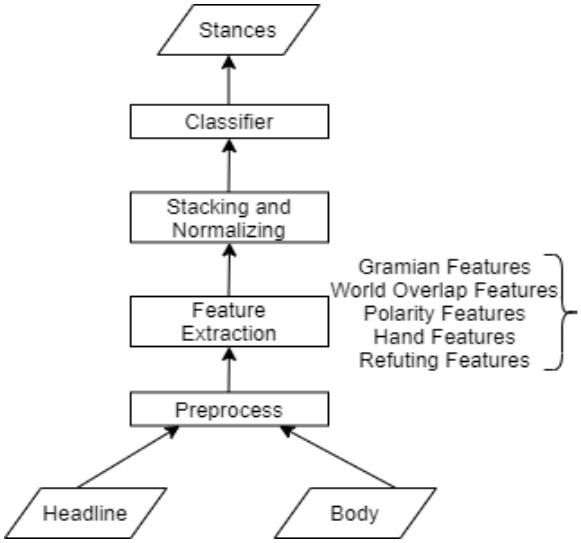

Figure 4: Classical Model

4 Experiments

4.1 Dataset

Emergent dataset(Ferreira and Vlachos, 2016) was taken as the dataset by Pomerleau and Rao for the FNC-1 competition. The training data comprises of total 49972 stances with 1683 bodies where each input consists of a 'Body Text', 'Headline' and a corresponding 'Stance'. On analyzing the dataset, a highly unbalanced class distribution was recorded with 73% of 'unrelated', 18% of 'discuss', 7% of 'agree' and 2% of 'disagree' stances.

After implementing the models on unbalanced dataset, best models were also implemented on undersampled dataset formed using 50% of the 'unrelated' stances keeping other 'stances' same.

4.2 Evaluation Metrics

Metric provided by FNC-1 is used which is quite robust and penalizes the equalized scoring scheme for 'unrelated' stances.

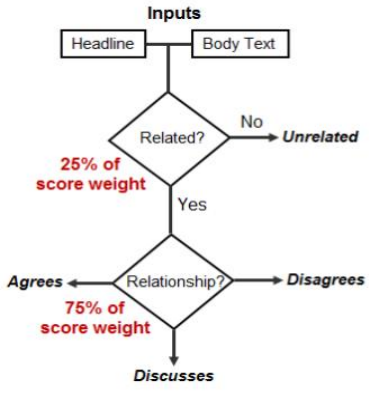

Figure 5: FNC-1 Metric

This metric is used to analyze the models for the bias towards the 'unrelated' stances provided in the dataset and to see if the models overfit for the given stances giving low recall on the other stances.
In addition to the default baseline metric, Confusion matrix, precision, recall, F-1 Scores(Classification Report) and unbiased accuracy(Actual accuracy) was used for the performance comparison between the models.

4.3 Configurations

The models were implemented on Colab and NVIDIA® GeForce® GTX 2060 Ti 6GB.
The classical Model approach discussed in **Section 3.3.2 and Figure 4** is implemented with the configurations shown in Table 1. These configurations are best obtained after k-fold validation on different hyperparameters. The Best Gradient Boosting model was also implemented on under sampled dataset.

Model	Parameters
Gradient Boosting	Estimators=100, subsample=0.9
XGBoost	objective= *"multi: softmax"*, num_class=*4*, max_depth=*2*, eval_metric=*"merror"*, n_rounds=*600*, early_stopping = *40*
Random Forest	max_depth=15, estimators=200
Logistic Regression	C=1.0, solver="lbfgs", max_iter=4000

Table 1: Classical Model configuration(Best)

Pretrained Bert with default configurations provided by SimpleTransformer was taken as the baseline. Dataset was simply fed into the model without any preprocessing.
The Neural Network pipeline discussed in **Section 3.3.1 and Figure 2,** is implemented having the different sequence length per model for concatenated headline and body. The average median body length is approximately 310, thus running the experiments using variable lengths help us determine the importance of various parts of the 'body' and the behavior of models in the presence of noise/long sequences and the sparsely padded inputs for the short articles. Three truncation lengths of 50,100,150 were taken for CNN, LSTM, Bi-LSTM keeping other parameters of the model constant. Embedding matrix was formed using the pre-trained 50- dimensional vectors of GloVe (Twitter corpus).
As discussed, CNN was considered as the baseline for this pipeline. 32 filters were taken in the Con1D layer having size 3, same padding and 'relu' as activation was taken for feature extraction. A fully connected Dense layer with 256 units as hidden layer followed by the output layer.
LSTM and Bidirectional LSTM were implemented with the respective layer(say LSTM) having 128 units followed by the same layer having 64 units, a dropout rate of 0.4 to prevent overfitting, a fully connected dense layer of 32 units with 'relu' as activation followed by a dropout rate of 0.2 and the output layer. All these were run on 3 truncation lengths viz. 50,100,150 i.e. 9 Models.
In order to understand the impact of different embeddings on the classification task, Best truncation length was recorded from the previous setting, and all the three models were implemented using 50 dimensional Word2Vec embeddings trained on our corpus with the best truncation length for input sequence and other parameters being kept same. This implementation was done to have a better vocabulary of vectors with words having minimum count of 8 and taking a window size of 3 to capture the strong context within the smaller ranges.
The robust Bi-directional LSTM's architecture as explained in **Section 3.3.1 and figure 3,** was implemented with the configurations explained in Table 2.

