Rumour Detection

SemEval19

Group 11
Abhijeet Panda (2018201044) , Aniruddha Deshpande (20161058),
Darshan Kansagara (2018201033), Sarvat Ali (2018201009)

Abstract

The task of Rumour Detection here involves determining rumor veracity and the rumour stance taken for each reply in the discussion that follows. Our initial work focused on stance classification of tweets towards the truthfulness of rumors circulating in Twitter conversations in the context of breaking news.

TASK-A. Classifying the replies as to whether they Support, Deny, Query or Comment on the source tweet. This task was carried out as a part of our earlier deliverable. Our two models based on BERT and CNN achieved accuracy of around 72% and 77% respectively.

TASK-B. Verify the veracity (True, False, Unverified) of the source tweet. This was our primary focus for the final deliverable of the project. For this task we used the results obtained from TASK-A using which we created additional features to be used for the task of Veracity classification. We gave veracity of rumour in the form of a confidence score that lies between 0-1.

We formulated problem as multiclass classification problem hence We experimented with multiple models and features that are most optimal to come up with the best functional model in terms of evaluation metrics. By far we achieved maximum accuracy of 69% and a F1-score of 0.6.

1. Defining the Tasks

The first task involved Stance Classification of the responses to the source tweet which is the first subtask for this project. This involves the categorization of these responses into the following categories:

- **Support**: The author of the response supports the veracity of the rumor they are responding to.
- **Deny**: The author of the response denies the veracity of the rumor they are responding to.
- **Query**: The author of the response asks for additional evidence in relation to the veracity of the rumor they are responding to.

- **Comment**: The author of the response makes their own comment without a clear contribution to assessing the veracity of the rumor they are responding to.

The **dataset** used was provided as a part of **SemEval19** can be found here. This dataset involves both **Twitter** and **Reddit** conversations. We decided to currently focus only on Twitter conversations as the implementation can be easily extended over the Reddit conversations. This is so because both the conversations are structured in a similar tree-like format with source tweets having their veracity labels (ie. True/False), which are joined by an ensuing discussion in which further users support, deny, comment or query (SDCQ) the source text. To carry out this classification task we decided to use **BERT** and **CNN (with ELMo sentence embeddings)** based models. These architectures are explained below in Section 4.

The second task of Veracity classification simply involved classifying the source post into **"Rumour / Not a Rumour / Unverified"** classes. For this task we decided to go with several baseline and deep learning based models. This was done to achieve maximum possible accuracy scores over a dataset which is significantly small in size.

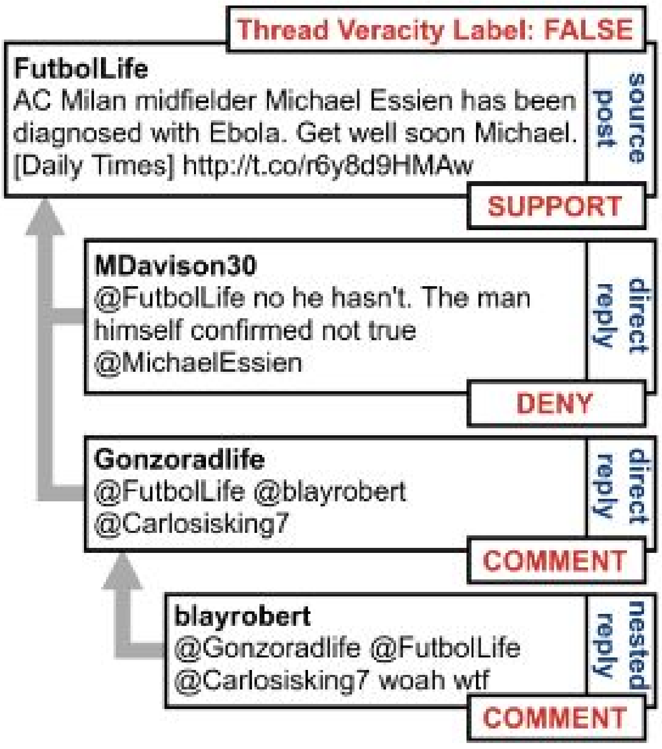
Thread Veracity Label: FALSE
FutbolLife
AC Milan midfielder Michael Essien has been diagnosed with Ebola. Get well soon Michael. [Daily Times] http://t.co/r6y8d9HMAw
source post
SUPPORT
MDavison30
@FutbolLife no he hasn't. The man himself confirmed not true @MichaelEssien
direct reply
DENY
Gonzoradlife
@FutbolLife @blayrobert @Carlosisking7
direct reply
COMMENT
blayrobert
@Gonzoradlife @FutbolLife @Carlosisking7 woah wtf
nested reply
COMMENT

