Detection and Resolution of Rumours in Social Media: A Survey

P Saran Pandian
DAIICT
Gandhinagar
202018004@daiict.ac.in

Ravi Satvik
DAIICT
Gandhinagar
202018008@daiict.ac.in

Akash Gupta
DAIICT
Gandhinagar
2020180015@daiict.ac.in

Bhavya Jain
DAIICT
Gandhinagar
202018023@daiict.ac.in

Sharvari Gokhale
DAIICT
Gandhinagar
202018038@daiict.ac.in

***Abstract*—This project addresses the challenge of rumor stance classification, which involves identifying the stances of users towards the veracity of a rumor. Stance classification is considered to be an important step towards rumor verification, therefore performing well in this task is expected to be useful in debunking false rumors. In this work we classify a set of Twitter posts discussing rumors into either supporting, denying, questioning, or commenting on the underlying rumors. We propose an LSTM-based sequential model which takes in the sequences of source and their corresponding reply tweets as input and gives stances for all the underlying tweets.**

Keywords—LSTM, Classifier, Rumour Verification, SDQC, RumourEval, Word2Vec

I. INTRODUCTION

To detect the veracity of a tweet, one of the needed features is of stances taken towards that tweet by the user. So, there is a need for implementing a stance classifier to fit it into the rumour veracity classifier. Here, we have implemented the state of the art model for rumour stance classifier which uses an LSTM-based sequential model. We have included features in addition to the word embeddings of each tweet. The goal is to classify each of the tweets in the conversation thread as either supporting, denying, querying, or commenting (SDQC) on the rumour initiated by the source tweet.

II. DATASET

The research presented in this paper is a submission to SemEval-2017, Task 8 (RumourEval: Determining rumour veracity and support for rumours), Subtask A (SDQC) (Derczynski et al., 2017). The objective of this subtask is to classify the relation between a tweet and the rumour it is related to in terms of support, deny, query or comment. Data for the task is available in the form of online discussion threads, each pertaining to a particular event and the rumours around it. These threads form a tree, where each tweet has a parent tweet it responds to. Together these form a conversation, initiated by a source tweet. The data has already been annotated for veracity and SDQC following a published annotation scheme (Zubiaga et al., 2016b), as part of Maintaining the Integrity of the Specifications

Our training dataset comprises 272 rumourous threads collected for 8 events in total, which include 272 source tweets and 3694 reply tweets, amounting to 3966 tweets in total. These events include well-known breaking news such as the Charlie Hebdo shooting in Paris, the Ferguson unrest in the US, and the Germanwings plane crash in the French Alps This dataset is already publicly available (Zubiaga et al., 2016a) and constitutes the training data.

For the test data, we annotated 25 additional threads. These include 25 threads extracted from the same events as the training set. The test dataset includes, in total, 256 tweets, 25 of which are source tweets.

III. FEATURES

For performing any feature generating techniques, it is required that all the tweets be transformed to a suitable form. For that purpose, we apply certain preprocessing steps in which we aim to remove any characters which aren't alphabetic in nature and then convert all the words to lowercase, tokenize the tweets and remove any stop words. We further perform stemming and lemmatization. After completion of this necessary preprocessing, the obtained data is used for extracting the features and are stored in embeddings specific to the features, such as:

Word Vectors: We have used a pre-trained model by Google word2vec in the genism package. The tweet corpus was given as an input to Word2Vec with output as word vectors.

Tweet Lexicon: A dictionary for negation words and swear words consisting of 19 and 450 words respectively, was taken. The presence of any of the words from both dictionaries determines the count of the negation words and swear words in the tweet. The counts were then stored in an embedding.

Punctuation: Punctuations govern the manner in which a sentence is perceived. The tweets could represent a question, a fact, etc. For this purpose, we check for determinant punctuations such as a period(.), an exclamation mark(!), a question mark(?), and the ratio of uppercase letters to lowercase letters in the tweet.

Attachments: We have considered URLs and images as the other entities, apart from the text in the tweet. The presence of an image outputs the label as 1 and its absence outputs label '0'. The same is the case of URL. These labels then form a separate embedding.

Relation between tweets: The measure of the relation of one tweet to another is established with the help of Word2Vec cosine similarity. Nature of tweets considered for this was Source tweet to Reply tweet, Tweet and its Preceding tweet.

Content-Length: Count of the total number of words and the total number of characters is calculated

Tweet: A tweet can be a source tweet or a reply tweet to some source. We have taken source tweets as criteria to be stored. In total, we have 313 features for each tweet.

IV. TRAINING THE LSTM MODEL

A. LSTM

An LSTM has a similar control flow as a recurrent neural network. It processes data passing on information as it propagates forward. The differences are the operations within the LSTM's cells.

The core concepts of LSTMs are the cell state, and its various gates. The cell state act as a transport highway that transfers relative information all the way down the sequence chain.