Parameters	Units
Embedding (Input Pad Seq. Features)	200(GloVe)
Max_len(headline)	30
Max_len(body)	50
Features TFIDF Count	4500
Bi-LSTM, Dropout (Input Pad seq Features)	128,0.3
Dense, Activation, Dropout (TFIDF)	128, 'ReLU',0.5
Dense, Activation, Dropout (Concatenated headline, body, TFIDF, hand features, polarity)	128, 'ReLU'

Table 2: Robust Bidirectional LSTM

For all these models, trainable was kept false in the embedding layer to prevent overfitting by not updating the weights at the layer, fully connected dense layer with 4 units having 'softmax' activation as the output layer, models were

compiled using 'categorical cross entropy' loss and 'Adam' optimizer with learning rate of 0.005 and 0.001 for Robust Network. Models were trained on batch size of 256 at 30 epochs each whereas the Robust Model was trained with the batch size of 64 and 15 epochs because of the time complexity.

Gradient Boosting was trained with a time complexity of 492 seconds whereas Robust bidirectional LSTM was trained in 5250 seconds when ran on Colab GPU. CNN's were seen to be the fastest at training within 60 seconds.

4.4 Results and Analysis

Initially, Models were evaluated on validation accuracy, competition accuracy metric provided by FNC-1 which gives less weight score to 'unrelated' stances and using unbiased accuracy. FNC-1 Baseline model achieved the test accuracy of 75.08%. Bert classifier which was also taken as the baseline model for the whole pipeline gave a validation accuracy of 90.76% and the test accuracy of 67.38% on competition metric. This was the expected performance from this classifier because of the static implementation using the SimpleTransformer library without any extended features and hyperparameter tuning.

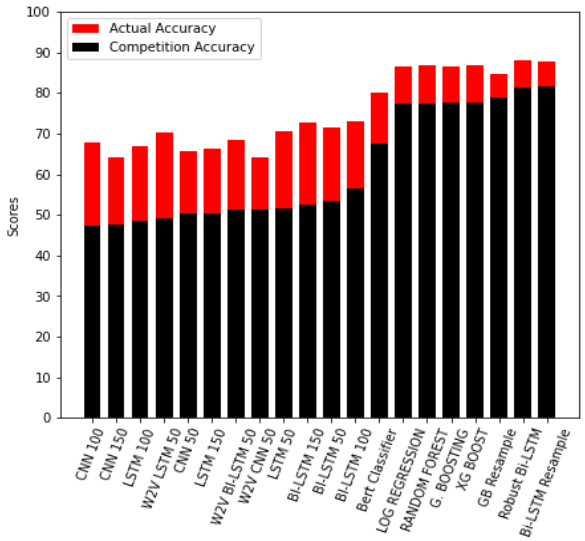

Figure 6: Competition Test Accuracy

The classical network architecture represented in Figure 4 when implemented with additional preprocessing and Gramian matrix resulted in the improved accuracies on validation and test set. An average increase of 2.5% in the test accuracy was observed in all the classical models which can be seen in Figure 6. On analyzing the behavior, it was seen that the stances labelled as 'discuss' were classified better than in the baseline implementation of gradient boosting. Reason behind this increase can be considered as the usage of Gramian matrix which extracted a new feature with cosine similarity between the tfidf of headline and body and helps mapping 'unrelated' misclassifications into correct 'discuss' stances. A steep drop in the 'discuss' mis-classifications as 'unrelated' and increased correct classifications for 'discuss' by a rate of 8% was seen due to this feature which was the expected behavior.

Further, In the Neural Network pipeline, we analyzed the behavior of sequences with truncating length and the word embeddings with CNN, and RNN as presented in figure 2.

