Stance Classification with Target-Specific Bidirectional LSTM Network

Mohit Khosla [1], Jigar Parekh [2]
[1, 2] School of Science, RMIT University, Melbourne, Australia
{s3732729, s3731304}@student.rmit.edu.au

Abstract: In stance classification, the tweets are classified based on the targets which defines a limit to the task, wherein a network would determine the stance by making predictions with the help of tweets and targets. In our work, we are proposing a Bidirectional LSTM Network which considers Targets and Tweets to predict the stance. Our approach explores the deterministic possibilities where the model can generalize well between the different targeted tweets. We implemented a RNN which takes multiple inputs and the features learnt from one distinct input is concatenated to the second input along with its features. Our training was based on the GloVe embeddings of 50 Dimensions. Based on our outcome, we realized that our model can corelate some useful information shared between different targeted tweets to minimize the generalization error in definite parameter settings. The use of Transfer learning let us utilize the available data to its maximum potential value.

I. Literature Review

Stance Detection is considered as an opinion-based mining where the system can detect whether the author of the tweet is trying to speak in Favor, Against or None for that specific target. Earlier researches have been mainly biased towards the sentiment of the tweets to determine the sense of positive of negative context for detection of stance.

However, the relevance of target was ignored for some proposed systems, such as Guido Zarrella & Amy Marsh [1] MITRE system, which makes use of unique Hashtags rather than target to determine the stance. Their system was able to prove the best amongst the 19 teams proposing their methodologies with an average F1 score of 67.8. We have considered this as a benchmark score to corelate our accuracy score. They used a Recurrent Neural Network which was able to learn the features from the corpus of unlabeled data. They trained their embedding vector of words by using the word2vec phrases using a skip-gram method. Those learnt representations were used with hashtags to make the predictions. Their phrase representations were tuned with enormous instances labeled to provide better performance on the test data. They also incorporated the use of Transfer Learning in their progression network. For their network architecture, the inputs were using the one hot encoding and the tokens of that encoded vector was represented to index value of the token position in the vocabulary. They had a sequential 256 Dimension embeddings to define the matrix which was an input to their LSTM layer of 128 units. The output from this layer was connected to a 128 Dense layer with 90% of dropout rate. Lastly, this output layer was connected to a softmax layer with 3 units to determine the output stance (Figure 3). However, they had performed some pre-processing such as lowercasing of the tweets to map the word2vec embeddings, removing of punctuation and finding the unique hashtags.

Unlike Guido Zarrella & Amy Marsh [1], Jiachen Du *et. al.* [2] made use of targets to determine the stance along with the tweet's corpus. They proposed a novel state-of-the-art approach with Deep Attention Neural Network which is on the urge of booming in the near future. Their novelty of building a Target-specific Neural Network proved to be a significant improvement over the approach given by Guido Zarrella & Amy Marsh [1]. TAN architecture had two main functional components, one, RNN used to do the feature extraction for the text, second, a densely connected layer network for target inputs using attention selector. Those two layers were combined to perform a cross multiplication element by element operation for the task. They compared their approach with multiple baseline models including MITRE [1] which outperformed them by 3.54% for the target "Hillary Clinton". The target specific attention proved to be significantly accurate to beat all the existing baseline models by 3.03% for an ordinary LSTM embedding. Not only this, but they also performed their evaluation of the Chinese tweets to build a cross-sectional interdependency between the learnt feature and weights. The results on Chinese data were better than the English data, due to better textual annotation of Chinese data and the balance ratio between targets. Lastly, their observations resulted that TAN performs better than the traditional LSTM in case of time as well of space complexity. This was proven, with the distinct learning curve comparison performed against ordinary LSTMs. It was claimed that usually a LSTM layer takes minimum 5 iterations to converge, but TAN takes lesser iterations to converge, proving to have a dominant capability (Figure 4).

Similar to Jiachen Du *et. al.* [2], Penghui Wei *et. al.* [3] had an identical approach which also made use of targeted tweet information. In this research, their implementation was mainly deviated towards extracting the target information for useful stance classification. Their model adapts an approach where it first learns the interdependency of tweets conditional representation based on the given target. In comparison to Jiachen Du *et. al.* [2] who were more biased towards the cross multiplication to perform the extraction elementwise, Penghui Wei *et. al.* [3] leans to an iterative learning process which is target guided to determine some clues which had stance dependencies by making a continuous correlation between tweets and target keywords. Their architecture TGMN-CR is segregated into three sub modules. Firstly, the Text Encoder Module, which determines target specific conditional representation of tweets based on the fixed length vectors for the target embeddings. Secondly, the Attention and Memory Module which considers the continuous relations of tweets and target keywords, is the heart of their model. Lastly, the vector representation of the tweet and target pair is concatenated with the memory vectors to predict the stance using the Stance Classification Module. They used GloVe embeddings of 100 Dimensions as a pre-trained corpus. They also used Adam optimizer with a learning rate of 5e-4. To prevent overfitting, L2 regularization of 0.001 and dropout of 0.3 was used. Their model outperforms some baseline models under the condition when a tweet does not explicitly provide an opinion for a given target. Their implementation had an overall 71.04% accuracy (Figure 5).