Figure 1: An Example Tweet Thread with the Veracity label for the source tweet and SDQC labels for the respective replies.

2. Related work

In recent years, there were **two SemEval competitions** targeting the stance classification. The first one focused on the setting in which the **actual rumour was provided**. Organizers of SemEval-2016 Task 6 prepared a benchmarking system based on SVM using hand-made features and word embeddings from their previous system for sentiment analysis, outperforming all the challenge participants. The second competition was the previous RumourEval won by a system based on word vectors, handcrafted features2 and an LSTM (Hochreiter and Schmidhuber, 1997) summarizing information of the discussion's branches (Kochkina et al., 2017) [6]. Other submissions were either based on similar handcrafted features (Singh et al., 2017; Wang et al., 2017; Enayet and El-Beltagy, 2017) [8], features based on sets of words for determining language cues such as Belief or Denial (Bahuleyan and Vechtomova, 2017) [9], post-processing via rule-based heuristics after the feature-based classification (Srivastava et al., 2017) [10], Convolutional Neural Networks (CNNs) with rules or CNNs that jointly learnt word embeddings (Chen et al., 2017).

Some **End to End approaches** like the one developed by Augenstein et al. (2016) [11] encoded the target text by means of a bidirectional LSTM (BiLSTM), conditioned on the source text. The paper empirically shows that the conditioning on the source text really matters. Du et al. (2017) [12] propose target augmented embeddings – embeddings concatenated with an average of source text embeddings – and apply them to compute an attention based on the weighted sum of target embeddings, previously transformed via a BiLSTM. Mohtarami et al. (2018) [13] propose an architecture that encodes the source and the target text via an LSTM and a CNN separately and then uses a memory network together with a similarity matrix to capture the similarity between the source and the target text, and infers a fixed-size vector suitable for the stance prediction.

3. Methodologies and Architectures used.

3.1 Data Preprocessing

Following steps were carried out as a part of Data Preprocessing:

- Lowercase everything
- Remove user handles & URLs
- Transform hashtags into words

Please refer to the following figure that describes our overall workflow that summarises all of the methodologies we tried and tested for the task of Rumour Detection.