Fifty dimensional pretrained GloVe embeddings with the input sequence's truncation length of 50 was seen outperforming the sequences with longer truncation lengths of 100 and 150 in CNN and LSTM. The validation accuracy for CNN and LSTM with sequence length of 50 was 85.65% and 93.68%. It can be inferred that these models at some point were not able to handle the noisy data and the sparse data created after padding the shorter input bodies as the test set accuracies reached were 50.32% and 51.66% at the competition metric whereas the actual accuracies also suffered at 65.5% and 70.5% respectively. Validation accuracy and the loss is shown in figure 7.

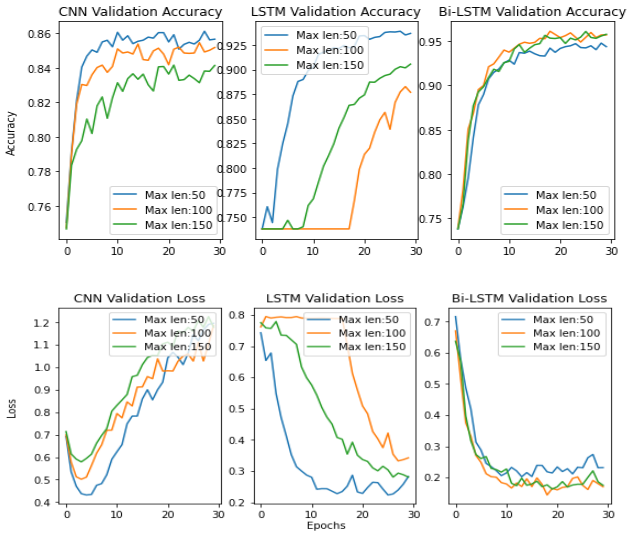

Figure 7: Val Loss and Accuracy of Neural Networks

The competition test accuracy achieved by the Bidirectional LSTM specified in section 3.3.1 was 56.55% at competition metric and actual unbiased accuracy was 73%. Our assumption of different body areas having varied importance as mentioned in section 4.3 was proven wrong for this dataset by Bidirectional LSTM giving high accuracy on the sequences at almost all the truncation lengths as these models process the information in both the directions and encode the structure dependent semantics of the input which can be seen in our results for classification.

Here our intuition made in Section 3.3.1 regarding CNN's not performing well for the local feature extraction when long-range semantic dependency corresponds to the stance is proven correct because Bidirectional LSTM can be seen performing well for the classification which can be due to the contextual semantic encoding. Loss in CNN can be seen high because of the big differences in the probability ratios whereas the validation accuracy for the same was increasing.

Word2vec embeddings were seen better than the GloVe for all the models at best truncation length of 50 which was achieved from the previous setting. Highest competition accuracy of 52% and 68% at both accuracy metrics were achieved by Word2Vec with Bidirectional LSTM. Word2Vec embeddings were trained on our corpus and a lower window size resulted in the better embeddings which can be seen in the results. Figure 8 shows the validation accuracies of models at sequence length of 50 for the GloVe and Word2Vec.

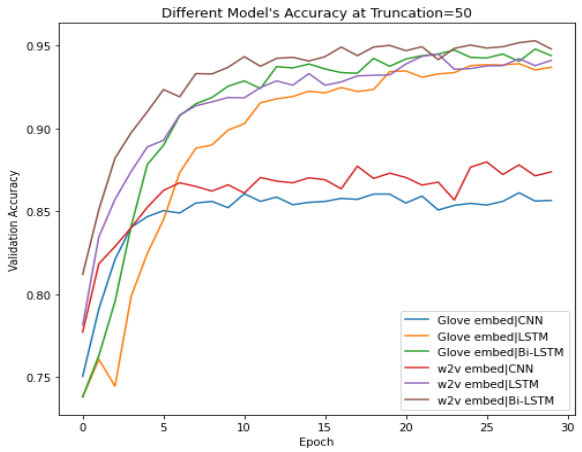

Figure :8 Neural Networks Validation Accuracy

Our best model turns out to be the Robust Bidirectional LSTM which provided the test accuracy of 81.19% and 88.17% at both the metrics. With all the features and separate processing of body and headline, it was the expected behavior from this network.

Best models obtained from the different pipelines viz. Gradient Boosting, Robust Bidirectional LSTM were further implemented on the under sampled dataset to analyze the effect of unbalanced dataset and an increase of 0.53% of accuracy was seen in case of Robust Bidirectional LSTM and an increase of 1.02% in case of Gradient Boosting.

