Stance Detection And Rumor Verification

Thelma Melissa Gomes
50416894
Department of Computer Science
University at Buffalo
thelmame@buffalo.edu

Shreya Mukherjee
50419574
Department of Computer Science
University at Buffalo
mukherj4@buffalo.edu

Shashank Desai
50388502
Department of Engineering Science
University at Buffalo
desaisha@buffalo.edu

Abstract

Identifying rumors, specially in today's day where social media like Twitter and Reddit lead to the fast spread of information, is a challenging task due to the absence of sufficient information about the news circulating on social media. It thus, becomes very hard to judge whether a particular news article is true or not. In this paper, we approach this problem by implementing 3 separate sub-tasks; rumor stance prediction, rumor verification and stance text generation.

1 Introduction

In this time and era, Social media has emerged as a powerful platform among the masses. Information and news travel faster through this medium than any traditional means such as News channels or the good old radio. This is greatly causing the problem of Information overload, giving rise to rapid spread of fake news and rumors. The increase in this unwanted overload calls for the need of timely rumor verification. Twitter is one such huge platform, and in order to validate huge amount of data here, it is important that we automate the verification process.

Multiple research studies have demonstrated the potential impact of false claims on important political outcomes. Living in a "post-truth world", in which perceived truth can matter more than actual truth (Dale, 2017), the dangers posed by unchecked social media platforms, as well as lack of critical thinking by many readers to discern credible information, are evident. Platforms are increasingly motivated to engage with the problem of damaging content that appears on them, as society moves toward a consensus regarding their level of responsibility. Most of the effort so far is manual, and struggles to keep up with the large volume of online material.

Within NLP research, the tasks of stance classification of news articles and social media posts and the creation of systems to automatically identify false content are gaining momentum. In such a scenario, studying the conversation between users discussing the event on Twitter can possibly give insights about the veracity of a circulating rumour story. Similarly, the source tweet itself, could give a lot of data about its own veracity. Depending on the language and depth of information in a tweet, we could infer knowledge about whether a tweet is true or false. Lastly, we use tweet threads to generate some stance text based on the stance of the replies within that source tweet.

The rest of the paper is organized as follows: Section 2 mentions some of the existing work in this field. Section 3 is a brief overview of the tasks. Section 4, 5 and 6 detail each of sub-tasks including the architectures implemented and the results obtained. We have also given a qualitative analysis of the implementations and areas of improvement for the same.

2 Related Work

In the scope of our project, we referenced the research paper presented in the RumourEval 2019 competition, as our dataset of Tweets was sourced from theirs. This research paper helped us understand the proceedings of Subtasks A B and their approach for its multi-class classification. For implementing improved models, we studied the architecture of the various models we used through useful blogs. For Subtask C, we didn't find any research work pertaining to reply tweet generation, but we studied the various architectures that can be used to either summarize required data or generate new text through some helpful articles in order to determine the most efficient method.

3 Problem Description

The main purpose of our project is to detect a particular stance for the tweets, verify whether it is a rumor and generate a stance text which supports our model output.

Below are the sub-tasks which will be implemented in phases:

1. **SDQC Support Classification:** The reply tweets in the conversation thread of each source tweet will undergo multi-class SDQC (Support, Deny, Query, Comment) classification.
2. **Veracity Prediction:** Multi-class text classification will be implemented on source tweets for them to be determined as True, False or Unverified. Based on the output class, we will obtain a confidence score ranging from 0 to 1 - where a score of 0 indicates an unverified rumor.
3. **Stance Generation:** Our model will generate a stance text using the information we obtained from the above sub-tasks. The source tweet, its reply thread and label assigned to each reply as per sub-task A will be used to generate the stance response using conditional distribution.