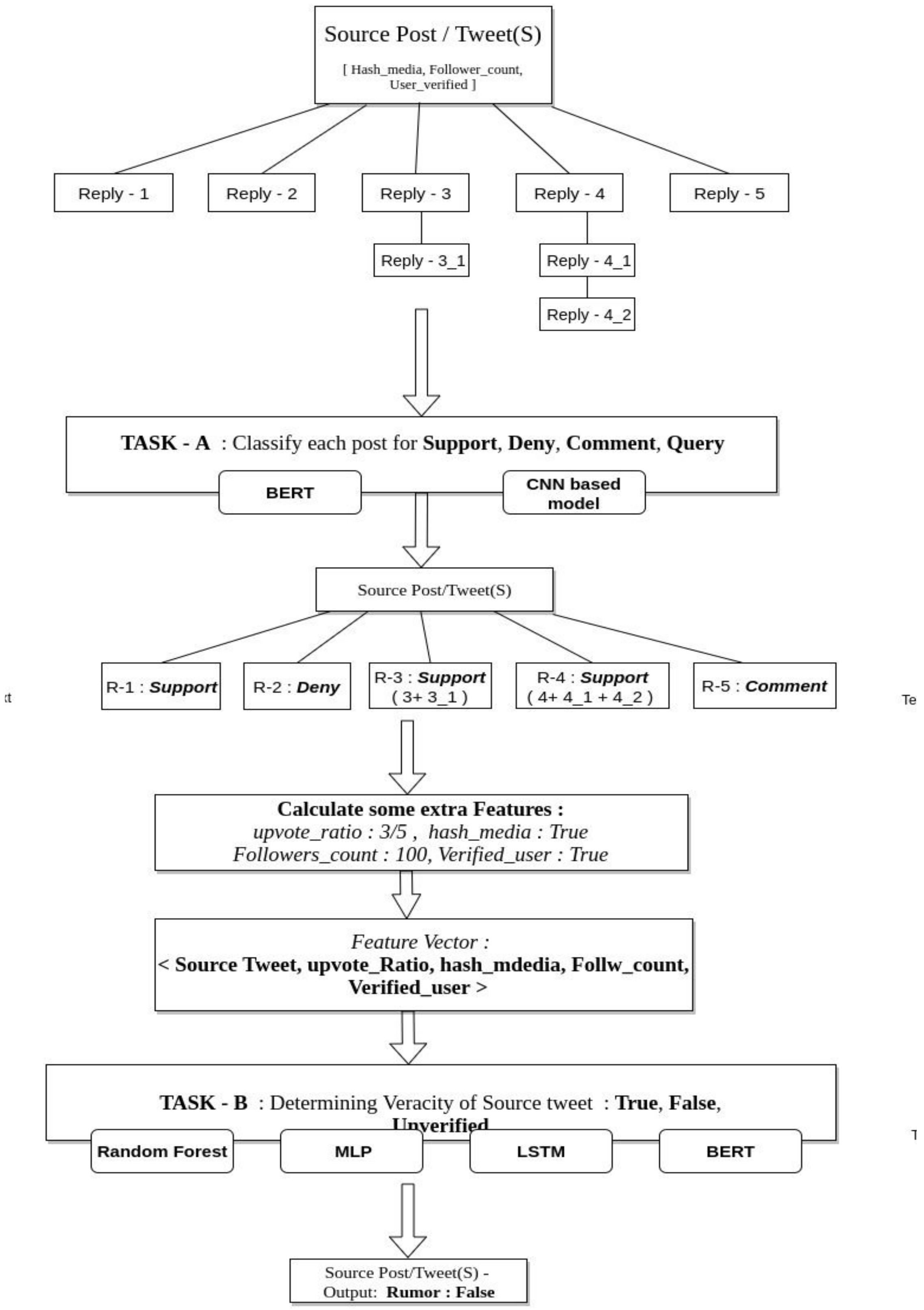
Source Post / Tweet(S)
[Hash_media, Follower_count, User_verified]
Reply - 1
Reply - 2
Reply - 3
Reply - 4
Reply - 5
Reply - 3_1
Reply - 4_1
Reply - 4_2
TASK - A : Classify each post for Support, Deny, Comment, Query
BERT
CNN based model
Source Post/Tweet(S)
R-1 : Support
R-2 : Deny
R-3 : Support (3+ 3_1)
R-4 : Support (4+ 4_1 + 4_2)
R-5 : Comment
Calculate some extra Features :
upvote_ratio : 3/5 , hash_media : True
Followers_count : 100, Verified_user : True
Feature Vector :
< Source Tweet, upvote_Ratio, hash_mdedia, Follw_count, Verified_user >
TASK - B : Determining Veracity of Source tweet : True, False, Unverified
Random Forest
MLP
LSTM
BERT
Source Post/Tweet(S) - Output: Rumor : False

Figure 2: Project Workflow

3.2 Our approaches with the Model's Architecture

TASK-A : Classifying the replies as to whether they Support, Deny, Query or Comment on the source tweet

3.2.1. Transformer based model :

We used transformer based model BERT (Bidirectional Encoder Representations from Transformers) bidirectional training of Transformer, a popular attention model, to language modeling. Transformer is an attention mechanism that learns contextual relations between words (or sub-words) in a text this approach allows the model to learn the context of a word based on all of its surroundings (left and right of the word). This already Pretrained model on large corpora such as Wikipedia dump and books corpus can be fine-tuned for our task of stance classification.

