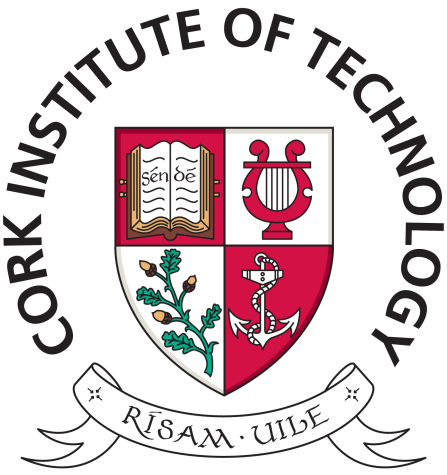


# Stance detection using Bi-directional LSTM model with Attention

by

Pankaj Singh

This proposal has been submitted in partial fulfillment for the module Computing Research and Practice.

in the
Faculty of Engineering and Science
Department of Computer Science

May 2019

# Declaration of Authorship

I, Pankaj Singh , declare that this proposal titled, 'Stance detection using Bi-directional LSTM model with Attention' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for an masters degree at Cork Institute of Technology.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this project report is entirely my own work.
- I have acknowledged all main sources of help.
- I understand that my project documentation may be stored in the library at CIT, and may be referenced by others in the future.

Signed: _____

Date: _____

CORK INSTITUTE OF TECHNOLOGY

# *Abstract*

Faculty of Engineering and Science
Department of Computer Science

Master of Science

by Pankaj Singh

Stance detection in any form of textual data has been an active area of research in the recent times within the context of natural language processing.The process of detecting stance generally determines the polarity of a piece of textual information towards a given target. This process has found widespread applications in textual entailment, information retrieval and various sub-modules of natural language processing (NLP).

One of the most interesting experiment of detecting stance was carried out by Augenstein et al. [1], the proposed model consists of bi-directional LSTM (Long Short Term Memory unit) layers, which is a variant of the traditional Recurrent neural network model with a forget gate embedded in it. The model achieves an F1 score of 0.4901 on SemEval 2016 Task 6 corpus for stance detection on twitter, and a F1 score of 0.5803 on automatically labelled tweets for the test targets. SemEval is a very popular competition for natural language processing enthusiasts, which is held every year and the topic for the year 2016 was stance detection. The proposed model achieves the best results to date as stated by Augenstein et al. [1] "In the weakly supervised seen target scenario, as considered by prior work, our approach achieves the best results to date on the SemEval Task B dataset" Augenstein et al. [1], making the model most ideally suited to be used as a baseline model.

The chapters to follow will give insights about the task of stance detection, the techniques which had been used till date, and the experiments that were been carried out to achieve improved F1 scores.

# *Acknowledgements*

I would like to thank Dr. Ruairi O Reilly, for his encouragement and invaluable advice. Finally i would like to thank my father Mr. Anand Singh and Mother Geeta Devi, who have supported me throughout the masters.. . .

# Contents

# List of Figures

# List of Tables

# Abbreviations

| | |
|---|---|
| **LSTM** | **L**ong **S**hort-term **M**emory |
| **CNN** | **C**onvolutional **N**eural **N**etwork |
| **RNN** | **R**ecurrent **N**eural **N**etwork |
| **SemEval** | **S**emantic **E**valuation |
| **NRC** | **N**ational **R**esearch **C**ouncil canada |

# Chapter 1

# Introduction

Stance detection can be defined as the task of evaluating the polarity of opinion of author of an article towards a target specified by a third party user. In this process the text of the article and the specified target plays a crucial role in determining the stance of the author as, in favour of, against or none with respect to the target.

Humans are capable of deducing whether the author of an article is likely in favour, or against a third party claim (or target). The goal of stance detection is to identify relevant bits of information in texts and the textual relationship that exists between them, and accordingly judge the polarity of this information gathered towards a target. A system is said to be efficient in detecting stance of an article, if it can identify relevant chunks of information that might not exist in the context of the text under focus, for instance, if a person is supporting foetus rights then it is more likely that he is against the abortion rights.

