TOPIC FAKE NEWS DETECTION

Group No - 33

Rishabh Gupta(MT21070)

Rajat Pal(MT21138)

Shivnath Singh Gaur(MT21085)

Problem Statement

1. In today’s world it is very easy to spread fake news through social media and it becomes very obvious for people to believe on it because of the words that this fakesters use with the combination of corresponding images
2. In current scenario it is a common practice to make fake news to grab the attention of the viewers for making money using views. It result in:
   - Misinformation lead to loss in business.
   - Misinformation spread wrong intuition in society

Word Embedding

1. TF-IDF (**Term Frequency — Inverse Document Frequency**)

TF -IDF = TF(t,d) * IDF(t)

- In TF-IDF word vector is depends upon the frequency of that word in the whole data-set, that’s why first we split the dataset in training and testing and then we apply TF-IDF in both the training and testing separately

2. Word2Vec — It is Unsupervised algorithm for creating embedding , It learns meaningful relations between words and then encode that relatedness into vector similarity. For our models we have used embedding size of 200/250/300 and window size of 5/10/ which tell the context i.e how many total words(backward and forward of that word) is considered when predicting the relations. Also we used CBOW (Continuous Bag of Words)model of Word2Vec In CBOW surrounding wordsare combined to predict the word in the middle.

Evalution metrics Used

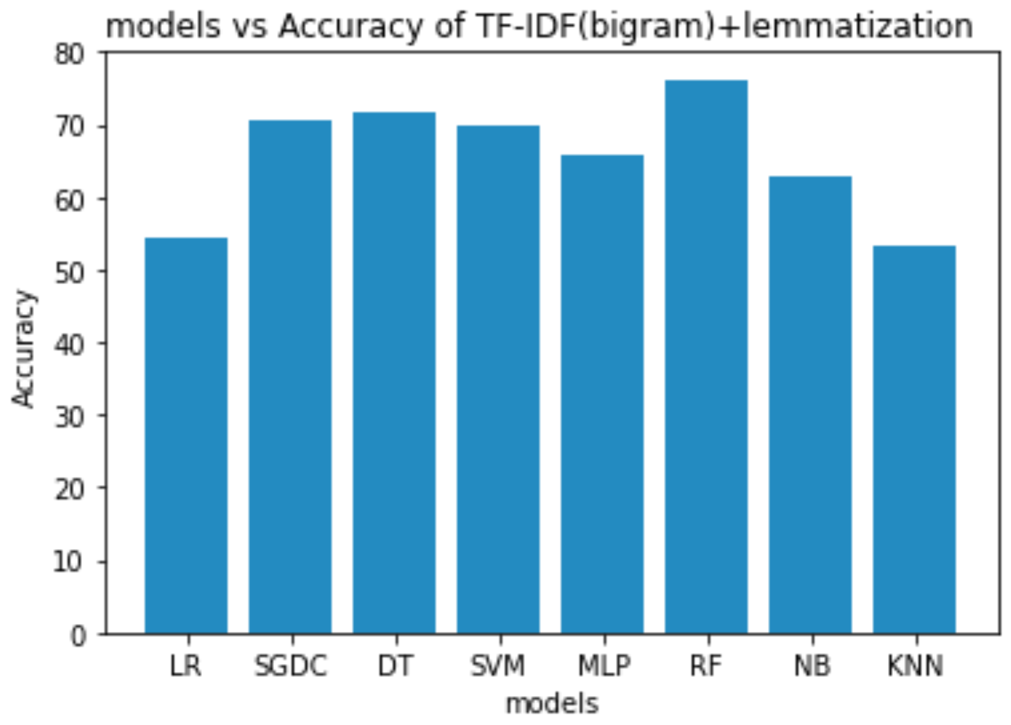
-> our task is binary classification task and also data-set is balanced so it is important to classify all true and false sentence correctly that's why we choose accuracy for evaluation metric
-> F1 score is right choose here because it's define as harmonic mean of precision and recall
-> we choose loss function to define loss in neural network training.

1. Accuracy — Out of all the predictions made by the model, what percentage did it get correct.
   Accuracy = T.P+T.N/Total
2. F1 Score — This metrics takes both precision and recall into account.
3. Confusion Matrix — show performance of task , This matrix show all TP, TN, FP, FN values for supervised learning algorithm.
4. Loss Function — Cross Entropy or Logarithmic loss for Neural network models, It is calculated as the average cross entropy
5. across all examples,   and smaller values represent a better model than larger values.