We then used this model for sentence pair-wise classification where each pair corresponds to the source text and its reply and output is whether the given reply supports, deny, query or just comment to given source tweet. We trained the model with various parameters and decided to go forward with the parameters mentioned below in Section 3.1.

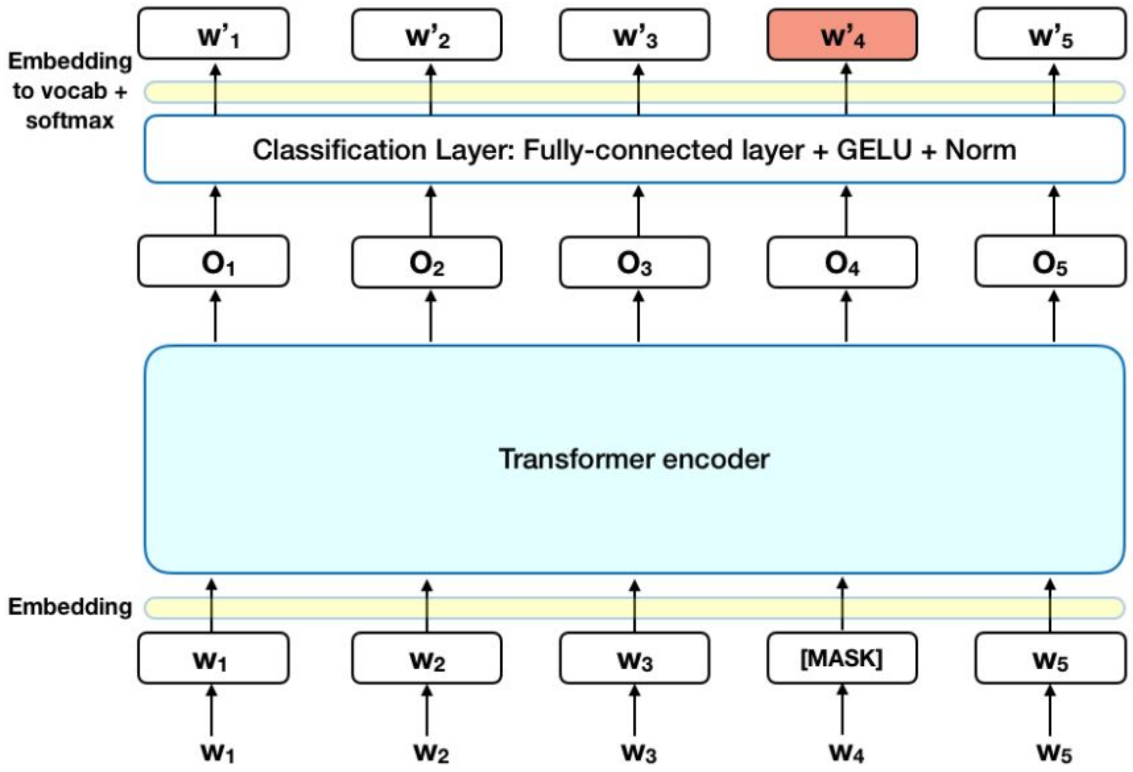
w'1
w'2
w'3
w'4
w'5
Embedding to vocab + softmax
Classification Layer: Fully-connected layer + GELU + Norm
O1
O2
O3
O4
O5
Transformer encoder
Embedding
w1
w2
w3
[MASK]
w5
w1
w2
w3
w4
w5

Figure 3: BERT Model Architecture

Following are the hyper parameter values we used for training our BERT based model.

