COSC2779 – Deep Learning

Assignment 2 – Stance Classification in Tweets

Rutvi Macwan | s3773570
Salina Bharthu | s3736867

Table of Contents

Table of Figures

Introduction

Purpose of this project is to develop a Deep Learning classifier that can identify the 'Stance' between a given set of 'Tweet' and 'Target'. The stance can be either in favor or against, or there could be no existence of stance between a tweet and its target.

The development process for this project has taken into consideration all the major phases of the model-building cycle, including data exploration, pre-processing, application of NLP practices, deep learning model building, parameter tuning and evaluation. Area of Transfer Learning was also explored to improve the performance.

At the initial phase, the data was explored to perform preprocessing with respect to Natural Language Processing practices. A base model was created to analyze the performance. Later, the final model is developed using context encoding layers. Hyper parameter tuning was performed to identify the combination of right parameters achieving the best performance. Different evaluation metrics were applied to understand the performance of the model over unseen dataset. As the final step, independent evaluation was performed on the set of tweets and targets that were fetched using Twitter API.

This document outlines the key decisions made during the project development including the literature review.

Literature Review

Nowadays, Social media platforms are widely used by most people for social interactions, making them the powerful information dissemination tools. There are several studies in the area of NLP that use microblogging platforms such as Twitter, to measure brand sentiment, to find specific trends or to understand the public opinion towards different social and political aspects.

In one such study [2], the major research conducted in the domain of stance detection is analyzed and discussed. It discusses the correlation and differences between stance and sentiment, various types of stance and various types of features (content based – linguistic and vocab of tweets and network based – user's metadata and behavioral data) that can be used for stance detection. Apart from this, it compares various algorithms from classical machine learning to deep learning and transfer learning. The comparison of approaches shows that transfer learning, which is the prominent direction of research at present, provides promising results. Moreover, the study summarizes many datasets available for stance detection. One such dataset is used for [4] stance detection contest that contains annotated tweets (stance and sentiment) across 6 targets. The contest has two tasks. For Task-A, the annotated tweet data across 5 targets is used, whereas, for Task B, no training data is provided, and data of 6th target is used for testing. MITRE [3] is among the most promising deep learning architecture mentioned in [4]. In this architecture, five different classifiers are trained using a similar approach. It uses tweets text and feeds it into a 256-dimensional embedding layer using one-hot representation followed by 128 LSTM units and 128 dimensional fully connected layer. The output layer contained 3 dimensional softmax layer with output units representing 3-classes of stance (that are, Favor, Against and None). For each classifier, the embedding and LSTM layers are initialized by pre-trained weights. For the first projection layer feature pre-training, the unlabelled additional tweets data is used along with word2vec skip-gram algorithm to learn features that are more related to task and the weights of this layer are further updated while backpropagation. For pre-training of LSTM layer, the weights are initialized with the help of auxiliary hashtag prediction task. For this auxiliary task, additional tweet data is gathered across top 197 hashtags. Each classifier is trained for 50 epochs along with early stopping. The model performance is evaluated using F1 score and the average F1 score on unseen test data is 67.8 on Favor and Against classes. The model developed in this study solves the issue of low amount of training data by using pre-trained feature weights and achieves the best result in the contest, however, it shows minor signs of overfitting and the most frequent stance class dominates the performance.

One more such study that uses the same dataset and has implemented stance detection is [5]. They have employed a two-stage architecture along with the concept of attention. Phase 1 focuses on finding subjectivity based upon the fact that the tweets with neutral stance are usually non-subjective. Using the retained subjective data of phase-1, in phase-2, polarity (favor/against) of the tweets is identified. In this architecture, there are mainly two components that are used similarly in both phases. A bi-directional LSTM layer that is used for feature encoding and attention mechanism that computes attention to each word in the text (concatenated tweet text embedding and average target embedding) by checking its relevance to the considered target topic. The model is evaluated using a macro F score of two classes (68.84 for favor and against classes), and average accuracy across 3-classes (60.24). The proposed model gives robust performance across all target topics and employing an attention layer is the crucial strength, however, the overall study gives a very brief explanation of experiment set-up making it hard to practice.

