Stance Detection with Neural Networks(FNC-1)

Jeswin Maria Cinthia Palthasar
ID: 20837700
Department of Management Sciences
University of Waterloo

Abstract

Fake news challenge was conducted in 2017 to develop multi-class text classification tool to overcome the problem of spreading fake news using Artificial Intelligence. The MSCI 641 fake news challenge is based on the FNC stage 1. The datasets and a Baseline with 79.5% accuracy were provided. The objective of this project is to utilize Natural Language processing algorithms to develop a solution for fake news classification. As part of the challenge, supervised learning techniques such as decision tree, neutral networks such as RNN, CNN were implemented for Stance Detection. Among the classification models and Networks Bi-directional LSTM with Glove embedding, BILSTM with Word2Vec and CNN and Bi-directional LSTM(Hybrid) with Word2Vec embedding has given the best accuracy around 95%.

1 Introduction

Fake news, explained by the New York Times as "a made-up story with an intention to deceive" [1], is of the most important problem faced in journalism. In a December Pew Research poll, 64% of US adults said that "made-up news" has caused a "great deal of confusion" about the facts of current events[2]. Evaluating the truthfulness of a news story is a complex task and it is difficult even for experts. But the evaluation can be done in a series of different stages. The automation of identification of fake news is called Stance Detection. Stance Detection is the process of assessing the relative perspective (or stance) of two pieces of text relative to a topic, claim or issue. The task of estimating the stance of a body text from a news article relative to a headline[3]. Specifically, the body text may agree, disagree, discuss or be unrelated to the headline. The version of Stance Detection selected for FNC-1 extends the work of Ferreira & Vlachos [4]. In this challenge, Sklearn classifiers and Neural Networks were built for classification. In the initial stage, with the provided baseline, a modified baseline with different boosting algorithms and added features were developed. In the next stage different Neural networks such as RNN and CNN were implemented and analyzed.

2 Description of Data

Datasets including training dataset and test dataset are provided in the fake news challenge github page. The Training dataset and validation dataset contain Headline and Body are provided in separate csv file. The merged dataset with headlines and bodies have a total of 49972 rows. The test data has a total of 25413 rows. Each article-body pair has been labeled with a stance such as Unrelated, Agrees, Disagree and Discuss, while the test dataset lack labels and needs to predict each row and label it by Machine learning models. The distribution of stances is given below. It is noted that majority of stances are Unrelated.

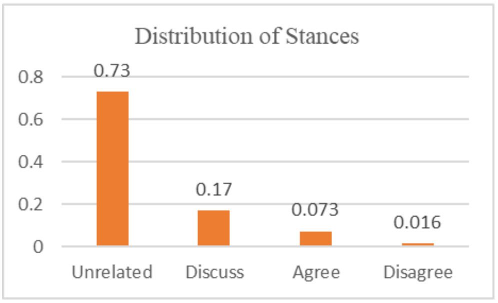

Fig 1: Visualization of distribution of Stances

3 Official Baseline

The baseline is based on GradientBoosting classifier with hand-coded features. The baseline implementation also includes code for pre-processing text, splitting data carefully to avoid bleeding of articles between training and test, k-fold cross validation, scorer.The hand-crafted features include word/ngram overlap features, and indicator features for polarity and refutation[5].

3.1 Scoring System

The accuracy will be evaluated based on a weighted, two-level scoring system:

- Classify headline and body text as *related* or *unrelated* 25% score weighting
- Classify related pairs as *agrees*, *disagrees*, or *discusses* 75% score weighting[5].

The Stance Detection task (classify as agrees, disagrees or discuss) is more difficult than identifying as unrelated. Hence the former is given more weightage than latter

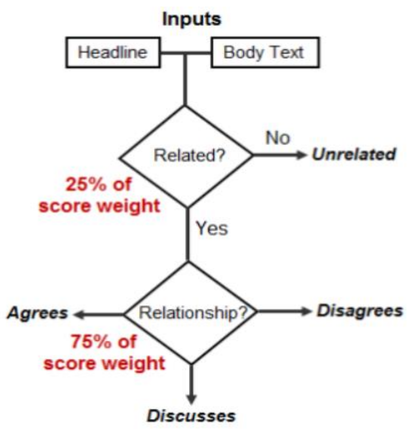

Fig 2: Scoring System

4 Modified Baseline

4.1 Overview

The provided baseline has been modified with different boosting algorithms like Adaboost and XGBoost. On top of the hand coded features, Cosine Similarity between headline and article body is added to improve the classification.