In traditional approaches, the use of LSTM and GRU does not address the problem of vanishing gradients and overfitting. This problem was addressed by Siddiqua et. al. [4] in her novel research. The problem of vanishing gradients can be overlooked by creating densely connected Bidirectional LSTM layers or nesting of LSTMs which will not only help reduce overfitting, but also would take care of long-term interdependencies. In this paper, two modules were coupled together with an attention mechanism to amplify the importance of the elements. These two modules are Bidirectional LSTM and nested LSTM. They also considered multi kernel convolution filters for their approach with a kernel size of [2,3,4,5] to find and fetch high level appending of tweets and targets. Next, these learnt features are given as an input to Bidirectional and nested LSTMs for employing a forward feed network to focus on hidden states sequences. They removed punctuations and stop words as they hold less meaning for stance classification. Adam optimizer was used with a learning rate of 0.001 for a batch size of 32. L2 regularization of 0.01 was used to prevent overfitting. The results depicted that their model outperformed baseline models for targets "Hillary Clinton" and "Feminist Movement" (Figure 7).

Another approach similar to Penghui Wei *et. al.* [3] was proposed by Qingying Sun *et. al.* [5] wherein the author describes about the concept of joint neural network for the prediction of stance and sentiment for correlations between the posts concurrently. Collective knowledge representation was considered for the interaction between the stance and sentiment. This research does not talk about the target specific data but covers the relevance of sentiment for the prediction of stance. However, their model is predicting sentiments and stance altogether. The main task is to perform the stance prediction and the auxiliary task is to predict the sentiments. They leverage the use of parameter sharing between the layers of LSTMs of tweets and sentiments by using a neural stacking framework. Like Guido Zarrella & Amy Marsh [1], this author also performed a unigram and n-gram of word2vec for the input features of the LSTM network. Their architecture uses two inputs, one for tweets and the other for stance. The main LSTM layer is connected to a series of hidden layers as multi-layer perceptron. In addition to this, the features learnt from the auxiliary shared LSTM layer is used for neural stacking. The output of the neural stack is combined with the output of dense layers using a join representation to make the stance prediction. On the flip side, the output of the auxiliary layer is used independently, connected to a series of dense layers to make the sentiment prediction. They also used 5-fold cross validation on training data to fine tune the parameters. They used 128-layer LSTMs with a batch size of 32 and epochs of 30. Each fold represents a targeted value for the tweet and validated individually. 5 separate models were prepared and evaluated and a macro F1 score was used to take an average across all the targets (Figure 8).

Isabelle Augenstein *et. al.* [6] research shows that keeping aside the targeted data for training, unlike all the previous researches, this author focuses on building conditional encodings for the LSTM layers where a tweet's representation is dependent on the target. The claim that their approach outperforms the traditional methodology for approaching the task. They started their implementation by first removing the irrelevant words which are "#SemST, "RT" and "via". They also converted the tweets to lowercase trained the network on word2vect embeddings. They collectively grouped the data which is unlabeled from the word2vec corpus with the training data and used a n-gram to perform the training, like Qingying Sun *et. al.* [5] and Guido Zarrella & Amy Marsh [1]. 100 units LSTM layers were chosen to keep the layer as same as the word embedding dimensions which is 100D. The initial learning rate of 1e-3 with Adam optimizer was used. The loss function used was cross-entropy. Lastly the use of dropout was also considered to reduce the overfitting. Results showcase that their conditional encoding methodology generalize well on the unseen data with a macro F1 score of 49% for FAVOR and AGAINST (Figure 6).

Chang Xu *et. al.* [7] research used a novel approach of building a network based on self-attention networks which considers cross-targets. They named their model as CrossNet which had four distinct layers to predict the stance. Starting from the bottom most layer (Embedding) to the topmost layer (Prediction Layer) they take multiple inputs as tweet and stance separately. The layer one which is the embedding layer, it takes two inputs, sentences, and targets. Word embeddings were used to represent each work as a unique token in their vector. Two sequences of word vectors were created, one for tweets and the other for sentences. Following the embedding layer, comes the context encoding layer, which takes the stream of vectors independently using a Bidirectional LSTM to cover the context of words in both the direction. i.e. left to right and right to left. Once the forward and backward pass process has been completed the Aspect Attention Layer comes into picture. This layer mainly focuses on inferring the knowledge from tweets and targets simultaneously as their major concern was to learn the domain specific knowledge for each tweet. Lastly, the prediction layer predicts the stance based on the labeled tweets and targets. A MLP architecture is used densely using a Softmax activation function. They used F1 score as their performance metric since the dataset is imbalanced. The results depict that their performance was improved by almost 26% per class when compared to the traditional SVM and CNN architectures (Figure 9).