Stance detection is related to sentiment analysis on various levels, but there exists a thin line of difference between both the terminologies. For instance, in the process of sentiment analysis, the systems classify a chunk of text (or article) as positive, negative or neutral. Unlike sentiment analysis, stance detection is focussed on determining the polarity of the author towards a given claim (or target). There are possibilities that the target of the text might or might not be explicitly mentioned in the context of the text under focus. For example: -

**Target:** Barack Obama

**Tweet:** Jebb Bush is the only sane candidate for 2016.

In the above example it can be deduced that the tweeter is in favour of Jebb Bush, however there are probabilities of deducing that he is against Obama or in favour of both Obama and Jebb, considering the target as Barack Obama. This is where it becomes an interesting task of assigning a stance to the tweet for an explicit target or claim.

The process of stance detection has wide range of applications, some major application areas are: -

**i) Information retrieval:** The process of information retrieval involves extracting documents (usually texts) which are unstructured in nature, based on the information requirement mentioned. Exploiting stance detection systems in this area have shown significant boost up in the process of information retrieval.

**ii) Textual entailment:** Textual entailment refers to the task of recognizing if the meaning of a text can be inferred from another text that might contain useful chunks of information related to the text in focus. A very good example of textual entailment can be found in the QA (Question and answer) system, which must look out for textual entailments in question and answers to select the most appropriate answer.

**Question:** Who killed Jessica?

**Answer:** X killed Jessica

Here the murder of Jessica by person X is entailed by X killed Jessica. The process of stance detection helps in determining the textual relationship between two utterances, thus providing us insights over any sort of entailments that might exist between them.

**Evaluation metric to be used:**

**F1 score:** The accuracy achieved on test data can be calculated using the F1 score, which is defined as the harmonic average of the precision and recall.

$$\text{F1 score} = 2 \; . \; \frac{\text{precision} \; .\text{recall}}{\text{precision} + \text{recall}} \tag{1.1}$$

where,

$$precision = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \tag{1.2}$$

$$recall = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \tag{1.3}$$

- True positives: number of instances of the test data correctly classified under class 0.
- False positives: number of instances of the test data correctly classified under class 0.
- False negatives: number of instances of the test data incorrectly classified under class 1.
- True negatives: number of instances of the test data incorrectly classified under class 1.

## 1.1 Motivation

Stance detection as a domain has been under continuous exploration process in the recent years. Many exploratory works were done in the following areas:

- **Fake news detection:** Many experiments have shown the potential of stance detection in the area of fake news detection, the work of Baird et al. [8], Hanselowski et al. [9] Chaudhry et al. [10] are some of the most exceptional works in this area, also the fake news challenge which was an online competition organized by the efforts of many volunteers from the industries around the world, held in the academic year 2016-2017 recognized their work by awarding them the top 3 rankings in the competition.
- **Stance detection in tweets:** One of the most popular competition of SemEval (Semantic evaluation) which is organized by the Special Interest Group on the Lexicon of the Association for Computational Linguistics every year, had stance detection as the topic for the year 2016. The datasets released by them are still been used by many researchers for exploring the domain of stance detection.

A detailed study of the effects of twitter bots on distorting the US presidential elections of 2016 Bessi and Ferrara [11], gives us the motivation of detecting stance of an tweeter towards each candidate or towards the agenda they propose. This process can be useful in doing a comparative analysis of how distort the election results were when compared with the ground reality.

## 1.2 Research Question

The study would be focused on answering the following research questions:

- The measure of how effective Bi-Directional LSTM units with context aware attention are in determining the stance of the author of a tweet.
- Explore different configurations and models that might have significant effect on the stance detection process.

## 1.3 Document Structure

This document will have the follwoing structure:

- **Chapter 1: Introduction** This chapter will contain a brief introduction of the term ‘Stance Detection‘, and the evaluation metric to be used.
- **Chapter 2: Background** This chapter will describe the neural network models which had been deployed in this area before and a detailed architecture review of them.
- **Chapter 3: Design and Implementation** This chapter will present the design decisions taken in-order to build the proposed design and how the proposed design is to be implemented.
- **Chapter 4: Result and Evaluation** This chapter will present the results generated and a comprehensive evaluation of them.
- **Chapter 5: Conclusion and Future work** This chapter will present the conclusions drawn based on the results, a discussion about the problems faced and possible future work.

# Chapter 2

# Background

This chapter will comprise of background for the thesis. This will include an overview of the current models in use for stance detection, their working and a brief review of the models.