4.2 Layout

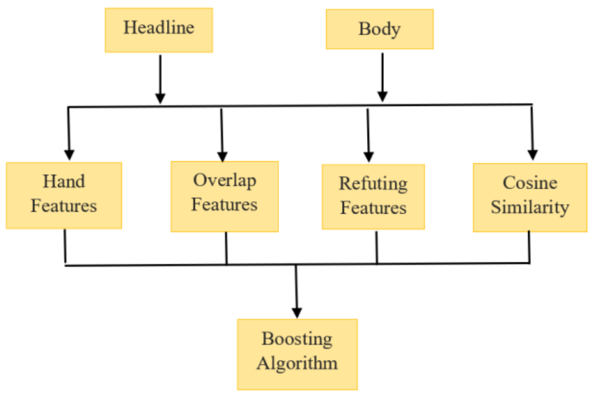

Fig 3: layout of Modified baseline

4.3 Related Work

The model submitted by UCLMR at the fake news challenge(2017) who came at third place has included cosine similarity between TF-IDF vectors of the headline and body at the beginning of the model before passing through multi-layer perceptron[6]. This feature is added in modified baseline along the other feature such as word/ngram overlap features, polarity features and refutation features.

4.4 Data Preprocessing

The dataset is split in the ratio of 0.9 :0.1 to extract validate dataset. The data is tokenized using Tfidf tokenizer with Unigram+Bigram features and stopwords are removed.

4.5 Feature Extraction

The script for word/ngram overlap features, polarity features and refutation features are taken from provided baseline. The cosine similarity between the TF-IDF vectors of the headline and body is calculated and stacked along with other feature vectors.

4.6 Boosting Algorithm

The processed training data along with is features is inputted into the classifier

4.6.1 Adaboost

A standard Adaptive boosting algorithm is used for classification. Parameters such as learning rate and n_estimators were chosen with GridSearch. This classifier reports accuracy of 78.02% on the validate dataset and weighted score of 1768.5 in validate dataset.

4.6.2 XGboost

A standard XGBoost algorithm is used for classification. Parameters such as learning rate and n_estimators were chosen with GridSearch. This classifier reports accuracy of 77.23% and weighted score of 1767.5 in validate dataset.

4.6.3 Results and Analysis

Model	Test accuracy	F1-Score	Recall	Precision
ADA-BOOST	74.9%	0.8265	0.844	0.8143
XG-BOOST	74.44%	0.834	0.849	0.822

Table 1: Results of modified baseline

The test accuracy of ADAboost is slightly better than XGboost. Both of the algorithms are based on converting weak learners to strong learning by updating the residuals. But the f1-score is a bit better for XGboost. It denotes that regularization is better in case of XGboost.

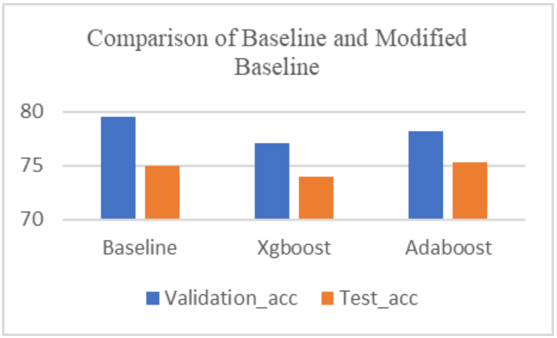

Fig 4: Comparison of results of modified baseline

The modified baseline with different boosting algorithms has given a performance nearly equal to the baseline.

5 Neural Networks

5.1 Overview

The aim of the project is to use different neural networks to classify the stances. In this project four neural networks were built. A baseline model of Long short-term memory (LSTM) was built. This baseline is improvised with combination of Convolutional neural network. Another model with Bidirectional LSTM and combination of Bidirectional LSTM and CNN were built for stance detection. All the four models used pretrained embeddings such as glove and word2vec.Totally 8 models were used for this project.