To sum up, Isabelle Augenstein *et. al.* [6] suggested that the use of #SemST keyword is irrelevant to the task and converting the digits to numbers and replacing the symbols such as % to words would be beneficial in slight improvements. Siddiqua et. al. [4] addressed the problem of vanishing gradients in traditional techniques and she had used Bidirectional LSTMs to overcome that. Her researches also suggested that an L2 regularization of 0.001 or smaller regularization would be favorable for the task, along with Adam optimizer with a LR of 1e-2. Isabelle Augenstein *et. al.* [6] research also backs up with the L2 regularization of 0.001 and Adam optimizer. Isabelle's research and Penghui Wei *et. al.* [3] both used a GloVe embedding for the twitter data with a 100D vectors which suggested that fewer the data points, fewer the dimensions to be used for the task. Guido Zarrella & Amy Marsh [1] and all the other researches depicted that the use of 128 units in LSTM layers for their implementations preserved the information for the fully connected dense layer after concatenation. Lastly, the golden nuggets found from all the state-of-the-art researches, and the insights learnt, we are now going to talk about our side of the implementation.

II. Proposed Methodology

Here, we will describe our proposed approach and architecture of our Target-Specific Bidirectional LSTM RNN. The task was to take multiple inputs with separate

embeddings for tweets and its target for the model and predict the stance label of tweet into either of the three classes i.e. AGAINST, FAVOR or NONE.

A. Exploratory Data Analysis

Firstly, the dataset was shuffled so that it makes the model training coverage faster while preventing any bias and not letting the model learn the order of training. To perform basic cleaning and pre-processing to the data, we first performed exploratory data analysis on the training and testing datasets to look at the basic statistics of the tweet data. By looking at the frequencies of each column we verified that there were no null values present in the dataset. Grouping the data based on "Target", we noticed that there was class imbalance and out of the five classes, the target "Climate Change is a Real Concern" had the lowest number of tweets, i.e. 395, meanwhile the rest four targets had tweets ranging from 510-690 tweets. So, considering the target "Atheism", it had 513 tweets out of which 304 belonged to the stance "AGAINST". The target "Climate Change is a Real Concern" had 395 tweets out of which 212 belonged to the stance "FAVOR". The target "Feminist Movement" had 664 tweets out of which 328 belonged to the stance "AGAINST". The target "Hillary Clinton" had 689 tweets out of which 393 belonged to the stance "AGAINST". Lastly, the target "Legalization of Abortion" had 653 tweets out of which 355 belonged to the stance "AGAINST". Hence, to check the class balance of tweet with respect to stance, we grouped the training data by stance. The frequency of tweets under the stance "AGAINST" was 1395 which was almost double the other classes. The stance "FAVOR" had 753 tweets meanwhile the stance "NONE" had 766 tweets (Figure 10 & 11).

B. Data Pre-processing

We then segregated the tweets for each target to build a word cloud (Figure 12) for each target class. It gave a visual representation of most frequent keywords used in the tweets and we observed that a hashtag "#SemST" was being used in the tweets throughout all the target classes, which won't have any impact on the model as suggested by Isabelle Augenstein *et. al.* [6], so we decided to discard it from all the tweets. Along with this, we even replaced the % and $ symbols to actual words in the tweets and replaced digits into numbers so that it can be meaningfully represented with the corresponding words, also described by Isabelle Augenstein *et. al.* [6]. Not only this but we also removed all the non ascii characters from the tweets so that the model does not get any unrecognized character.

The stance values do not have any intrinsic ordering and hence we simply encoded it with integers randomly. The integer encoding maps each stance value to an integer. Following this, tokenization was performed on all the tweets and targets which basically breaks the raw text of tweets into small chunks by splitting a string of tweet into list of tokens. The reason behind tokenizing is that it helps in evaluating the meaning of any text by analyzing a sequence of tokens or words. This was performed using TweetTokenizer package as it preserves the hashtag/tag symbol and the word in a single token, which was helped by Jiachen Du *et. al.* [2].

There are many words that are most common throughout the text content which generally do not add any meaning to the sentence or sequence of words and hence do not provide any significant information to the model. These words are referred as stop words and it can be discarded without compromising the meaning of a sentence or corpus. It even decreased the total number of tokens meanwhile making the training process faster without having any impact to the accuracy of the model. We followed this process as it was suggested by Siddiqua et. al. [4] research.

The input tokens are first converted into text sequences wherein each word from the corpus was stored in a dictionary having a unique key value for each word. Then these key values are stored in the sequences. The tweet data after preprocessing does not necessarily have the same size. However, as the RNN requires to have all the inputs of same shape, there was a need to add padding to each tweet and target sequences. The max length of the tweet and target was evaluated, and a padding was added to each tweet and target sequence with the max tweet and target length.

C. Transfer Learning

These target sequences are further converted into vectors, based on the semantics of the related words. So, the sequence of the tokens is converted into word embeddings, which is a type of representation that will enable the words to have similar representation having similar meaning. It basically maps the semantic meaning into geometric space. Here, all the word embeddings are stored in a predefined vector space as real valued vectors. This makes the representation of each word densely distributed with low dimension. We have used GloVe word vector of twitter corpus having 27B tokens from 2B tweets, Chang Xu *et. al.* [7]. GloVe stands for "Global Vectors for Word representation". It preserves the context of each word globally by creating a co-occurrence matrix by evaluating the probabilities of each words co-occurring with other words in the corpus. Since, we have a small dataset, the word embeddings are initialized with a low pre-trained 50 dimension, as described by Penghui Wei *et. al.* [3] and Isabelle Augenstein *et. al.* [6] where fewer the data points, fewer the dimensions to be used for the task.