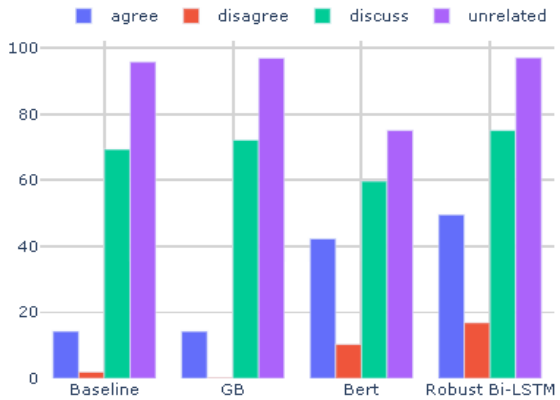

Figure 9: F-1 Scores on Test data

Figure 9 shows the F-1 Scores of our best models and it can be inferred that the Robust Bidirectional LSTM proved to be effective in correct classifications majorly for 'disagree' and 'agree' stances followed by the Bert classifier which lags behind the Gradient boosting classifier on classifying 'unrelated' and 'discuss' stances only. Thus, without any preprocessing and parameter tuning, SimpleTransformer's BERT handled the problem of unbalanced data better than our other models leaving Robust Bi-LSTM out. Our Robust model exceeds the threshold of 0.5 for 'agree', 'discuss' and 'unrelated' when compared on precision and exceeds on 'discuss' and 'unrelated' for recall. Our model only underperforms for the 'disagree' stance giving a recall of 0.11 which is still better than the other models. It can be inferred that our model was not able to learn from the small number of 848 'disagree' stances and it made 619 misclassifications for the 'disagree' class out of total 697 'disagree' stances. It makes the highest

misclassifications of ‘disagree’ to ‘discuss’ class which is 47% of the total misclassifications.

Body: MAIDUGURI, Nigeria, the leader of Nigeria Islamic extremist group Boko Haram has denied agreeing to any cease-fire with the government and said more than 200 kidnapped schoolgirls all have converted to Islam and been married off.

Headline 1: Nigeria Claims Boko Haram Will Release the Kidnapped Schoolgirls

Stance: discuss

Headline 2: Boko Haram ceasefire ignored as violence flares in Borno state

Stance: disagree

Models trained on such dataset with very few inputs for a particular class generally underfits for that class which has happened in our case of above example for the stance ‘disagree’. Also, the semantic similarity between both the cases is very high in many input patterns. Thus, all the models mostly misclassify the stance ‘disagree’ to ‘discuss’ but our Robust bidirectional LSTM still gives a recall of 0.11.

5 Conclusion

A detailed comparison between the effects of word embeddings, feature engineering, truncation lengths and different DNN’s and classical models was presented. In depth analysis of model performance on various parameters reflected the importance of features viz. TFIDF, cosine similarity, n-gram feature sets and the separate processing of two texts contribute directly towards the performance. It was seen that the fine-tuned architecture of RNN’s outperform all the models tried at our settings. Our model achieved an accuracy of 81.19% at competition test set which is an improvement of 6.11% from the baseline which is a decent improvement. We were also introduced to the highly unbalanced dataset, on which our model performed well as per the capacity and the undersampling technique also resulted in the improved results. As a future work, we are generating the artificial dataset using techniques like Smote. Bert classifier achieved drastic improvement without any preprocessing. We are currently working on Bert as a service and other feature engineering techniques to focus on the classification of undersampled classes.

References

1. Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis and Sebastian Riedel. May 2018. A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. arXiv: 1707.03264.
2. Stephen R. Pfohl. Oskar Triebe and Ferdinand Legros. “Stance Detection for the Fake News Challenge with Attention and Conditional Encoding”.2017.
3. Sean Baird and YuxiPan, June 2017.”Talos Targets disinformation with Fake News Challenge victory”.
4. Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. Efficient Estimation of Word Representations in the Vector Space. January 2013. arXiv:1301.3781
5. Jurgen Schmidhuber and Sepp Hochreiter, 1997. Long Short-term memory. Neural Comput 9(8):1735– 1780.
6. Mike Schuster and Kuldip K Paliwal.1997. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11):2673–2681.
7. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008.
8. ThilinaRajapakse.2020.SimpleTransformers. https://simpletransformers.ai
9. Zhiguang Wang and Tim Oates, 2015. Spatially Encoding Temporal Correlations to Classify Temporal Data Using Convolutional Neural Networks arXiv: 1509.07481
10. Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543.
11. Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. arXiv:1408.5882v2.
12. Richard Davis, Chris Proctor, 2018. Fake News, Real Consequences: Recruiting Neural Networks for the Fight Against Fake News.
13. FERREIRA, W., AND VLACHOS, A. Emergent: a novel dataset for stance classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, ACL.