5.2 Data Preprocessing

The raw data has to be processed to give it as an input to the neural networks. The initial step is to merge the headline, bodies and stances into a single data frame. The Stances are integer coded as they have been encoded in the baseline. The headline and bodies are converted into list of words i.e. they have been tokenized using text_to_word_sequence function in keras. This function splits the words, removes special characters and converts text into lower cases[7]. Then list of Headlines and bodies have been concatenated. This concatenated list of texts has to be converted to integers. This done by using Tokenizer in keras. The text_to_sequence function converts the text into sequence of integers. It is important to feed the input with fixed length to the neural network. The input has varying lengths of the sentences. Hence the input is padded before feeding into the neural network.

Pre-trained were used as in the embedding layer. Both Glove and Word2vec were used. GloVe stands for "Global Vectors for Word Representation". The 100-dimensional GloVe embeddings of 400k words computed on a 2014 dump of English Wikipedia was used in this project. [8]. Word2Vec is a pretrained word embedding developed by Google. Word2Vec is trained on the Google News dataset (about 100 billion words)[9]. A matrix is created for one embedding for each word in the training dataset. It is done by enumerating all unique words in the *Tokenizer.word_index* and locating the embedding weight vector from the loaded pretrained embeddings.

5.3 LSTM

Long Short-Term Memory networks are a specific kind of RNN, capable of learning long-term dependencies[10]

5.3.1 Layout

The concatenated headline and body vectors are given as the input to LSTM with 100 hidden units. The pretrained vectors GloVe and Word2Vec are used as embedding. Dropout has been used to avoid overfitting of the data. A dropout of 0.5 is used. A final layer with softmax activation function is used.

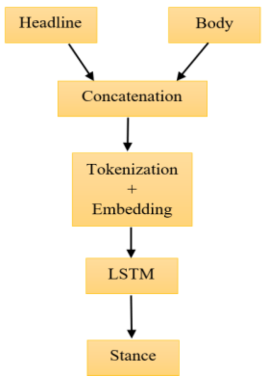

Fig 5: Layout of LSTM

5.4 CNN and LSTM (HYBRID)

Convolutional Neural Networks (CNN) is a multilayer neural network that can detect information in different positions with better accuracy. One-dimensional CNN and max pooling layers can be added on top of LSTM after the embedding layer of the model.

5.4.1 Layout

The concatenated headline and body vectors are given as the input to CNN with Convolution layer of 64 filters with relu activation function and LSTM with 100 hidden units. The pretrained vectors GloVe and Word2Vec are used as embedding. A dropout of 0.5 have been used to avoid overfitting of the data. A final dense layer of Sigmoid Activation function is used.

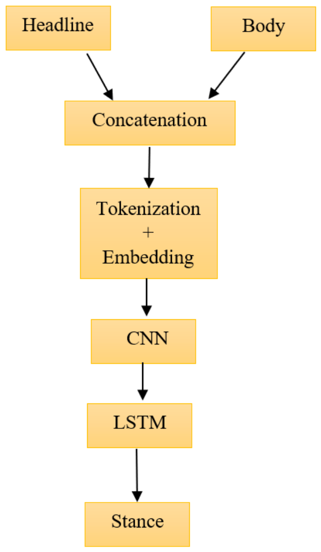

Fig 6: Layout of LSTM+CNN

5.5 BILSTM

Bidirectional LSTMs is an extension of regular LSTMs which can improve the performance of models for sequence classification problem. Bidirectional LSTMs can be considered as two LSTMs that the input sequence train on the first LSTM and a reversed copy of the input train on the second one[11].

5.5.1 Layout

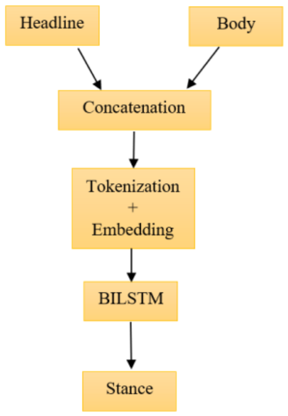

Fig 7: Layout of BILSTM

The concatenated headline and body vectors are given as the input to BILSTM with 100 hidden units. The pretrained vectors GloVe and Word2Vec are used as embedding. Dropout of 0.5 has been used to avoid overfitting of the data. A final layer with softmax activation function is used.