Now, as the context plays an important role in identifying the stance, one interesting model “CrossNet” [1], that deals with cross target stance classification is explored. The study uses the SemEval-2016 dataset and additional tweet data gathered related to mining. It focuses upon the general classifier between different target topics. The model architecture consists of four layers (that are, embedding layers, context encoding layer, aspect attention layer and prediction layer). There are two separate embedding layers for tweet text and descriptive target text and further separate encoding layers for both inputs. In context encoding layers, Bidirectional LSTM layers are used with the concept of conditional encoding that uses the dependency of target on tweet text and outputs the contextually encoded sequences. After this, the aspect attention layer is attached that identifies the domain specific features, that can be useful for cross target stance detection, followed by MLP prediction layer with softmax. For model evaluation F-score (average of F macro and F micro) is used for in-target performance and transfer ratio is used for cross-target performance. The average F-score across all targets is 59.4. The in-target performance of the model is not as promising as other studies; however, the major strength of this model is capturing central information for related domains by using attention layer. For instance, the features learned from the tweet text of ‘Climate Change is real Concern’ target are very useful in determining the stance for other related target ‘Mining’. The detailed experiment is described making it easy to practice.

Implementation Methodology

DATA EXPLORATION AND PREPARATION

Provided train dataset contained 2914 observations of 5 features including tweet, its target, stance, opinion towards and sentiment. ‘Opinion towards’ feature was discarded as it was irrelevant to this task.

- To have a better understanding of the data, a class frequency distribution was examined for both ‘Stance’ and ‘Target’ variables as shown in the figure1. As observed from the frequency distribution plot, the provided data was imbalanced in nature.

- Apart from this, to understand the dependency of sentiment over stance of a sentence, a chi--squared test of association is performed between Stance and Sentiment. As the p-value received is very small (p << 0.05), sentiments are also discarded for this task of classification.

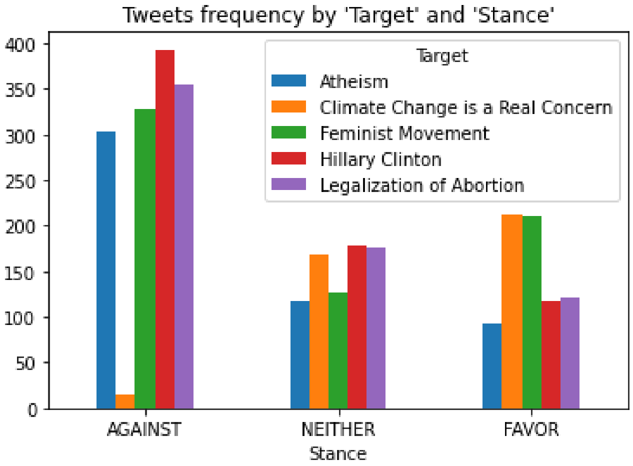


Figure 1 - Data Distribution Across targets

- The distribution data across stance classes and target topics are plotted. It shows that the 'Against' stance class has comparatively more data than the other two.

APPLYING NLP PRACTICES

The unstructured tweet text data was further cleaned and preprocessed to feed into the neural network. The cleaning steps such as removal of whitespace, special characters, digits, emoticons, punctuations and stopwords; and conversion of text in lowercase, were performed, followed by tokenizing the cleaned data and applying stemming to normalize the word distribution. The cleaned tweet data is also converted into bigrams to analyze the model performance on different corpus.

Each sequence of tweet corpus is converted into integer encoded representation and length of the sequences are normalized. The similar actions are performed on target topic data. The padded sequence data is further divided into train and validation sets using 20% split.

MODEL BUILDING, TUNING AND EVALUATION

The task of stance detection from tweet text across 5 different targets required a classifier model that captures the useful domain information by considering stance bearing sentence and target and generalizes well among all 3 stance classes. In this experiment, we developed a base model by considering only tweet text and further developed a more task specific model architecture.

As inferred from the literature, LSTM networks are predominantly used in the NLP applications of Deep learning domain due to minimal requirement of feature engineering and ability to find long and short-range dependencies in text data. Following this idea, we developed our base model with a bidirectional LSTM layer for the task of stance detection.

Sequential Base Model Creation & Analysis

Initially for the purpose of stance detection, a sequential model was created having one Embedding layer (with no pre-trained weights), one Bidirectional LSTM layer, and one output layer as shown in figure.