D. Target-Specific Bidirectional LSTM Architecture

As discussed above, the performance of the model to classify the stance could only be improved by feeding in both tweet and target features to the model. Hence, we propose an RNN model which is Target-Specific and uses Bidirectional LSTM Network.

Our model is target-specific i.e. considers both the target and the tweet where the information is inferred and shared for the stance prediction, unlike Guido Zarrella & Amy Marsh [1] MITRE system which was trained on unlabeled corpus of data of just tweets. The word embeddings for input are initialized by GloVe 50-dimensional Twitter corpus embeddings, Chang Xu *et. al.* [7]. The model takes in two inputs, one was the tweet's embedding and other was the target's embedding. But the traditional RNN usually face a problem of vanishing gradients as the weights must propagate through many steps which was even addressed by Siddiqua et. al. [4]. To overcome this issue, LSTM (Long short-term memory network) comes in picture. So, our multiple embeddings are fed into LSTM. We used 128 units of dimensionality of the output space for each LSTMs. The

output of the LSTM keeping all the return sequences of al batches is fed to bidirectional LSTM.

Entire sequence was passed on to the respective next bidirectional-LSTM layers having which have 256 units of dimensionality of the output space including both the forward and backward passes. By using such densely connected bidirectional LSTM, the weights are preserved, and the vanishing gradients and overfitting is avoided. The forward pass of the Bi-LSTM reads the tweet sequence from left to right in forward direction. Whereas the backward pass reads the tweet from right to left in backward direction. So, here we apply forward pass twice, once for the forward cells and the other for the backward cells. The main reason behind using Bi-LSTM is that the tweets generally have the hashtags at the end of the tweets. So, it makes more sense to the model while performing the backward pass, as the model can now correlate the hashtags with the context of the tweet.

Furthermore, to reduce the complexity of the model and reduce overfitting, we have used L2 regularization as it forces the weights to be smaller but avoids from making it zero. As Siddiqua et. al. [4] suggested that an L2 regularization of 0.001 or smaller regularization would be favorable for the task, along with Adam optimizer with a LR of 1e-2. Hence, we used a lower value for L2 i.e., 0.0001 and LR of 1e-2. Added dropout and recurrent dropout of 0.2 to the LSTM and Bi-LSTM layers and 0.3 to the dense layers. The dataset was split into 80% training and 20% testing during the fitting of model with a batch size of 64 (Figure 15 & 16).

We have used reduce on plateau, which reduces the learning rate by a factor of 0.1, when the validation loss has no improvement for 20 epochs, this will then drop the learning rate to monitor noticeable changes. Since training the model too little leads to under-fit train and validation dataset and training the model too much may overfit the training data and perform poorly on the validation data. Our epochs are 500 for training the data, so that when the train and validation curves converges, the best weights can be utilized to make the predictions. Therefore, we decided to add early stopping which monitors the performance of the model and triggers to stop the training when the generalization error increases.

E. Predictions and Results

Our implementation achieved an F1 score of 63.22% (Table 1). The reason we had used F1 score as a metric is because of the data imbalance across classes and this issue has also been addressed by Chang Xu *et. al.* [7] where the author is considering F1 score for their results along with Qingying Sun *et. al.* [5] and Isabelle Augenstein *et. al.* [6].

	precision	recall	f1-score	suppor
AGAINST	0.68	0.91	0.78	715
FAVOUR	0.74	0.39	0.51	304
NONE	0.46	0.26	0.33	230
accuracy			0.67	1249
macro avg	0.63	0.52	0.54	1249
weighted avg	0.65	0.67	0.63	1249

Table 1: Prediction Results on Test Data

Based on our score, we have set our model's accuracy to closely align with our benchmark score of 67.8 achieved by Guido Zarrella & Amy Marsh [1]. They had considered a macro average F1 score of AGAINST and FAVOR for their results which made them achieve 67.8%, whereas when we computed our macro average F1 score for both the same classes we achieved 64.5%, which depicts that we nearly land up close enough to their score. In addition to this, we had used categorical cross-entropy as loss metric and categorical accuracy for performance metric, following Isabelle Augenstein *et. al.* [6]. We also plotted a confusion matrix (Figure 1.1) to determine the overall significance of the class predictions, which indicated that the model is getting more biased towards the AGAINST class over the other two classes.