5.6 CNN and BILSTM (HYBRID)

One-dimensional CNN and max pooling layers can be added after the Embedding layer which then feed the consolidated features to the BILSTM.

5.6.1 Layout

The concatenated headline and body vectors are given as the input to CNN with Convolution layer of 64 filters with relu activation function and BILSTM with 100 hidden units. The pretrained vectors GloVe and Word2Vec are used as embedding. A dropout of 0.5 have been used to avoid overfitting of the data. A final dense layer of Sigmoid Activation function is used.

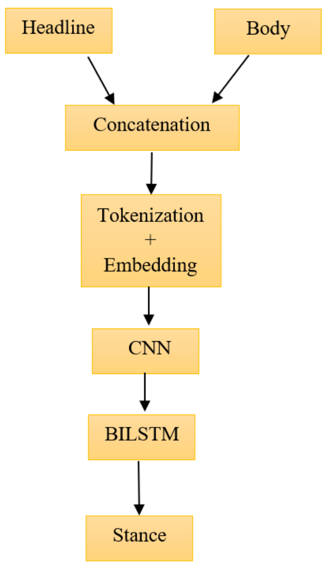

Fig 8: Layout of BILSTM+CNN

6 Results and Analysis

The training dataset has been split into 0.9:0.1 ratio to extract validate dataset. The results on the validate dataset with respective to training time is shown in the below table. The model is run on test dataset and the accuracy is reported.

Model	GloVe	
	Validation Accuracy	Training Time for 1 epoch(sec)
LSTM	90.74%	239-262
LSTM + CNN	92.04%	80-83
BILSTM	95.4%	469-480
BILSTM+ CNN	93.16%	150-170

Table 2: Validation accuracy for model with GloVe Embeddings

Model	Word2Vec	
	Validation Accuracy	Training Time for 1 epoch(sec)
LSTM	79.39%	360-385
LSTM + CNN	91.6%	141-144
BILSTM	94.98%	583-600
BILSTM+ CNN	95.16%	234-240

Table 3: Validation accuracy for model with Word2Vec Embeddings

Model(with GloVe)	Test Accuracy	F1-score	precision	recall
LSTM	39.23	0.603	0.549	0.704
LSTM+ CNN	38.41	0.582	0.5717	0.6128
BILSTM	40.10	0.559	0.589	0.515
BILSTM+ CNN	35.74	0.5617	0.567	0.560

Table 4: Evaluation metrics for model with GloVe Embeddings

Model(with W2V)	Test Accuracy	F1-score	precision	recall
LSTM	38.45	0.5465	0.553	0.5483
LSTM+ CNN	41.13	0.611	0.566	0.681
BILSTM	40.40	0.5489	0.5589	0.5515
BILSTM +CNN	43.08	0.567	0.5959	0.519

Table 5: Evaluation metrics for model with Word2Vec Embeddings

6.1 Accuracy

6.1.1 Models with GloVe Embedding

The BILSTM model has reported the highest accuracy with 95%. BILSTM manages the inputs in two ways, one from past to future and one from future to past[11]. It preserves information from the future and past, thereby understanding the contextual information in inputs[12]. Hence it has given the best accuracy.

The next best accuracy was given by CNN+BILSTM with 93%. Addition of CNN has reduced the training time, but the accuracy is slightly less than former(BILSTM) best model.

The standard LSTM has given the least accuracy. But addition of CNN to standard LSTM have improved the accuracy.