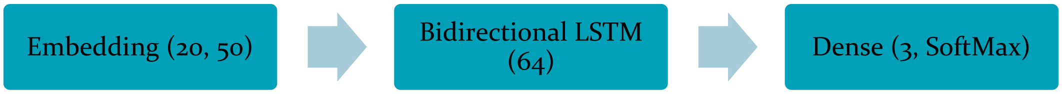


Figure 2 - Base Model Architecture

- The model was trained and validated using the provided test dataset to understand its behavior.
- Tweets were merged with their respective target topics and these merged data (with maximum length of 20) was provided as an input to the base model.
- 'Softmax' activation was applied to the output layer, as this belongs to the multi-class classification problem.
- Adam optimizer was used to compile the model as it converges faster.

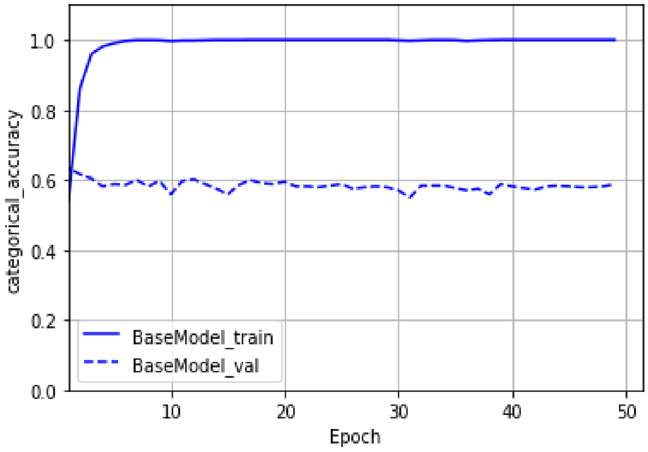


Figure 3 - Learning plots of Base Model

Now, as we can infer from the performance curve of the base model, by using only tweet data as an input, the model is not able to extract the important features for stance. Target topic is a crucial feature to provide contextual information about tweets. Therefore, the model with two inputs is developed by following the experiment setup from [1].

Functional Model Creation

The model architecture is as shown in the figure.

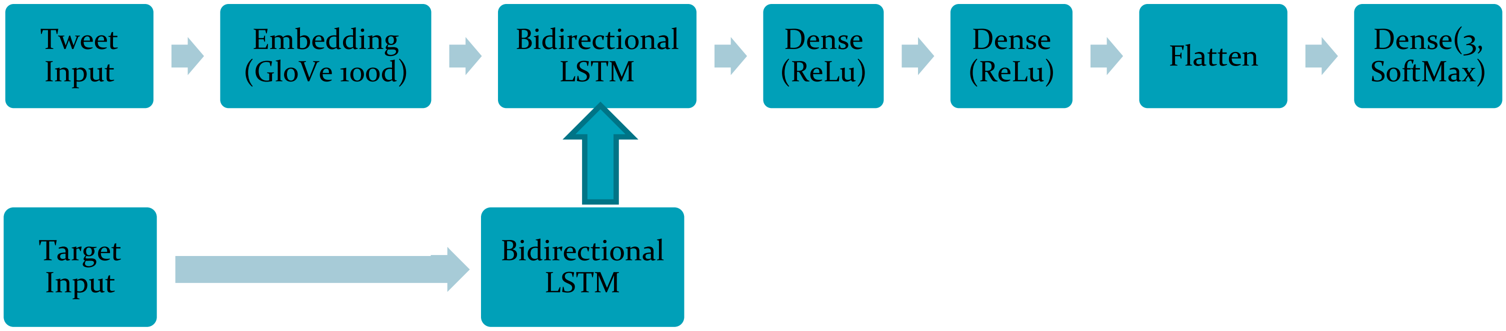


Figure 4 - Final Model Architecture

The first layer of model architecture consists of two embedding layers for stance bearing text data. Using 100d pre trained glove embedding vector, which is trained on twitter data, the dimensions are added to our tweet text vectors and it is further used in tweet embedding layer. The vector outputs of tweet embedding layer are fed into encoding layers. Bidirectional LSTM layers are used as an encoding layer. For the target encoding layer, the weights of forward and backward pass are captured and are further used as initial state inputs for tweet text encoding layer. This serves the purpose of passing contextual information (target information) in the tweet text encoding layer. Further, the fully connected layers are attached, followed by the output layer with 3 class nodes and softmax activation. To prevent overfitting, kernel regularizers and dropout layers are used. For fast convergence of gradient, 'Adam' optimizer is used with exponential decay of learning rate. The model is trained using 50 epochs and evaluated using categorical loss and categorical accuracy during training and validation phase. The model is trained using 85% of training and 15% of validation data.