Hyperparameter for BERT	Value
Training Batch Size	32
Learning Rate	5e-5
Number of Epochs	4.0
Warmup Proportion	0.1
Maximum Sequence Length	128
Board Size	19
Drop Out	0.1

Table 1: Hyper Parameters for BERT Based Model

3.2.2. CNN based model using ELMo embedding

We implemented a CNN-based neural architecture using Elmo embeddings of post text combined with its auxiliary features. Traditional embedding methods such as word2vec or GloVe work independently of the context and always map the same word to the same vector. In contrast, ELMo recent embedding approach is a bidirectional LSTM network that considers the context of the word, that the same word can have different meanings depending on its context. We represent each text to ELMo embedding, Next, the embedded text is fed into many convolutional layers. Each convolution operation is batch normalized after a ReLU activation. Please refer to Section 3.2 for the hyperparameter values used during training.

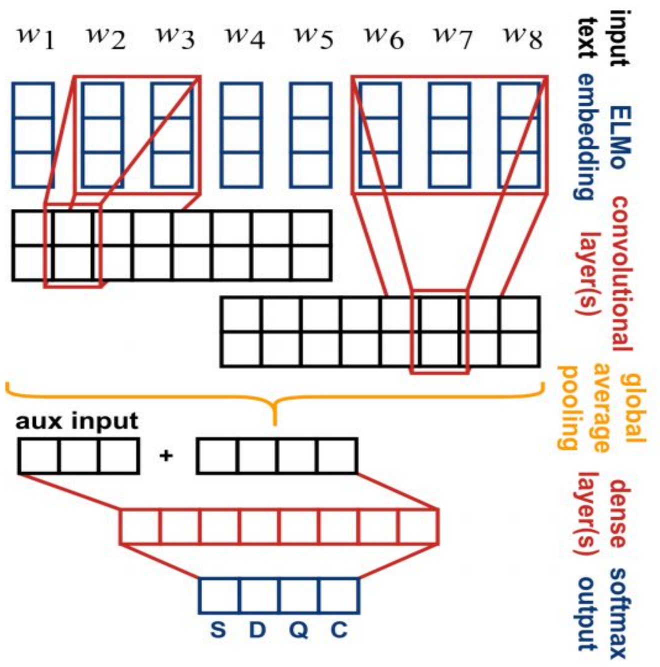

Figure 4: CNN Based architecture with ELMo sentence embeddings.

Following are the hyper parameter values we used for training our CNN Based model.

Hyperparameter for CNN based Model	**Value**
Maximum Sentence Length	32
Batch Size	512
Number of Epochs	100
Learning Rate	1e-3
Weight Decay	1e-2
Class Weights	[1,1,1,0.2] for [S,D,Q,C] respectively
Number of Convolutional Layers	1
Kernel Sizes	[2,3]
Number of Channels	64
Number of Dense Layers	3
Number of Hidden Dense Layers	128
Dense Dropout	0.5

Table 2: Hyper Parameters for CNN Based Model

Please refer to the following section for the scores achieved using the above-used models with given architecture for Subtask A.

TASK - B: Verify the veracity(true, false, unverified) of the source tweet

We experimented with several features provided as a part of the dataset but we found the following features most optimal for training the model:

1. **Upvote Ratio:** We used the results from Task-A to gather how many replies and their nested replies that support the corresponding source tweet to gather upvote ratio parameter. This is calculated as ratio of number of replies that support to source tweet to total replies for source tweet(support,deny,comment,query)
2. **Availability of Media:** Whether the source post has an attached Image or a URL to it.
3. **Is Verified:** Whether the user is verified by social media platform.
4. **Number of Followers:** Number of followers of the author of the source tweet.

3.2.3. Models implemented for Task - B

After finalising the optimal features set as mentioned above we tried classifier models to classify rumour As true,false or unverified in terms of confidence score.

We tried many **baseline** models some performed well, others didn't. Some baseline models we are mentioning here are **Decision Tree, Random forest classifier, Gradient boosting** .some performed well others gave very little accuracy.our baseline models achieves results as given.