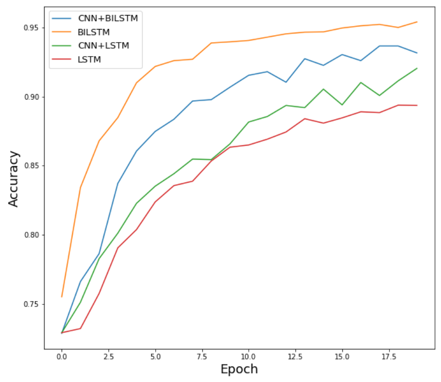

Fig: 9 Graph between accuracy and no of Epoch for models with GloVe Embeddings

6.1.2 Models with Word2Vec Embedding

The BILSTM and BILSTM+CNN model has given the best accuracy around 95%. It is noticed from the graph that the accuracy for CNN+BILSTM and BILSTM are nearly equal.

It is also noticed that CNN+LSTM accuracy is much better than standard LSTM. The addition of CNN to standard LSTM have improved the accuracy significantly. The standard LSTM has reported the least accuracy.

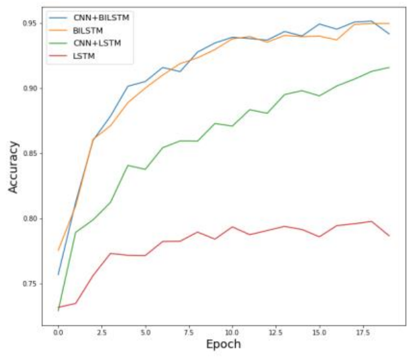

Fig: 10 Graph between accuracy and no of Epoch for models with Word2Vec Embeddings

6.2 Training Time.

The models training time for 1 epoch is given in the table. It is noticed that Hybrid models(CNN+LSTM/BILSTM) have taken less timing for training. The addition of CNN for feature extraction has effectively reduced the training time by more than half of the standard models. Another observation is that BILSTM models have taken more time for training than LSTM models.

6.3 Loss

From the graph plotted between validation loss and no of epoch it is evident that the loss of LSTM for models with both GloVe and Word2Vec embeddings are high compared to other models. Addition of CNN layers to basic LSTM has reduced the loss and improved the accuracy. The models with BILSTM has less loss when compared to other models.

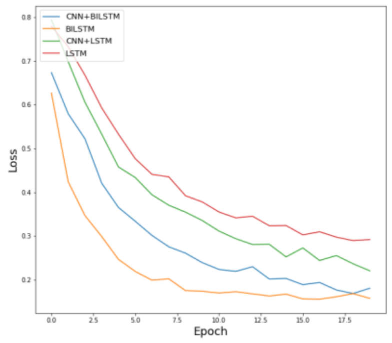

Fig: 11 Graph between loss and no of Epoch for models with GloVe Embeddings

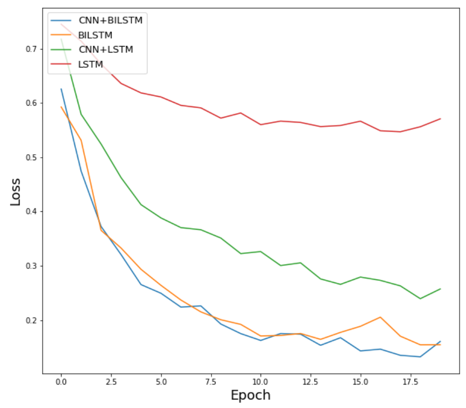

Fig: 12 Graph between loss and no of Epoch for models with Word2Vec Embeddings

6.4 f1-Score

F1-Score is an important metric when the dataset is unbalanced. In this challenge, stances are not uniformly distributed, hence the weighted average of f1 score is calculated. The f1- score of LSTM models are slightly better than other models(0.54-0.6). It means that the LSTM has low false positives and low false negatives. But the accuracy is comparatively lower than BILSTM.