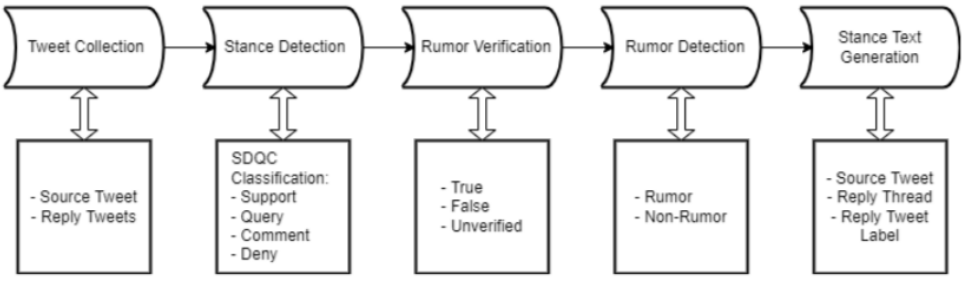

Figure 1: Project Pipeline

4 Sub-task A: SDQC support classification

4.1 Objective

Given a source tweet, tweets in a conversation thread discussing the claim are classified as either supporting, denying, querying or commenting on the rumor mentioned by the source tweet.

4.2 Architecture Implementations

Our input training/ testing data contains tweet id, reply tweet and class label belonging to either support, deny, query and comment. The source tweet may contain mention, any additional URLs or some unwanted punctuation which have been cleaned accordingly in our preprocessing stage. We followed the below data preprocessing steps:

1. Removing of slang words, punctuation and URLs
2. Converting the text to lowercase
3. Lemmatizing the text
4. Converting the textual target labels into numeric using a one-hot encoder.

4.2.1 Approach 1: TF-IDF Word Embeddings with SVM and Naive Bayes

In this approach we are converting the target variable "label" which is one of support,deny,query,comment.Here we are converting the categorical textual values into numeric values.Inorder to convert the input textual into numeric vector we are using TF-IDF count vectorizer.This approach will help us in converting the textual field into numeric vector which takes each word importance into consideration.We have used classical ML models pertaining to NLP tasks such as MultiNomial Naive Bayes and Linear SVM.

4.2.2 Approach 2: Word2Vec Embeddings with SVM

The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence.In our tasks we want to experiment with one of the state of the art word embedding word2vec.Here we have used pre-trained word2vec to apply on our input textual data and convert the same into vector representations.This approach will help us in convert our textual data into more meaningful numeric vectors.We fed the word2vec trained word embeddings to classical ML model support vector machine.

4.2.3 Approach 3:

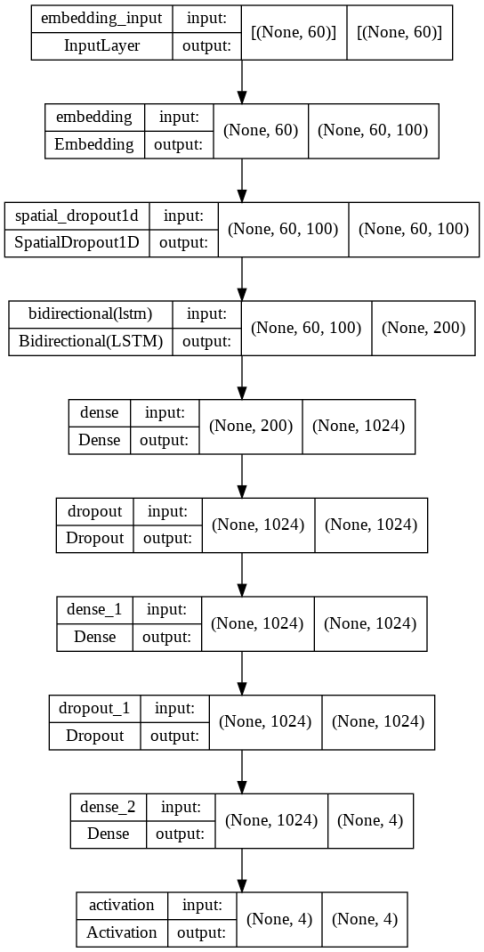

Figure 2: Subtask A BiLSTM Architecture

In this approach we have used GLOVE word embedding.GloVe stands for Global Vectors for word representation. It is an unsupervised learning algorithm developed by researchers at Stanford University aiming to generate word embeddings by aggregating global word co-occurrence matrices from a given corpus. The basic idea behind the GloVe word embedding is to derive the relationship between the words