A. Decision Tree : Decision Tree Classifier, repetitively divides the working area(plot) into sub part by identifying lines. To determine how well a test condition performs, we need to compare the degree of impurity of the parent before splitting with degree of the impurity of the child nodes after splitting. The larger the difference, the better the test condition. The measurement of node impurity/purity we used is **Information Gain** as parameter.

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

B. Random Forest : It consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction.

- Ideal behind using this model is s large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.
- We got better results than previously single Decision tree. It was satisfactory than earlier.

C. Gradient Boosting : This technique employs the logic in which the subsequent predictors learn from the mistakes of the previous predictors.Gradient boosting combines weak "learners" into a single strong learner in an iterative fashion. It was the best baseline we implemented.

D. RNN (LSTM) : In this technique The embedding layer encodes the input sequence into a sequence of dense vectors of dimension embed_dim and The LSTM transforms the vector sequence into a single vector of size lstm_out, containing information about the entire sequence. We used k=5 for K-fold cross validation as training data is very less to train the model.

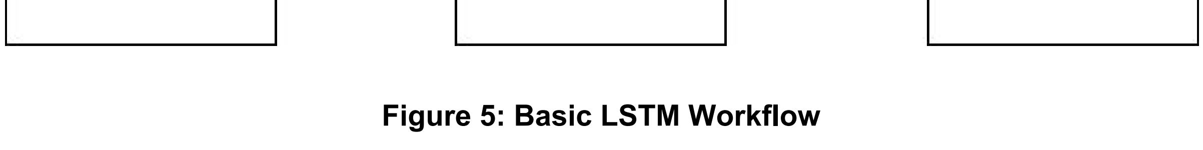

Figure 5: Basic LSTM Workflow

Following are the Hyperparameters we used for training our RNN based Classifier.

Hyperparameter for RNN based Model	**Value**
Maximum Sentence Length	200
Batch Size	32
Number of Epochs	10
Number of Layers	3
Number of Dense Layers	2
Dense Dropout	0.2
Activation	Softmax
Loss	Categorical_Cross_Entropy
Optimiser	Adam

Table 3: Hyper Parameters for RNN Based Model

E. BERT - This model has already been explained for Task-A.It performs well when training data is less.As an extension for Task B we modified input to this model to include our Task-A results that is upvote ratio.The model tried to learn using parameters as mentioned below

Following are the Hyperparameters we used for training our RNN based Classifier

Hyperparameter for BERT	**Value**
Training Batch Size	32
Learning Rate	4e-5
Number of Epochs	10
Warmup Proportion	0.2
Maximum Sequence Length	128
Board Size	19
Drop Out	0.5

Table 4: Hyper Parameters for BERT Based Model

4.Evaluation Mechanism & Results

We evaluated model on metrics of Accuracy, F1-score, Recall and Precision and Accuracy. We used k-fold cross validation, upsampling and tuned model with many different parameters to come up with most appropriate results.

- **Accuracy:** It indicates how many instances are classified correctly.
    - Accuracy :- Ratio of number of True Positives to total number of examples.
- **Precision:** The ability of classification model to identify only relevant data points.
    - Precision :- Ratio of number of True Positives to number of (True Positives and False Positives).
- **Recall:** Ability of model to identify only relevant data.
    - Recall :- Ratio of number of True Positives to number of (True Positives and False Negative)
- **F1-Score:** Harmonic Mean of Precision and Recall
    - F1-Score :- Ratio of 2 * Precision * Recall to (Precision + Recall)

4.1 Task - A Results

Transformer based model (BERT) :

Class	**Precision**	**Recall**	**F1-score**
Comment	0.78	0.87	0.83
Deny	0.38	0.20	0.26
Query	0.49	0.60	0.54
Support	0.50	0.29	0.37

Task A: SDQC overall Result : Accuracy: 0.72, **F1-Score:** 0.54

CNN based Model :

Class	Precision (in %)	Recall (in %)	F1-score (in %)
Comment	86.2	89.3	87.6
Deny	15.9	12.1	12.7
Query	56.9	33.5	41.7
Support	40.8	37.2	36.9

Task A: SDQC overall Result : Accuracy: .077, **F1-Score:** 0.448

4.2 Task - B Results

Model	Accuracy	F1-score
Random Forest	0.65	0.57
Decision Tree	0.58	0.52
Gradient Boosted Trees	0.6	0.52
LSTM	0.52	0.43
BERT	0.66	0.65

Task B: Overview of the results.