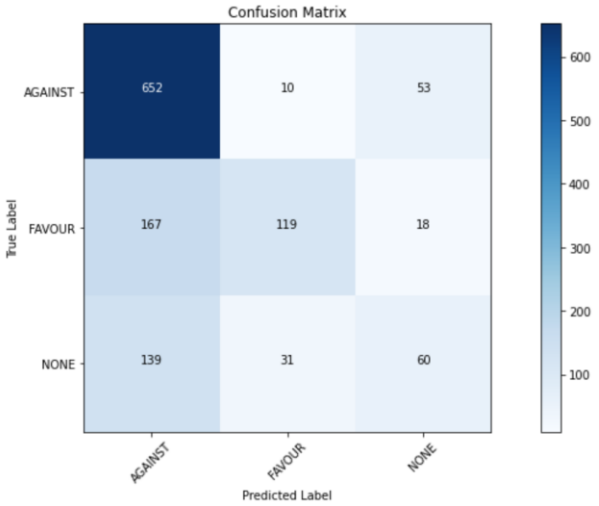

Figure 1.1: Confusion Matrix on Test Data

F. Independent Evaluation

To perform independent evaluation, we collected tweets by searching relevant keywords and hashtags. We saved 10 tweets for each target class out of which 4 tweets for "AGAINST" and "FAVOR" stance classes each and 2 tweets for "NONE" class, summing up to a total of 50 tweets. We manually annotated stance values to the tweets.

The model's performance on this data was evaluated using the F1 score metric as well. The model achieved an F1 score of 42% (Table 2).

	precision	recall	f1-score	support
AGAINST	0.46	0.80	0.58	20
FAVOUR	0.40	0.20	0.27	20
NONE	0.60	0.30	0.40	10
accuracy			0.46	50
macro avg	0.49	0.43	0.42	50
weighted avg	0.46	0.46	0.42	50

Table 2: Prediction Results: Independent Evaluation Data

As the training dataset for the stance "AGAINST" is double the other classes, our model is much inclined towards predicting other stance classes also as "AGAINST". Hence, considering the macro average F1 score of the targets "AGAINST" and "FAVOR", it is the same as the weighted average F1 score, i.e. 42%.

There are several reasons behind this poor performance on the real-world data. Firstly, the class imbalance and the size of dataset plays a major role (Figure 1.2). The tweets are even inconsistent while having some words sticked to each other. This could have been improved by preprocessing it again. Secondly, observing the training data, the annotations are not

very accurate by our observation. There is no relevance of some tweets with its corresponding target and stance class.

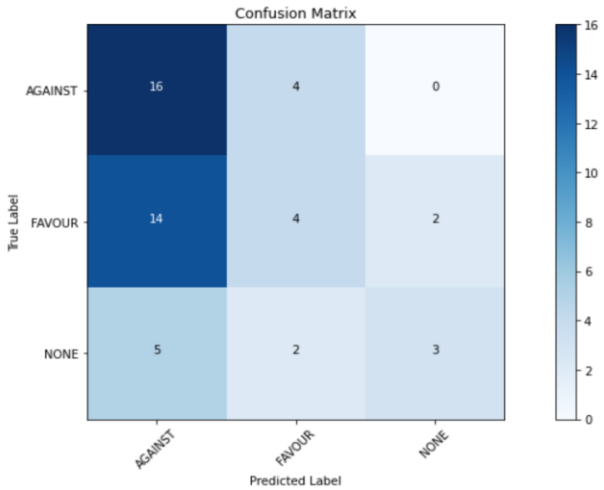

Figure 1.2: Confusion Matrix: Independent Evaluation

G. Ultimate Judgement

By performing various hyper-parameter tuning and by observing approaches implemented in various research papers, we chose to build our architecture as Target-specific Bidirectional LSTM. Depending on the overall outcome from all the models that we had experimented (Figure 13 & 14), we prefer to have 128 units for both LSTM and Bidirectional-LSTM layers. Along with this, a dropout and recurrent dropout ratio of 0.2 and L2 regularization of 0.0001 was the optimal hyper-parameter setting for our approach. The learning rate 0.001 of Adam optimizer was the best for the task. We aimed to reduce the generalization gap by improving the accuracy and reducing the loss between the train and validation curves. However, the vocabulary for training and testing data and evaluation data had some variations. Few words which are present in the test data and evaluation data are not being used in the training set, creating a bias towards the prediction. This is a limitation that we had discovered where our model might predict incorrectly (Figure 2).

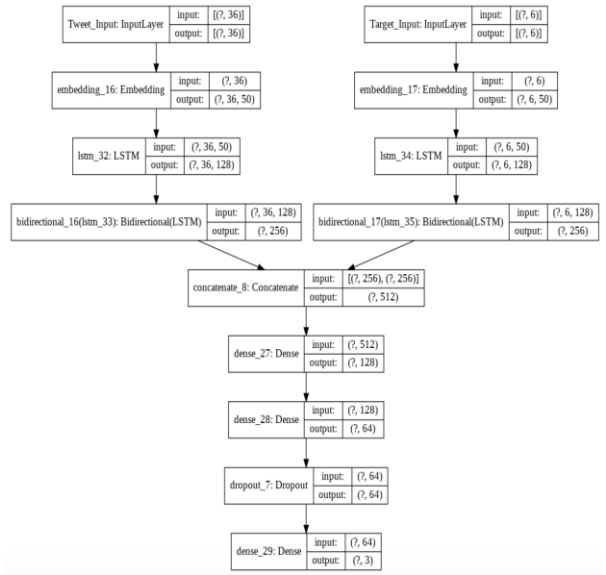

Figure 2: Model Architecture

H. Technical Issues Encounterd

Some underlying concerns with the biases are present, although the model evidently showcase the solution for the given problem.