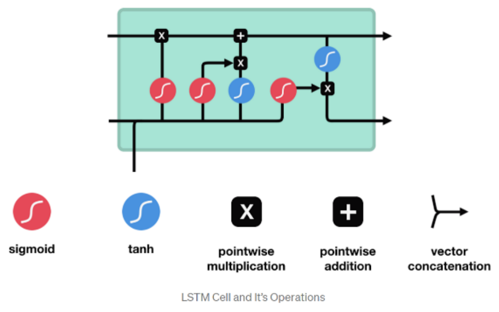

Fig 1 : LSTM Model

As the cell state goes on its journey, information gets added or removed to the cell state via gates. The gates are different neural networks that decide which information is allowed on the cell state. The gates can learn what information is relevant to keep or forget during training.

Gates contain sigmoid activations. A sigmoid activation is similar to the tanh activation. Instead of squishing values between -1 and 1, it squishes values between 0 and 1. That is helpful to update or forget data because any number getting multiplied by 0 is 0, causing values to disappears or be "forgotten." Any number multiplied by 1 is the same value therefore that value stays the same or is "kept." The network can learn which data is

not important therefore can be forgotten or which data is important to keep.
So we have three different gates that regulate information flow in an LSTM cell. A forget gate, input gate, and output gate.

Forget gate decides what information should be thrown away or kept. To update the cell state, we have the input gate. First, we pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 indicate not important, and 1 indicates important. We also passed the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network. Then we multiplied the tanh output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output.

Then we take the output from the input gate and do a pointwise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new cell state.

The output gate decides what the next hidden state should be. Remember that the hidden state contains information on previous inputs. The hidden state is also used for predictions.

B. TRAINING THE MODEL USING PYTORCH

The model was trained using the PyTorch library. The input consists of embeddings of source tweets and their corresponding reply tweets. Since source tweets have a different number of reply tweets, the input consists of different sequence lengths. To deal with such input, padding was used. Dataloader was used to do batching. After trial and error the following hyperparameters were used:

Batch Size	32
Hidden Size	500
Learning Rate	0.1
Epochs	100

The optimiser used, was stochastic gradient descent (SGD) and the loss function used was cross entropy. The input dimension for LSTM layer is 313 and hidden dimension is 500. This 500 dimension is further reduced 4 class output which is sent through a softmax layer. The loss is calculated between the softmax output and actual ground truth. The loss is optimised using the SGD optimiser.

V. EVALUATION

Since this is a classification problem, we used accuracy and F1 macro averaged score to evaluate.

Label	C	D	S	Q
Commenting	96	39	36	2
Denying	5	2	3	1
Supporting	26	12	24	7
Querying	21	3	4	0

Table 1. Confusion Matrix

Measure	**Value**
Accuracy	**0.43416370106761565**
F1 Macro	**0.2526933775323492**

Table 2. Results

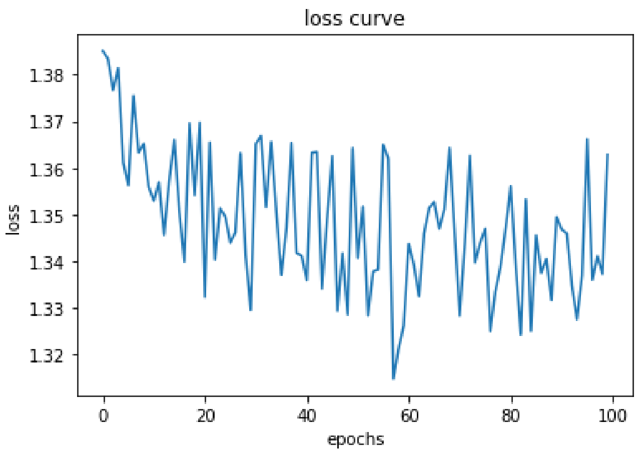

Fig 2 : Loss Curve

VI. CONCLUSION

We tried different hyperparameters but we were not able to achieve better accuracy and F1 score. There is a high imbalance in the dataset as most of

the stances taken by users were commenting. So it was difficult to build a model to accurately predict the output. The loss function did not converge properly as shown in figure(Fig 2).
Also, proper feature engineering techniques were not mentioned in the reference paper only features to be included were mentioned.

VII. FUTURE SCOPE

We can try different feature engineering techniques on the input data like including sentiment data using textblob, including the emoji embedding and correcting the spellings in the tweet, and try to increase the accuracy and f1 score.

VIII. INDIVIDUAL CONTRIBUTION:

P.Saran Pandian 202018004 – LSTM MODEL
Ravi Satvik Gorthi 202018008 – LSTM MODEL
Akash Gupta 202018015– Dataset collection and explanation
Bhavya Jain 202018023 – Feature Extraction
Sharvari Gokhale 202018038 – Feature Extraction.

IX. REFERENCES

[1] Kochkina et. Al, Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification with Branch-LSTM.

[2] Derczynski et. Al, SemEval-2017 Task 8: RumourEval: Determining rumour veracity and support for rumours.

[3] https://colah.github.io/posts/2015-08-Understanding-LSTMs/K. Elissa, “Title of paper if known,” unpublished.