from statistics. Unlike the occurrence matrix, the co-occurrence matrix tells you how often a particular word pair occurs together. Each value in the co-occurrence matrix represents a pair of words occurring together. In this approach we wanted to experiment with the SOTA word embedding Glove word embeddings as this learn from meaning of word in the context which perform better than word2vec embedding for sentiment analysis tasks. We have choosen to use 300 dimensional word embedding vector. For the model we have used BiLSTM which again is the SOTA deep learning model for the sentiment analysis tasks. A Bidirectional LSTM, or biLSTM, is a sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction. BiLSTMs effectively increase the amount of information available to the network, improving the context available to the algorithm (e.g. knowing what words immediately follow and precede a word in a sentence). The glove word embeddings is then converted to relevant embedding matrix such that it is passed to BiLSTM network.The network being bidirectional will help us in overcoming the vanishing gradients thus helping us in memorizing the important words which are useful for later.The LSTM network with forget gates will help us in throwing the information which no longer useful. Then the output of the BiLSTM model is fed into Feed Forward Neural Networks consisting of two dense layers with relu activation function.The ouput is then transformed into concerned probabilities by softmax activation function.We have added the necessary dropout values to the architecture in order to improve generalization hence reducing both over fitting and under fitting.

4.2.4 Approach 4:

In this approach we have used the SOTA BERT word embedding which takes into account contextual representation of the word in the sentence. BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection. (In NLP, this process is called attention.) In our architecture we have BERT pre-trained model "bert-base-cased".Here as part of the architecture we have transformed our input text with the max length vector of 70 and padding equal to the max length and used the adam optimizer.

4.3 Subtask A: Results

The Subtask A result score comparisons are provided below:

Parameters	Approach 1	Approach 2	Approach 3	Approach 4
F1	0.2564	0.2164	0.2254	0.2164

Table 1: Subtask A Results

4.4 Subtask A: Qualitative Analysis

As we can observe from the above table we can observe that simple approaches are giving better results.Given the scarcity of the data the deep learning algorithms are not able capture the pattern properly.Also the other deep learning architectures such as RNN and BiRNN were also giving results in a similar manner.Also may be usuage of more input variables either by feature extraction or other information regarding the user of the tweet might have helped in overcoming the lower scores.Also fine tuning the hyper parameters such as max length and number of epochs might have got out job done.

5 Subtask B - Veracity Prediction

5.1 Objective

The goal of subtask B was to predict the veracity of a given rumor. The rumor introduced by the source tweet that spawned the discussion is classified as true, false or unverified.

5.2 Approach 1: TF-IDF Word Embeddings with SVM and Naive Bayes

In this approach we are converting the target variable "label" which is one of support,deny,query,comment.Here we are converting the categorical textual values into numeric val-

ues.Inorder to convert the input textual into numeric vector we are using TF-IDF count vectorizer.This approach will help us in converting the textual field into numeric vector which takes each word importance into consideration.We have used classical ML models pertaining to NLP tasks such as MultiNomial Naive Bayes and Linear SVM.

5.3 Approach 2: Word2Vec Embeddings with SVM

The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence.In our tasks we want to experiment with one of the state of the art word embedding word2vec.Here we have used pre-trained word2vec to apply on our input textual data and convert the same into vector representations.This approach will help us in convert our textual data into more meaningful numeric vectors.We fed the word2vec trained word embeddings to classical ML model support vector machine.