Hyper Parameter Tuning

After a few manual experiments with a number of epochs, batch size and train-test split, other hyper parameters such as number of neurons, dropout rate and lambda value for L2-regularizer are fetched via Keras Random Search operation. The random search is performed over 20 trials each executing with at max 10 sets of different parameters. The model is fitted using the best parameters extracted.

The final stance detector model was achieved by having 60 neurons at each Bidirectional LSTM layer; 60 and 30 neurons respectively for the next two dense layers; dropout of 0.1 and L2-regularizer of 0.001. Default cross validation was used while performing random search for parameter tuning with keras-tuner.

Model Evaluation

For model evaluation, training and validation curves are plotted for categorical loss and categorical accuracy.

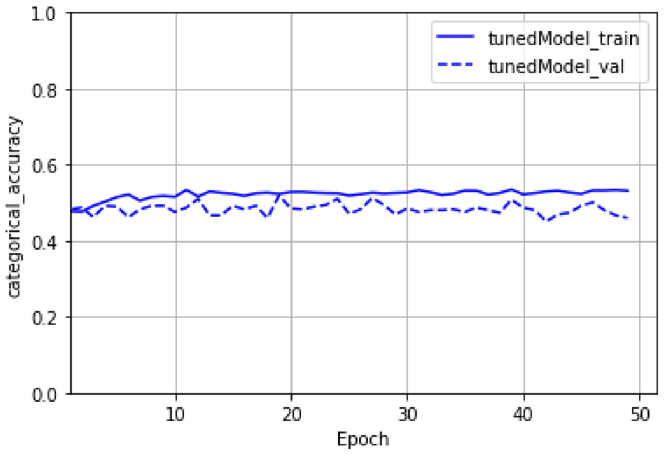


Figure 5- Learning curves with categorical accuracy

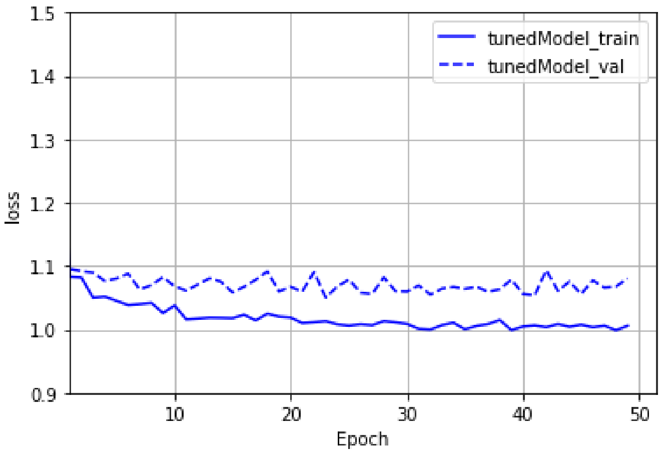
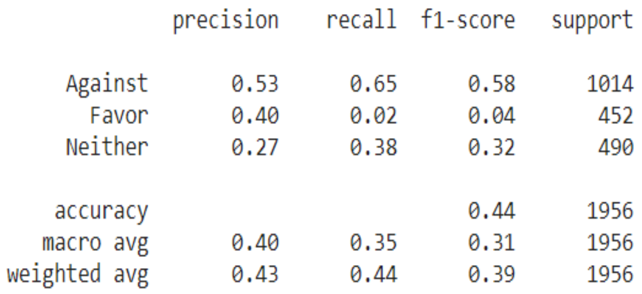


Figure 6 -Learning curves with categorical loss

As observed from the closely following training and validation curves, the model seems to be generalizing well in terms of bias and variance trade-off, however, the accuracy and loss curve show that the model is weakly learning the domain specific features.



```
              precision    recall  f1-score   support

     Against       0.53      0.65      0.58      1014
       Favor       0.40      0.02      0.04       452
     Neither       0.27      0.38      0.32       490

    accuracy                           0.44      1956
   macro avg       0.40      0.35      0.31      1956
weighted avg       0.43      0.44      0.39      1956
```

Figure 7 - Classification report on Unseen Data

To evaluate the model performance over unseen data, the provided test dataset was utilized. Classification matrix is displayed to understand the performance across 3 classes and as a final performance metric, F measure is used due to imbalance of target classes. The classification report shows that the model provides better performance for the class with more number of training samples and does not generalize well across all stance classes. Moreover, the model gives F measure of 0.31 over unseen data.