Baseline Models

Each model Below is trained and tested on word embedding TF-IDF with unigram , bigram, trigram, fourgram

1. Logistic Regression
2. Stochastic Gradient Descent Classifier
3. Decision tree Classifier
4. Support vector machine,
5. MLP classifier
6. Random forest,
7. Naive Bayes



Final Models

CNN ARCHITECTURE

Convolution Layer
Filter=32,kernel size=3
activation func =relu

⬇

Max pooling 1D
Pool size =3

⬇

Convolution Layer
Filter=64,kernel size =3
activation fun =relu

⬇

Max pooling 1D
Pool size =3

⬇

Convolution Layer
Filter=128,kernel size=3
activation func =relu

⬇

Global Max pooling 1D

⬇

Dense Layer
Output
unit=1, activation=sigmoid

BILSTM FLOW

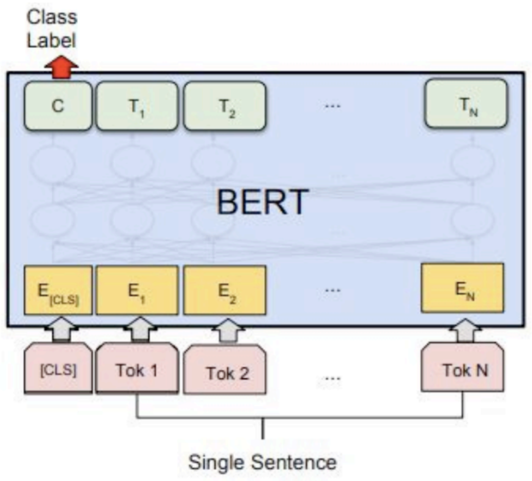
Embedding
Input length = 1000

⬇

BILSTM
Units =32

⬇

Dense Layer
Output unit =1
Activation =sigmoid



RNN ARCHITECTURE

Embedding
Input length = 1200

⬇

LSTM
Units =100

⬇

Global Max pooling 1D

⬇

Dense Layer
Output unit=1
activation=sigmoid

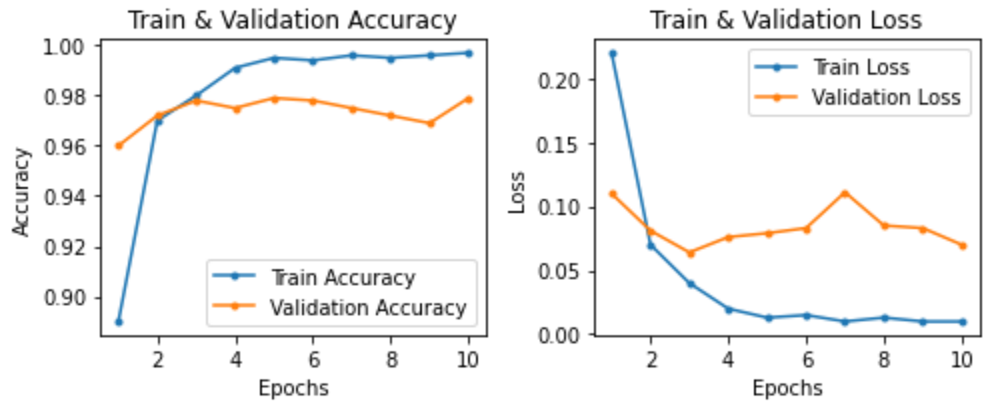
Project Control Flow

Dataset
⇩
Preprocessing
⇩
Word2Vec Embidding
⇩
Keras Embidding
⇩
CNN
⇩
Dense Layer
⇩
Prediction