## 2.1 Background Research

### 2.1.1 Convolutional neural network (CNN)

The area of artificial neural networks has seen some significant developments in the recent years, in context of processing unstructured data, like images, text, and speech. The development of Convolutional neural networks (CNN) gave a boost to the work of dealing with these unstructured data, by extracting important features out of this unstructured data. CNNs are widely used in image classification, speech synthesis tasks, however in the recent years it has found application in many natural language processing tasks as well, for instance sentiment analysis, stance detection.

The most crucial operation in a CNN is the processing of data through the convolution operation. In this operation two signals are passed through a convolution to produce a third signal which might reveal some more information about the signals.

Using CNN's have provided improved results over the traditional feed forward neural networks because of its abulity to successfully capture the spatial and temporal dependencies in the given dataset.

#### 2.1.1.1 Components of Convolution neural network

- **Input layer:**

  This layer contains the vectorized matrix form of the input sentence which is obtained by mapping the individual words in a sentence to a corresponding pre-trained word embeddings matrix.

- **Convolution layer:**

  This layer is responsible for detecting patterns in a given text. It applies a convolution operation on the inputs received from the input layer, which in turn reveals some useful insights about the text under focus.

- **Activation function:**

  This function helps to map the resulting values obtained from the convolution layer to a given range depending on the type of activation function used. Hence helping in assigning a class to the input values. CNNs generally use a ReLU (Rectified Linear Unit) activation function because of its capability to add non-linearity in the network and providing non-saturating gradients for positive net inputs.

- **Pooling layer:**

  A pooling layer in CNN has the task of combining the outputs of the neuron clusters in the convolution layer to a single neuron in the next consequent layer. The two widely used methods for pooling are, max pooling: it utilizes only the maximum value from each cluster of the neurons in the convolution layer, average pooling: this utilizes the average value from each cluster of neurons in the convolution layer.

- **Fully connected layers:**

  In a fully connected layers terminology, every neuron in a layer is connected to every neuron in the consecutive layer. For classification purpose, the output neurons receive inputs from these fully connected layers.

#### 2.1.1.2 System designed using CNN for stance detection

The proposed architecture of convolution neural network by Wei et al. [2] was inspired from the architecture proposed by Kim [12], due to a similarity in the process of stance

detection and sentence classification. Kims model was designed for the sole task of sentence classification.

#### 2.1.1.3 Architecture and its working:

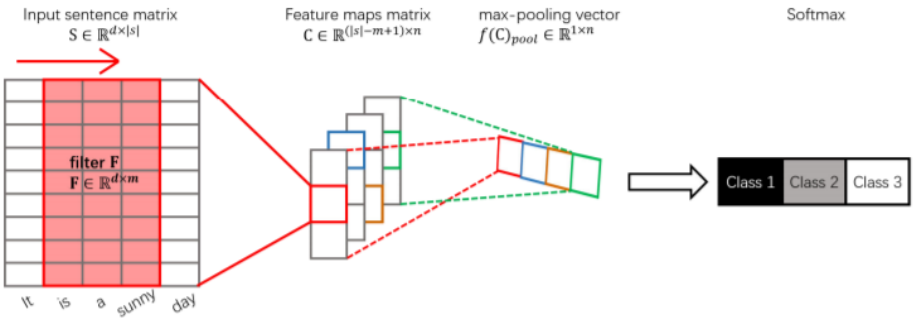


FIGURE 2.1: Architecture of a system designed for stance detection using convolution neural network. Wei et al. [2]

The proposed architecture has the following components:

- **Look-up table:**

  The look-up table is basically an enormous word embedding matrix, wherein each column which is d-dimensional in nature corresponds to a word in a sentence. The word embedding in the look-up table are pre-trained vectors from the word2vec team.

- **Input matrix:**

  The input matrix is a matrix of dimension (d * s), where s is the length of the input sentence. This matrix contains the d-dimensional vectorised representations of the words in the input sentence. These d-dimensional vectorised representations are obtained from the look-up table. If a word doesnt have a corresponding vectorised notation in the look-up table then it is either assigned as a zero vector of d-dimension or as a vector of numbers randomly generated in a given range.

- **Convolutional layer:**

  The major target of this convolutional layer is to extract patterns in a given sentence by simultaneously using a filter F which is sensitive to various patterns in a sentence. The output of this joint operation on a sentence is a feature vector

matrix C corresponding to that sentence. A n-dimensional bias vector is added to this feature vector matrix in-order to train a much efficient model.