Data Imbalance: The tweets data had a very limited number of textual contents for this task to be completed adequately. Therefore, it is difficult to derive some conclusions for this model if the data density is increased. On the contrary, improved data would be helpful to design a newer architecture using our proposed approach and tend to achieve some higher accuracies across classes.

Modality (Fewer Parameters): Considering the information provided for the task, it is difficult to determine the stance just with the tweets and targets. With such fewer parameters it is insufficient for the verification of the derived conclusion. However, by increasing the parameters for the data i.e. by extracting some additional information along with the targets such as user profiles, their relationship networks etc. would result in an improved performance.

I. Ethical Considerations

Some ethical considerations are needed to understand while claiming a stance for a particular organization, group or an individual.

When encountered, ethical considerations are the most subjected issues while working closely with some political and some controversial targeted tweets. The confidentiality of the tweets should be maintained while taking care of the user's consent. Not revealing someone's identity for making a stance on a particular target is must. Lastly, adhering to handling the user's claim about a particular target, must not be altered, or modified leading to unethical tampering of user's integrity and honesty obliged to that tweet.

III. References

[1] Zarrella, G. and Marsh, A., 2016. Mitre at semeval-2016 task 6: Transfer learning for stance detection. arXiv preprint arXiv:1606.03784.

[2] Du, J., Xu, R., He, Y. and Gui, L., 2017, August. Stance classification with target-specific neural attention networks. International Joint Conferences on Artificial Intelligence.

[3] Wei, P., Mao, W. and Zeng, D., 2018, July. A target-guided neural memory model for stance detection in Twitter. In 2018 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[4] Siddiqua, U.A., Chy, A.N. and Aono, M., 2019. Tweet Stance Detection Using Multi-Kernel Convolution and Attentive LSTM Variants. IEICE TRANSACTIONS on Information and Systems, 102(12), pp.2493-2503.

[5] Siddiqua, U.A., Chy, A.N. and Aono, M., 2019. Tweet Stance Detection Using Multi-Kernel Convolution and Attentive LSTM Variants. IEICE TRANSACTIONS on Information and Systems, 102(12), pp.2493-2503.

[6] Augenstein, I., Rocktäschel, T., Vlachos, A. and Bontcheva, K., 2016. Stance detection with bidirectional conditional encoding. arXiv preprint arXiv:1606.05464.

[7] Xu, C., Paris, C., Nepal, S. and Sparks, R., 2018. Cross-target stance classification with self-attention networks. arXiv preprint arXiv:1805.06593.