1. Random Forest classifier

Class	Precision	Recall	F1-score
False	**0.27**	**0.38**	**0.32**
True	**0.69**	**0.77**	**0.73**
Un-verified	**0.79**	**0.58**	**0.67**

Accuracy: 0.65 , **F1-score:** 0.57

2. Decision Tree

Class	Precision	Recall	F1-Score
False	**0.23**	**0.38**	**0.29**
True	**0.69**	**0.58**	**0.63**
Un_verified	**0.65**	**0.65**	**0.65**

Accuracy: 0.58**, F1-score:** 0.52

3. Gradient Boosted

Class	Precision	Recall	F1-Score
False	**0.17**	**0.25**	**0.2**
True	**0.64**	**0.68**	**0.66**
Un_verified	**0.8**	**0.62**	**0.7**

Accuracy: 0.6 **, F1-score:** 0.52

4. RNN based model (LSTM)

Accuracy: 0.53

5. BERT

Class	Precision	Recall	F1-Score
False	**0.57**	**0.67**	**0.62**
True	**0.74**	**0.65**	**0.69**
Un_verified	**0.62**	**0.68**	**0.65**

Accuracy: 0.66, **F1-score:** 0.65

5. Analysis

The final Bert based model achieved the macro F1 score of 0.62, improving all our baseline models. A detailed analysis of the provided data shows that the employed information sources are not sufficient to correctly classify some examples (tweets not having much replies). External features like wikipedia data dump can be Included to improve the results further. We tried to include many extra features but most of them did not improve the results significantly so relevance of all features vary with different models.

6.Code link to the baseline implementations

6.1 Reference for code section of Github Repository

https://github.com/darshank15/IRE-major-Project---SemEval-Rumour-Detection

7. Link to the webpage of project and youtube video

7.1 WebPage of our Project & Youtube Videos.

https://darshank15.github.io/IRE-major-Project---SemEval-Rumour-Detection/

References

[1] Rani Horev - BERT Explained: State of the art language model for NLP
[2] Genevieve Gorrell, Kalina Bontcheva, Leon Derczynski, Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga - RumourEval 2019: Determining Rumour Veracity and Support for Rumours
[3] Thilina Rajapakse - A Simple Guide On Using BERT for Binary Text Classification
[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova - BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
[5] Martin Fajcik, Lukáš Burget, Pavel Smrz - SemEval-2019 Task 7: Determining the Rumour Stance with Pre-Trained Deep Bidirectional Transformers
[6] Elena Kochkina, Maria Liakata, Isabelle Augenstein - Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification with Branch-LSTM
[7] Ruoyao Yang, Wanying Xie, Chunhua Liu, Dong Yu - BLCU NLP at SemEval-2019 Task 7: An Inference Chain-based GPT Model for Rumour Evaluation
[8] Vikram Singh, Sunny Narayan, Md Shad Akhtar, Asif Ekbal, Pushpak Bhattacharyya - IITP at SemEval-2017 Task 8 : A Supervised Approach for Rumour Evaluation
[9] Hareesh Bahuleyan and Olga Vechtomova - UWaterloo at SemEval-2017 Task 8: Detecting Stance towards Rumours with Topic Independent Features
[10] Ankit Kumar Srivastava et. al. - DFKI-DKT at SemEval-2017 Task 8: Rumour Detection and Classification using Cascading Heuristics
[11] Isabelle Augenstein and Tim Rocktaschel - Stance Detection with Bidirectional Conditional Encoding
[12] Jiachen Du, Ruifeng Xu, Yulan He, Lin Gui - Stance Classification with Target-Specific Neural Attention Networks
[13] Mitra Mohtarami, Ramy Baly, James Glass - Automatic Stance Detection Using End-to-End Memory Networks