5.3.1 Approach 3:

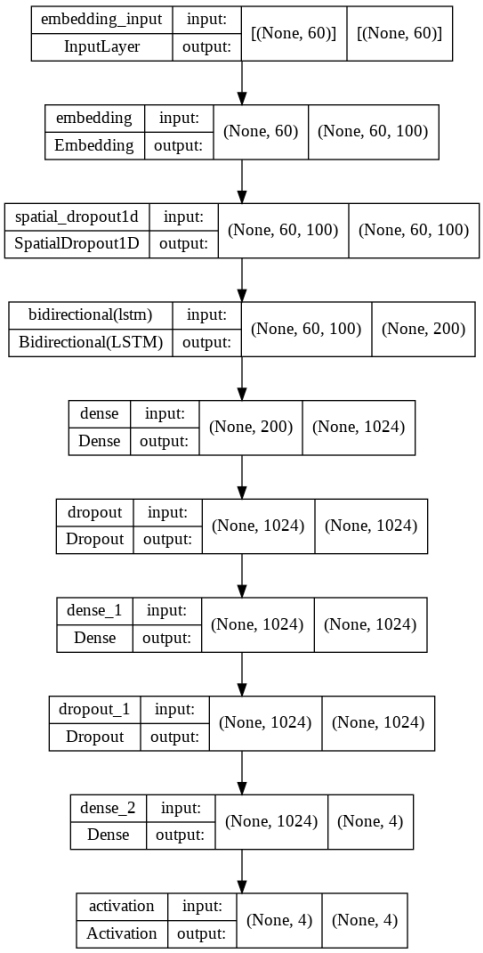

Figure 3: Subtask A BiLSTM Architecture

In this approach we have used GLOVE word embedding.GloVe stands for Global Vectors for word representation. It is an unsupervised learning algorithm developed by researchers at Stanford University aiming to generate word embeddings by aggregating global word co-occurrence matrices from a given corpus. The basic idea behind the GloVe word embedding is to derive the relationship between the words from statistics. Unlike the occurrence matrix, the co-occurrence matrix tells you how often a particular word pair occurs together. Each value in the co-occurrence matrix represents a pair of words occurring together. In this approach we wanted to experiment with the SOTA word embedding Glove word embed-

dings as this learn from meaning of word in the context which perform better than word2vec embedding for sentiment analysis tasks. We have choosen to use 300 dimensional word embedding vector. For the model we have used BiLSTM which again is the SOTA deep learning model for the sentiment analysis tasks. A Bidirectional LSTM, or biLSTM, is a sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction. BiLSTMs effectively increase the amount of information available to the network, improving the context available to the algorithm (e.g. knowing what words immediately follow and precede a word in a sentence). The glove word embeddings is then converted to relevant embedding matrix such that it is passed to BiLSTM network.The network being bidirectional will help us in overcoming the vanishing gradients thus helping us in memorizing the important words which are useful for later.The LSTM network with forget gates will help us in throwing the information which no longer useful. Then the output of the BiLSTM model is fed into Feed Forward Neural Networks consisting of two dense layers with relu activation function.The ouput is then transformed into concerned probabilities by softmax activation function.We have added the necessary dropout values to the architecture in order to improve generalization hence reducing both over fitting and under fitting.

5.4 Approach 4:

In this approach we have used the SOTA BERT word embedding which takes into account contextual representation of the word in the sentence. BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection. (In NLP, this process is called attention.) In our architecture we have BERT pre-trained model "bert-base-cased".Here as part of the architecture we have transformed our input text with the max length vector of 70 and padding equal to the max length and used the adam optimizer.

5.4.1 Subtask B: Results

The Subtask B result score comparisons are provided below:

Parameters	Approach 1	Approach 2	Approach 3	Approach 4
Root Mean Square Error	0.4983	0.4731	0.5666	0.1818
F1	0.285682	0.2966	0.1985	0.1888

Table 2: Subtask B Scores

5.4.2 Subtask B: Qualitative Analysis

As we can observe from the above table approaches 1 and 2 are giving the better results.The reason might be as the other two approaches are deep learning approaches which in general requires a lot of data but here in our task the dataset is very small the neural network approaches did not perform well.Hence for our task simple approaches and algorithms are able to correctly identify the pattern hence the results.Moreover We feel that apart from source text we were not able to utilize the other releavant information pertaining to the relevancy of the source tweet.Building new features and using those might have improved our results.Also may be use of other algorithm may have helped us in achieving our tasks more efficiently.

6 Subtask C - Stance Generation

6.1 Objective

The goal of this task is to train a language model to generate a stance text(R) given a source tweet(S), tweets in a conversation thread discussing the claim (CTX) and the stance prediction label (L) from Subtask A, where the conditional distribution of the stance text(R) is p(R—S,CTX,L).