Independent Evaluation

For independent evaluation of this model, tweets containing hashtags of different target topics (5 same target topics given in train data and few others) were fetched using rest API. The data was labelled manually to evaluate the model performance against it. The F measure for this dataset is 0.51 for 'Against' , 0.50 for 'Neither' and 0 for favor stance class. This shows that model performance is close to weak learners and not able to generalize well for all stance classes. The classes with higher number of instances in training data are always achieving higher F measure.

Limitations and Future Work

The model developed in this experiment does not extract the important features for stance detection, therefore, showing very steady learning and validation curves. This can be further improvised by applying attention layers that extracts more task relevant information. Moreover, by using transfer learning, the models developed for similar tasks can be employed to enhance the performance. The individual classifiers for target domain can be developed to achieve more accuracy target type-wise. Due to limited time and scope constraint, steps mentioned here were not implemented, however they seem like a viable option to enhance the learning capability of the model.

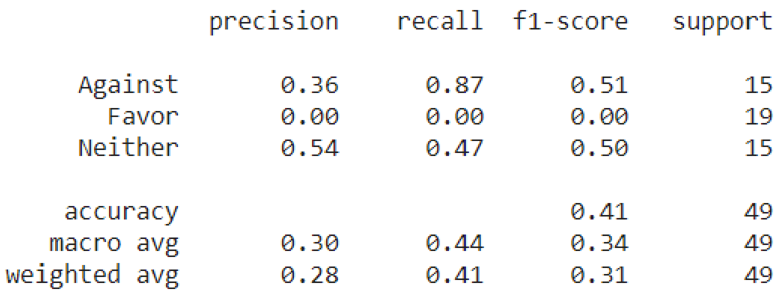
The experiment suffers from insufficient input data and model is notoriously hungry for training data. For this, as suggested in literatures, more unlabeled tweet data can be gathered, and semi supervised learning algorithm can be developed.

References

1. C. Xu, C. Paris, S. Nepal and R. Sparks, *Cross-Target Stance Classification with Self-Attention Networks*. Melbourne, Australia: Association for Computational Linguistics, 2018.
2. A. ALDAYEL and W. MAGDY, *Stance Detection on Social Media: State of the Art and Trends*. Social and Information Networks (cs.SI); Computation and Language (cs.CL), 2020.
3. G. Zarrella and A. Marsh, *MITRE at SemEval-2016 Task 6: Transfer Learning for Stance Detection*. San Diego, California: Association for Computational Linguistics, 2016.
4. "Task 6: Detecting Stance in Tweets < SemEval-2016 Task 6", *Alt.qcri.org*, 2020. [Online]. Available: http://alt.qcri.org/semeval2016/task6/. [Accessed: 10- Oct- 2020].
5. K. Dey, R. Shrivastva and S. Kaushik, "Twitter Stance Detection — A Subjectivity and Sentiment Polarity Inspired Two-Ph][ase Approach", Computation and Language (cs.CL); Information Retrieval (cs.IR); Social and Information Networks (cs.SI), 2017.
6. *Curiousily.com*, 2020. [Online]. Available: https://www.curiousily.com/posts/hackers-guide-to-hyperparameter-tuning/. [Accessed: 10- Oct- 2020].
7. "Natural Language Toolkit — NLTK 3.5 documentation", *Nltk.org*, 2020. [Online]. Available: https://www.nltk.org/. [Accessed: 10- Oct- 2020].

Appendix

1. Classification report of independent evaluation Data



```
              precision    recall  f1-score   support

     Against       0.36      0.87      0.51        15
       Favor       0.00      0.00      0.00        19
     Neither       0.54      0.47      0.50        15

    accuracy                           0.41        49
   macro avg       0.30      0.44      0.34        49
weighted avg       0.28      0.41      0.31        49
```