IV. APPENDIX

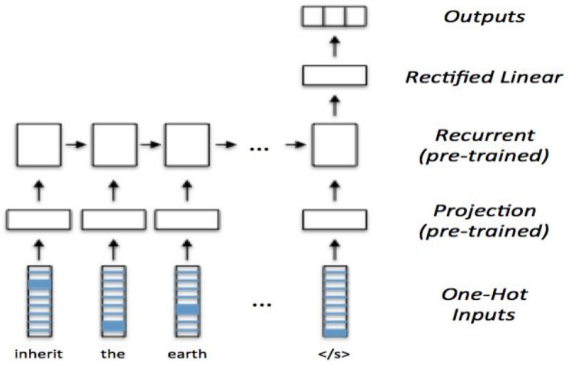

Figure 3: Guido Zarrella & Amy Marsh [1] Architecture

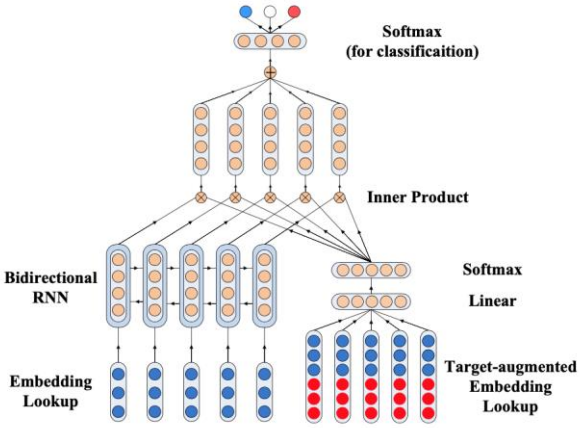

Figure 4: Jiachen Du *et. al.* [2] Architecture

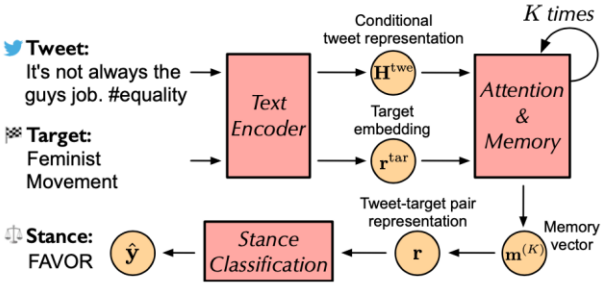

Figure 5: Penghui Wei *et. al.* [3] Architecture

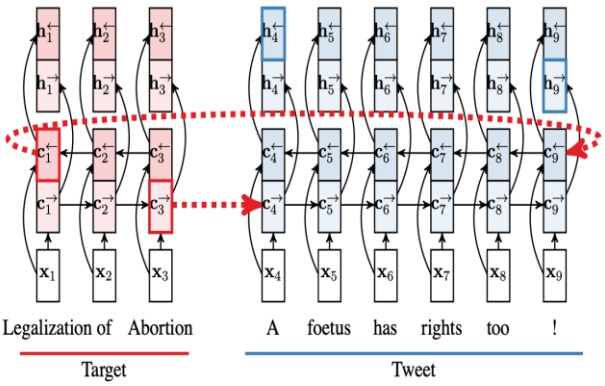

Figure 6: Isabelle Augenstein *et. al.* [6] Architecture

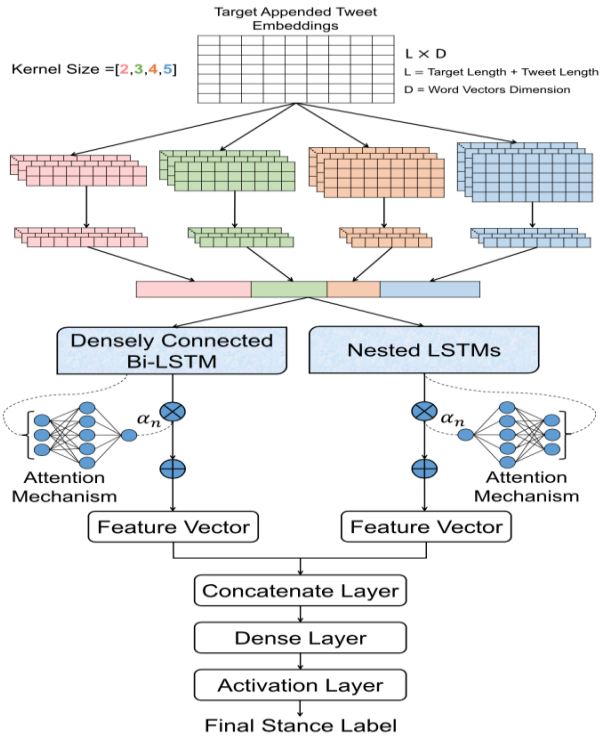

Figure 7: Siddiqua et. al. [4] Architecture

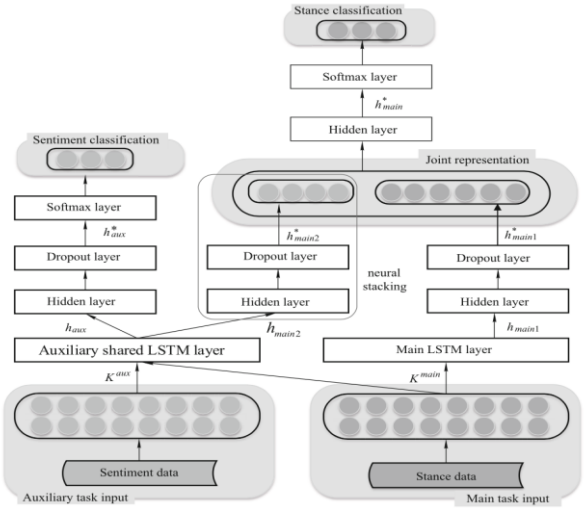

Figure 8: Qingying Sun *et. al.* [5] Architecture

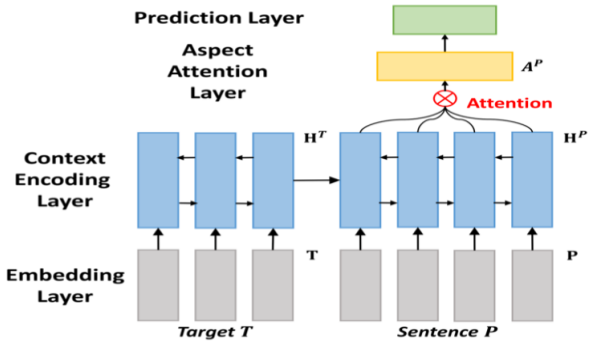