Similar to Subtask A and B, we had preprocessing for Subtask C. Since we have an all tweet-text-based input data, we need to do sentence based preprocessing like removing stop-words and punctuation

as well as spellcheck. We also did some tweet based preprocessing including removal of URLs and mentions in the tweet to generate coherent data. For training our models in Subtask C, we divided the tweet corpus into 4 different categories based on the SDQC classification.

6.2 Approach 1: Seq2Seq Model

We used a Seq2Seq architecture as our baseline model. Seq2Seq is a type of Encoder-Decoder model using RNN. It can be used as a model for machine interaction and machine translation. In our model, we have used this for reply tweet generation. For training our Seq2Seq model, we have 4 different datasets with source tweet-reply tweet pairs based by SDQC classification. We bring each reply tweet in the dataset to same length by padding them with special tags denoting the beginning ⟨START⟩ and end ⟨END⟩. The main procedure is encoding the sentences into fixed length vectors. We used the encoder–decoder LSTM model for this aim. We generate the vocabulary of the input text using Tokenizer. We then change this Bag of Words to Bag of IDs using Embedding Layer which then acts as input to our LSTM cells.

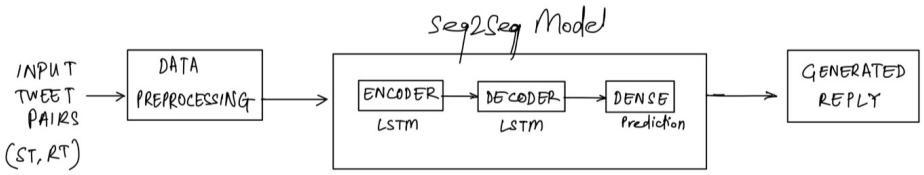

Figure 4: Seq2Seq Architecture

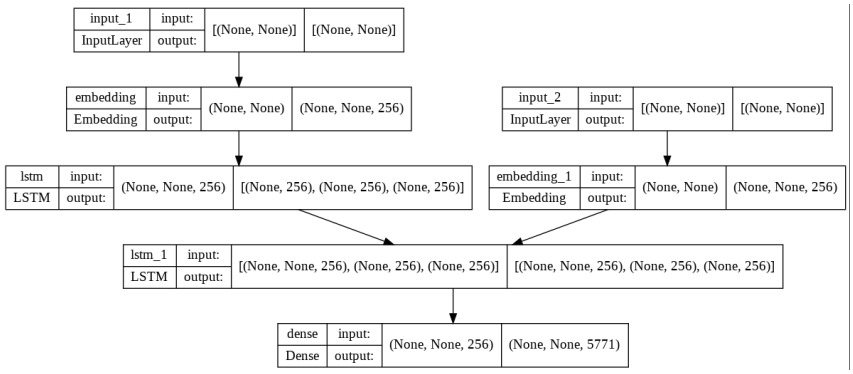

Figure 5: Seq2Seq Model

Then we input these pairs of conversations into Encoder and Decoder. So that means our Neural Network model has two input layers. We have the source tweet as input for the encoder. Similarly, the reply tweets for the respective source act as input to the decoder. In the model itself, the layers can be broken down into 4 parts:

1. Input Layer (Encoder and Decoder)
2. Embedding Layer (Encoder and Decoder)
3. LSTM Layer (Encoder and Decoder)
4. Decoder Dense Output Layer

We designed the model to estimate the most probable following word, up to the end. Fully connected layer computes the probability of each word to appear next with the current state of the cell. The word with the highest probability is accepted and appended to the sentence. We feed the model with the embedding of the chosen word to determine the following state. This process repeats until the model outputs the ⟨end⟩ tag. Generated tweets seem very similar to the real ones.However, this approach always comes up with the same reply tweet for the source tweet entered.