- **Activation function:**

  A typical CNN model is traditionally followed by an activation function for a better fitting of the non-linear boundaries. The proposed architecture uses a ReLU (Rectified Linear) activation function which proved out to be more efficient than the sigmoid and tanh activation function in this use case scenario.

  The ReLU activation function is a very simple function which returns 1 for the derivative of any positive number and 0 for the derivative of any negative number.

  **ReLU(x) = max (0, x)**

- **Pooling layer:**

  The proposed architecture uses the max-pooling method in-order to reduce the complexity of the information generated as an output by the convolutional layer + ReLU activation function. This technique forms a condensed vector representation of the output generated, by choosing a maximum value from each column of the output of the ReLU activation function.

- **Output layer (SoftMax layer): -**

  The softmax layer is responsible for classifying the sentence or article to a class as declared by the user. This layer calculates the probability of each class and assigns the class label of highest priority to the input, hence we get the predicted label associated with the set of inputs.

### 2.1.2 Recurrent Neural networks(RNN)

The Recurrent neural networks (RNNs) were specifically designed to exploit the sequential information available in data. For this very reason RNNs have been widely used in natural language processing task, as there exists a sequential dependency between words in any given language.

A recurrent neural network has the capability to remember its past information and the things that it has learned over time by making relevant decisions. A basic feed-forward

neural network is capable of remembering things too, but only that which it has imbibed during the training phase of the networks.

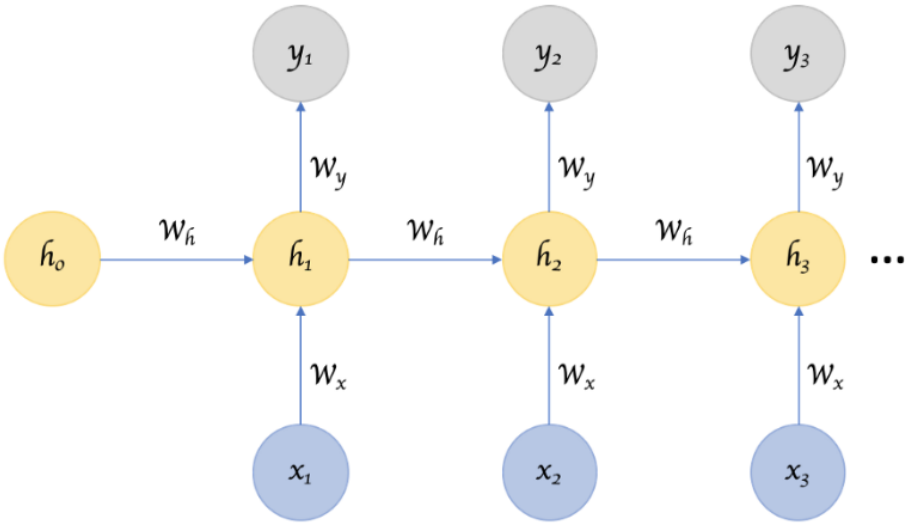


FIGURE 2.2: Architecture of recurrent neural network. Venkatachalam and Venkatachalam [3]

The RNNs remember things that were learnt from the previous inputs while it continuously forms outputs.

#### 2.1.2.1 Components of Convolution neural network

- The inputs x1, x2, , xn denotes the vectorized representation of input words at each time steps.
- The hidden state vectors h1, h2, , hn . The length of these vectors are defined by the user.
- iii) The weight matrices which connect the input states to the hidden states (Wx) and the weight matrices that connect hidden states to the output states (Wh)

In the case of multi-class classification problems, for instance predicting the next word, the output layer would be a huge softmax function of the number of words in the vocabulary.

#### 2.1.2.2 System designed using RNN for stance detection

The proposed system Zarrella and Marsh [4] employs a recurrent neural network that has been initialized from features that are been learned from external supervision on two unlabelled datasets. The sentence vector representations in this system were generated via pre-trained word embedding word2vec (skip-gram method). The features thus generated were used to learn sentence representations via hashtag prediction task. The vectors for each sentence were fine tuned for stance detection task.