6.5 Hyper-Parameters Tuning

The model was trained on default parameters at the initial stage. Many parameters were tuned with trial and error approach. It is noticed that as the batch size and the no of epoch increased there was a considerable improvement in accuracy. To avoid the problem of over-fitting dropout rate 0.5 is used at the hidden layers of all the four neural networks. The advisable dropout is 0.2-0.5. The dropout of 0.5 has given good accuracy. 100 hidden units were used in LSTM. The default learning rate is used. Since it is multi class classification problem, categorical_crossentropy is used to evaluate the loss and adam optimizer was used. It is recommended in Keras documentation to use Trainable=False when using the embedding layer in Keras to prevent the weights from being updated during training[13]. The following table gives the parameters used in the models

Parameters	Values
Batch Size	100
Epoch	20
LSTM Units	100
Dropout Rate	0.5
Trainable	False
Learning rate	0.001
CNN Filters	64

Table 6 : Tuned Parameters

6.6 Pre-trained Vectors

All the models had an embedding layer of GloVe or Word2Vec pretrained Vectors. There is a difference in performance between the same models with different vectors. In most of the cases, GloVe has given better accuracy than Word2Vec. It is also observed that training time for one epoch is more in case of Word2vec. The reason is that it might be due to the difference in dimension of embeddings used.

7 Future Work

7.1 Simple Transformers

The Sequential processing in LSTM put a restriction to the model to be trained in parallel. RNN works in a way that past information is retained through hidden states i.e. information is retained by previously computed hidden states. This problem of recursion could be avoided by using Bi-directional models, but only to a certain extent. Similarly, in case of CNN that the number of different kernels needed to get the dependencies among all combination of words in a sentences would be cumbersome because of large number of combinations when increasing the maximum length size of input sentences[14]. The above problems can be avoided using Simple Transformers. ***Transformers*** allow parallel computations and avoids recursion (to reduce training time) and also to reduce drop in performances due to long dependencies. The main characteristics are:

- **Non sequential**: The input sentences are processed as a whole instead of word by word.
- **Self-Attention**: This layer computes similarity between words in a sentence.

- **Positional embeddings**: It uses fixed or learned weights which encode information related to a specific position of a token in a sentence.[14]

This transfer learning technique for classification does not require any preprocessing of data. The whole classification process could be done within three lines of code. In this project BERT transformer was used because, BERT uses an attention mechanism that leans the contextual relationship between the words[15]. This property is an important needed for text classification. The training time was taking more than 12 hours for one epoch. Due to the technical limitation (Disk space ,connectivity, RAM requirements) whole process wasn't able to get completed. I hope to complete this model using different coding platforms.

8 Conclusion

- BILSTM models have given better accuracy than other models. All the BILSTM models have given accuracy above 93%
- LSTM models and LSTM+CNN models gave less accuracy when compared with other models.
- Addition of CNN layers to the LSTM model have reduced the training time and loss.
- The accuracy of BILSTM with GloVe is 95%
- The accuracy of BILSTM and BISTM+CNN with Word2Vec Embedding is approximately 95%
- The performance difference between different pretrained vectors were observed for different models.

References

[1]-The New York Times-As Fake News Spreads Lies, More Readers Shrug at the Truth https://www.nytimes.com/2016/12/06/us/fake-news-partisan-republican-democrat.html

[2]Pew research center
http://journalism.org/2016/12/15/manyamericans-believe-fake-news-is-sowing-confusion

[3]Dean Pomerlau and Delip Rao- Fake News Challenge
http://www.fakenewschallenge.org/

[4] William Ferreira and Andreas Vlachos, "Emergent: a novel data-set for stance classification"

[5]MSCI-641 Fake news challenge

[7] Keras: text_to_word_sequence
https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/text_to_word_sequence

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation.

[9]Word2Vec
https://code.google.com/archive/p/word2vec/

[10] Christopher Olah- Understanding LSTM
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11] Jason Brownlee-How to develop BILST for sequence classification
https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/

[12]Difference between LSTM and BILSTM
https://intellipaat.com/community/8906/whats-the-difference-between-a-bidirectional-lstm-and-an-lstm

[13] Keras : Using pre trained word embeddings
https://blog.keras.io/using-pre-trained-word-embeddings-in-a-keras-model.html

[14]Transformer better than RNN and LSTM
https://ai.stackexchange.com/questions/20075/why-does-the-transformer-do-better-than-rnn-and-lstm-in-long-range-context-depen

[15]BERT: State of the Art NLP
https://www.kdnuggets.com/2018/12/bert-sota-nlp-model-explained.html

[6] UCL Machine Reading - FNC-1 Submission(2017)
https://github.com/uclnlp/fakenewschallenge