6.3 Approach 2: Dual Transformer model using T5

In this approach, we use a dual transformer layer architecture, whereby both the layers are pre-trained T5 models. Our objective is to generate an input dataset which consists of the given dataset information for this project in the first layer, and train this data using the second layer such that we can generate new replies efficiently for new reply threads against this trained model for each of the 4 labels separately. The first transformer layer is a pre-trained hugging face transformer model (snrspeaks/t5-one-line-summary) which has been trained on a lot of research papers. Using the data that we extracted, i.e. parent tweet id, tweet label and list of reply tweets per source tweet, we generate a golden stance reply for each source tweet in a encoder-decoder style. For this we use the summarization technique, in which all the reply tweets for a particular source tweet (obtained from the 'text list' attribute in our pre-processing step) are summarized into a new reply tweet, This is compared against each reply tweet that will be our golden text each time. After the new reply tweets are generated, we create a new input dataset which maps each of this to its respective source tweet, i.e. having 2 attributes - 'source-text' and 'target-text'

In the second layer, we used the above input and trained it according to our model (more details below) for each label. When we get new reply threads and its source tweet, we are then able to generate its respective new reply tweet. We are able to generate a decent reply within 3 epochs for the same.

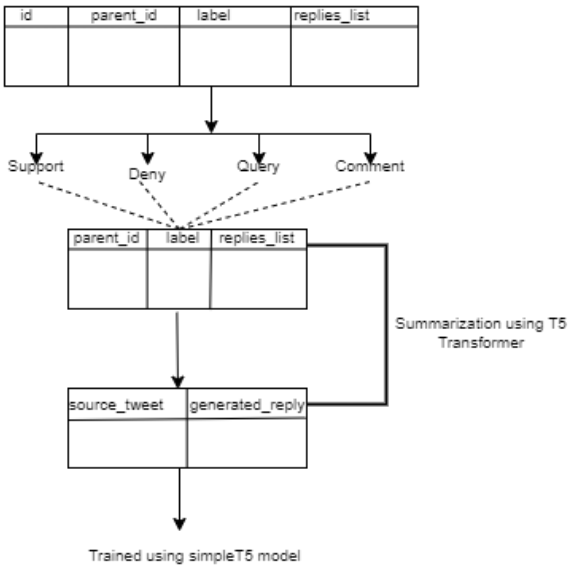

Figure 6: Subtask C Dual Transformer Architecture

T5 (Text-To-Text Transfer Transformer) is transformer library by Google AI, which has many pre-trained models. In both the layers, we use the simpleT5 framework (built on PyTorch) which helps us fine tune our model.

We generate an input dataset from the first layer separately for each of the labels separately to enhance its performance and output - as during summarization, like replies will help generate a better reply tweet. These 4 input files are then trained in the second layer using the simpleT5 model to efficiently generate new replies.

6.3.1 Subtask C: Results

The Subtask C result score comparisons are provided below:

Similarly, BLEU scores can be generated for the rest of the 4 label groups as well. As we can see, the BLUE score has efficiently increased in the enhanced model for the support label group.

6.3.2 Subtask C: Qualitative Analysis

The main issue with the Seq2Seq approach is that we did not have enough data to train the models properly. As we can see from the scores generated above, the amount of data greatly influenced the

Parameters	BERT Score	BLEU Score
Support Tweets	0.55	0.35
Deny Tweets	0.61	0.46
Query Tweets	0.61	0.58
Comment Tweets	0.76	0.71

Table 3: Seq2Seq Scores

Parameters	BLEU Score
Support Tweets	0.86

Table 4: Transformers Scores

scores of this model. Since the Comment based dataset was the largest, with 2907 data points, it gave the best results out of all the other categories in Seq2Seq model. In the second approach, We considered using transformers in this approach as it performs faster processing and a pre-built model would give us a different perspective with respect to the reply generation. For new reply tweet generation, as we use the summarizing technique as in each label group (support, query, comment, deny), like reply tweets appended into a multi-line string (which will be used as an abstract to summarize) provide a better short text.

6.3.3 Subtask C: Error Analysis and Drawbacks

During these approaches, we analyzed that the tweet data extracted from the project dataset was not sufficient to provide good results in the baseline model. To overcome the problem of small datasets in Seq2Seq model, we tried using external datasets (like Twitter Customer Service Corpus from Kaggle) which had similar format to that of our data. However,due to the training process in both approaches, we found less computing power available to us to be a drawback in the timely computing of these models. We also tried using a specific tweet-based model (vinai/bertweet-large) which was trained on a large tweetbase, but the results during the summarization step for our tweetbase weren't that effective as compared to the other model that we eventually used. Hence, we did not proceed training our data with that model.