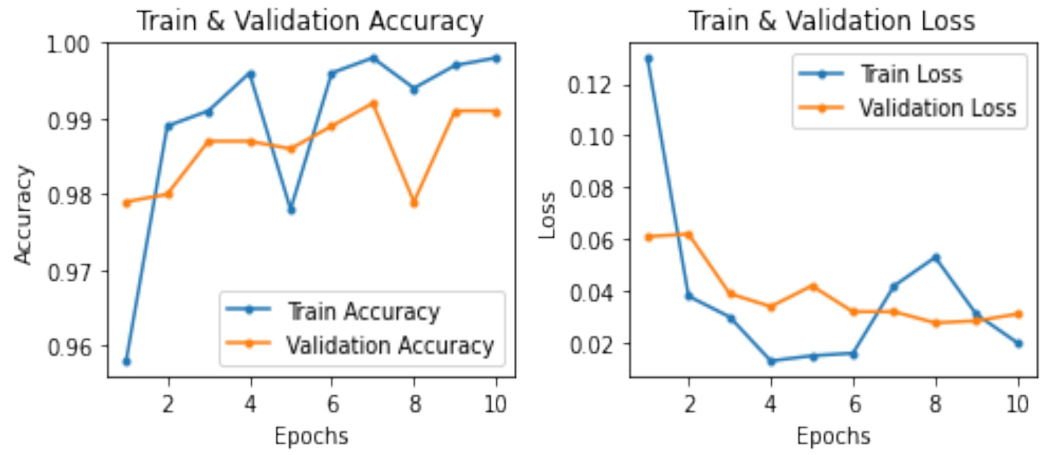
Result

This slide shows Train & Validation accuracy and Loss for each model

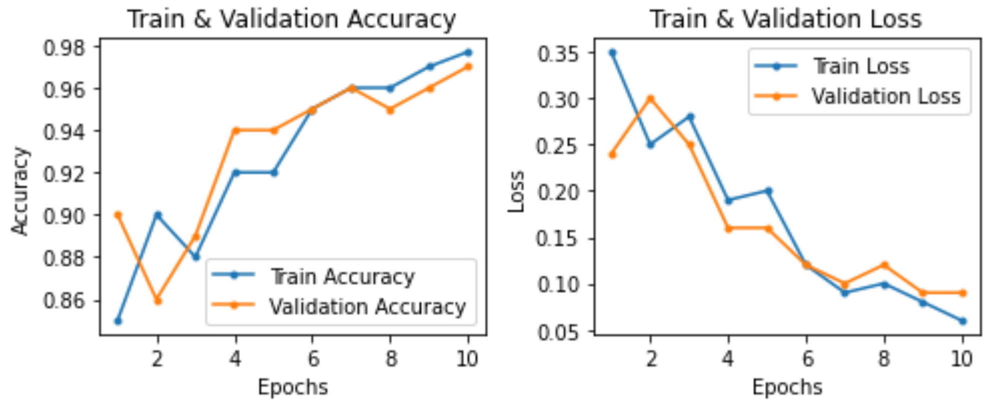
W2Vec+CNN



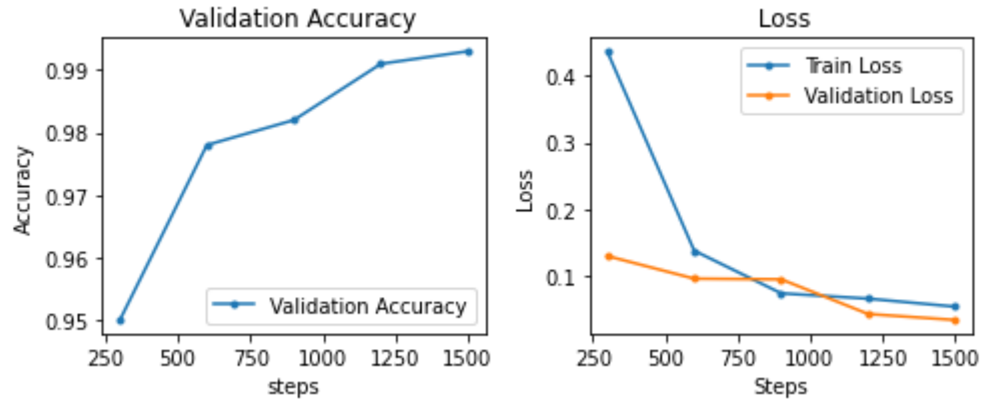
W2Vec+RNN



W2Vec + BILSTM



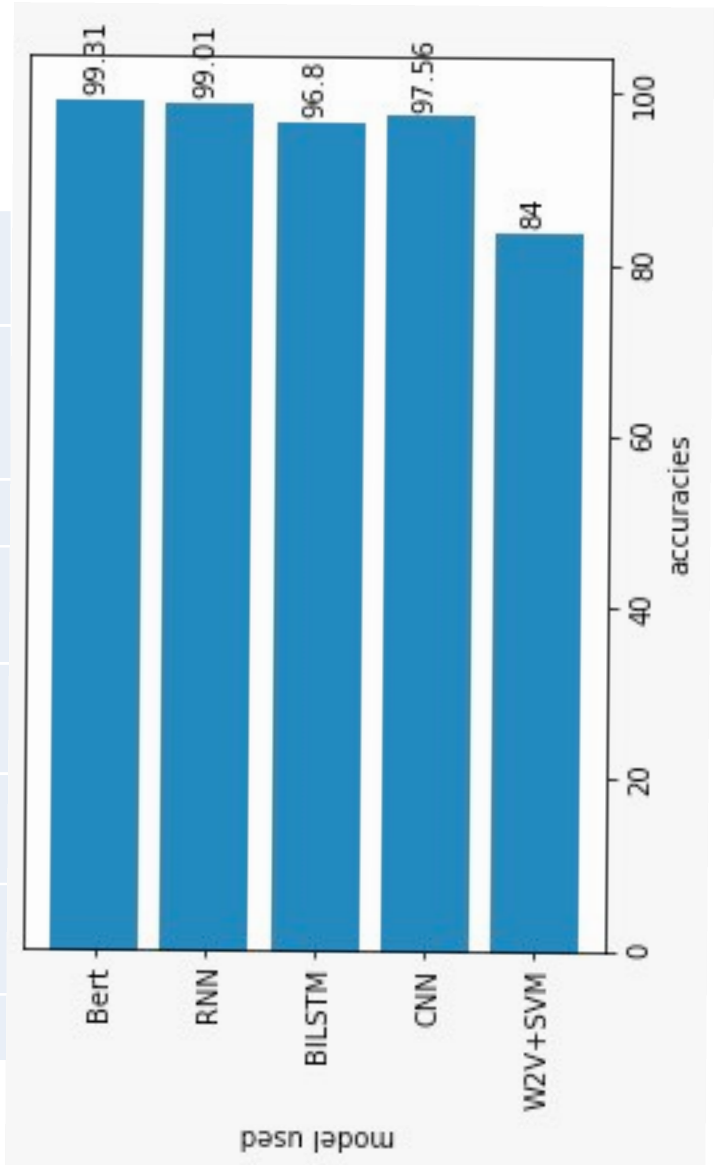
BERT Model no of epoch -1



Result

- Here we are comparing our top 2 baseline model with final model .in base line model Random forest with Tf-Idf with bigram was best with 76.20% accuracy and F1-score 78.88%
- In final model RNN with Word2Vec give accuracy 99.01% and F1-score 98.97%
- Bert is best out of all model and give 99.18% accuracy

Serial No.	Model	accuracy	F1 score
1.	Random forest classifier	76.20%	78.88%
2.	SVM	69.74%	71.08%
3.	Word2vec + SVM	84%	80.02%
4.	Word2vec + CNN	97.56%	97.43%
5.	Word2vec + BILSTM	96.80%	95.20%
6.	Word2vec + RNN	99.01%	98.97%
7.	**Bert**	99.31	



Analysis

- In base line we used Tf-idf but the problem is that Tf-idf take a sentence as a bag of words and there is no semantic meaning of words so now for final evaluation we used Word2vec .
- We also used Word2Vec + keras embedding for Neural Network model
- RNN is very good for capturing the sequential relationship and here in text writing follows the English grammer rule which itself is sequential in nature so RNN performs well on that.
- BILSTM is very good in capturing the relative positions of the word in a line in both the direction, so it save both forward and backword context of the word so it works well.
- The reason of using CNN is that it can detecet best feature without human itrecvation
- **CNN vs RNN-** As CNN don't remember their past context word and RNN is able to remember past important words and predicts next words according to past words. In our case both are outperforming.
- **RNN vs BILSTM** -in RNN only past inputs are remembered but in BILSTM inputs is processed both from forward and backward. Both model is outperforming in the data-set.
- **Why Bert over other N.N** -It returned different vectors for same words in different context that is it understand the context of the surrounding words and then return vector, while other embedding return same vector for same words.
- Eg — 1. They are busy with someone **Matter**
- 2. **Matter** are substances that has mass volume and space.
- Here both Matter has different context BERT will return different embedding for both matter while other word embedding like wor2Vec are failed to does so.