Figure 9: Chang Xu *et. al.* [7] Architecture

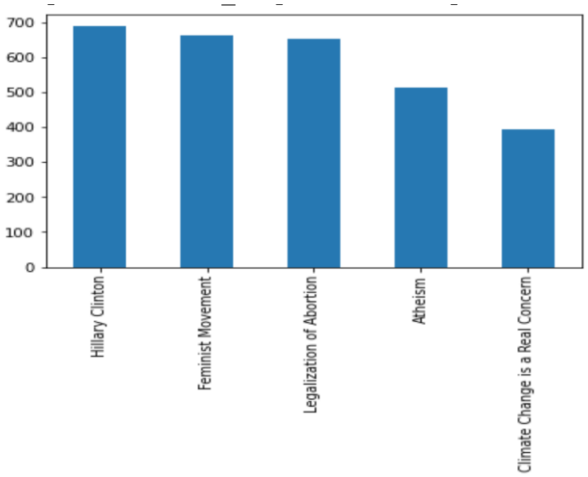

Figure 10: Data distribution based on Targets

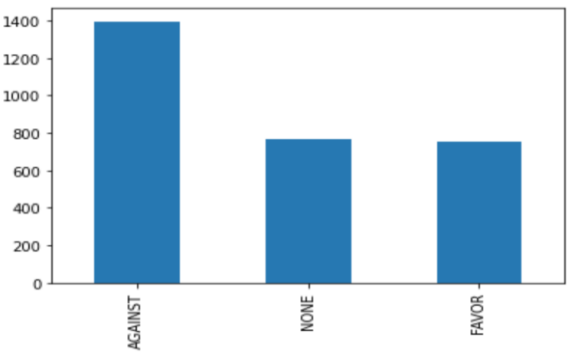

Figure 11: Data distribution based on Stance

Figure 12: Word cloud for all tweets

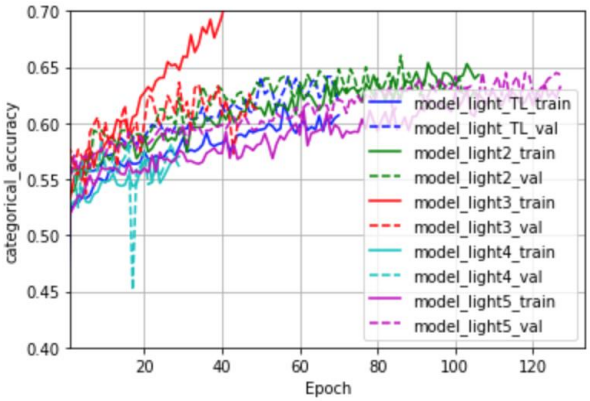

Figure 13: Accuracy for Multiple Architectures

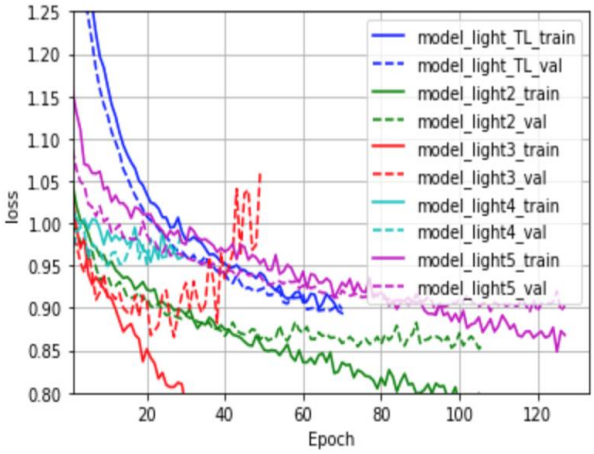

Figure 14: Loss for Multiple Architectures

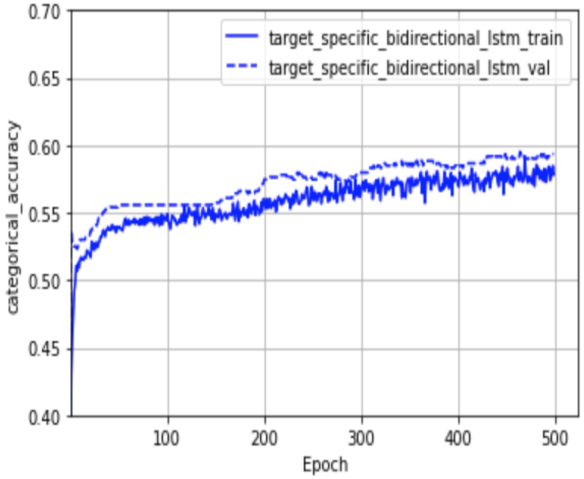

Figure 15: Accuracy for our Approach

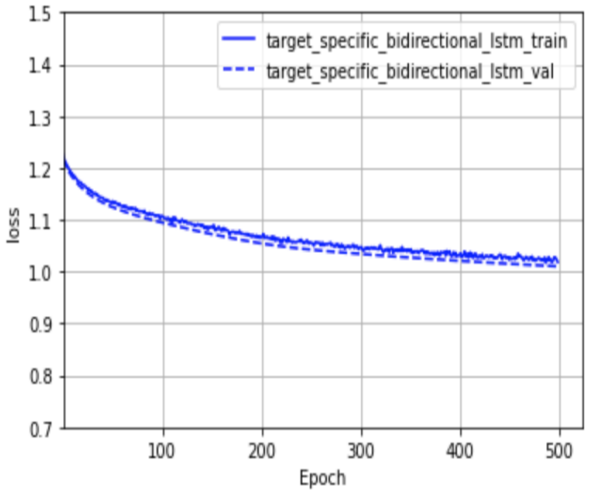

Figure 16: Loss for our Approach