6.3.4 Subtask C: Future Scope of improvements

To further analyze performance levels of tweet generation using the required parameters, we can consider implementing the approaches using GRU and GPT2 models respectively.

7 Conclusion

Through this project, we were successfully able to classify a reply tweet into a Support, Query, Comment, Deny label class for the SDQC classification task. And for a source tweet's rumor detection, we classified it properly into whether it's True, False or Unverified and for its veracity prediction, we analyze the RMSE scores calculated. Both the above sub-tasks undergo multi-class classification and are analyzed using their Macro-averaged F1 scores. For stance generation, we were able to generate new tweets using its source tweet, corresponding tweets in the reply thread and their labels through Sequence to Sequence and Transformer models approach.

8 Work Distribution

UBID	Name	Distribution
50419574	Shreya Mukherjee	33.33
50416894	Thelma Gomes	33.33
50388502	Shashank Desai	33.33

All the tasks were divided equally among all the team members. Eeveryone contributed in some part or the other in all sub-tasks including preprocessing, model generation, testing and research as well as report writing.

References

Hareesh Bahuleyan and Olga Vechtomova. 2017. UWaterloo at SemEval-2017 Task 8: Detecting Stance towards Rumours with Topic Independent Features. In *Proceedings of SemEval*, ACL.

G. Gorrell, E. Kochkina, M. Liakata, A. Aker, A. Zubiaga, K. Bontcheva, and L. Derczynski 2019. SemEval-2019 Task 7: Rumoureval, determining rumour veracity and support for rumours, in *n Proceedings of the 13th International Workshop on Semantic Evaluation*, ACL.

Elena Kochkina, Maria Liakata, and Isabelle Augenstein. 2017. Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification with Branch-LSTM. In *Proceedings of SemEval*, ACL.

Ilya Sutskever and Oriol Vinyals and Quoc V. Le 2014. *Sequence to Sequence Learning with Neural Networks*, CoRR.

Yusuf Mücahit Çetinkaya, İsmail Hakkı Toroslu, and Hasan Davulcu 2020. *Developing a Twitter bot that can join a discussion using state-of-the-art architectures)*, Springer Nature 2020.

Kim J, Oh S, Kwon O-W, Kim H 2019. *Multi-turn chatbot based on query-context attentions and dual wasserstein generative adversarial networks)*, Appl Sci 9(18):3908

Lei W, Jin X, Kan M, Ren Z, He X, Yin D 2018. *Sequicity: simplifying task-oriented dialogue systems with single sequenceto-sequence architectures)*, In: Proceedings of the 56th annual meeting of the association for computational linguistics, ACL 2018, Volume 1, Association for computational linguistics, pp 1437–1447

Varol O, Ferrara E, Davis CA, Menczer F, Flammini A 2017. *Online human-bot interactions: detection, estimation, and characterization)*, In: Proceedings of the eleventh international conference on web and social media, AAAI Press, pp 280–289

Wang D, Nyberg E 2017. *A long short-term memory model for answer sentence selection in question answering.)*, In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 2: Short Papers), pp 707–712

Wen T, Gasic M, Mrksic N, Su P, Vandyke D, Young S J 2015. *Semantically conditioned LSTM-based natural language generation for spoken dialogue systems.)*, In: Proceedings of the 2015 conference on empirical methods in natural language processing, EMNLP 2015, The Association for Computational Linguistics, pp 1711–1721

Wu Y, Li Z, Wu W, Zhou M 2018. *Response selection with topic clues for retrieval-based chatbots)*, Neurocomputing 316:251–261

Sourav Bhattacharyya 2020. *Classification using Long Short Term Memory GloVe (Global Vectors for Word Representation)*, Medium.

Shivanand Roy 2021. *Abstractive Summarization with SimpleT5*, Medium.

Abhishek Jaiswal 2020. *Multiclass Classification Using Transformersfor Beginners*, Analytics Vidhya