#### 2.1.2.3 Components of this system

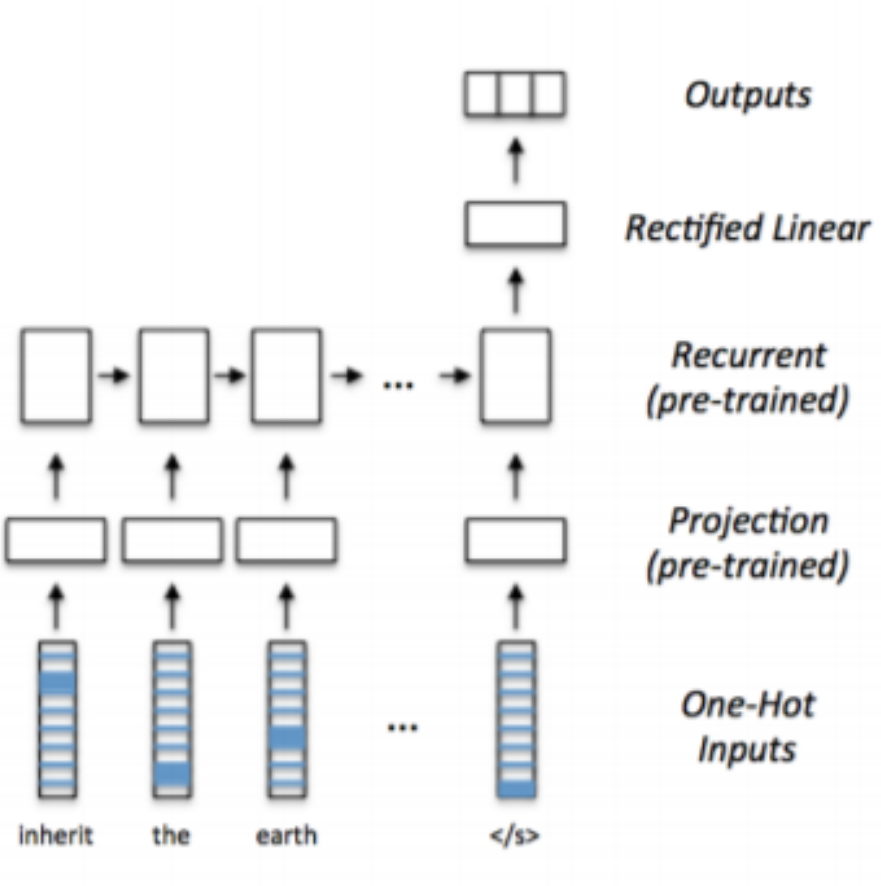


FIGURE 2.3: Architecture of a system designed for stance detection using recurrent neural network. Zarrella and Marsh [4]

- **Pre-training the projection layer:**

  Input tokens in this architecture were encoded in a one hot fashion, which gave a sparse binary vector representation of tokens.

The weights corresponding to the projection layer were initialized from a 256-dimensional word embedding which were learned using a word2vec skip-gram algorithm, this initialization capture the nuances like informal word usage in the tweets. These weights are tuned by backpropagation during the training of the RNN in a later stage.

- **Pre-training the Recurrent layer:**

  This layer consisted of 128 Long Short-Term Memory (LSTM) units. An initialization of weights associated with these units was carried out using the distance supervision of a hashtag prediction task. With this method the LSTM based network learned distributed sentence representations from the dataset rather than depending on the labels mentioned in the dataset. The network is trained using a gradient descent (AdaDelta) and categorical cross entropy minimization.

  Note In this process the word embeddings and recurrent network are fine tuned until the network reaches its maximum accuracy generation potential.

- **ReLU unit:**

  The output of this recurrent layer is then fed to a 128-dimensional ReLU (Rectified linear) unit , which are trained with a dropout of 90 percent. The last unit in this architecture consists of a three-dimensional softmax layer which classifies the stance of the tweets in one of the categories FAVOR, AGAINST, or NONE.

## 2.2 Long Short-Term Memory Networks(LSTM)

Long Short-Term Memory networks (LSTMs) are a special variety of the Recurrent neural networks (RNNs). LSTMs were first brought into picture by Hochreiter and Schmidhuber in the year 1997. These advanced versions of the recurrent neural networks were designed in-order to solve the long-term dependency problem in the RNNs. The LSTMs care capable of learning the distant dependencies of words in a given sentence, hence helping in preserving the correlation between distant words.

**Example of the long-term dependency problem: -**

**Sentence:** I spent 15 years working for under-privileged people in France. I then moved to India.

**Question:** I can speak fluent _____.

In this example the person has worked in France for 15 years, hence it is very likely that he knows French language. But in-order to make accurate predictions the RNN needs to remember this fact, this is where the Recurrent neural network fails.

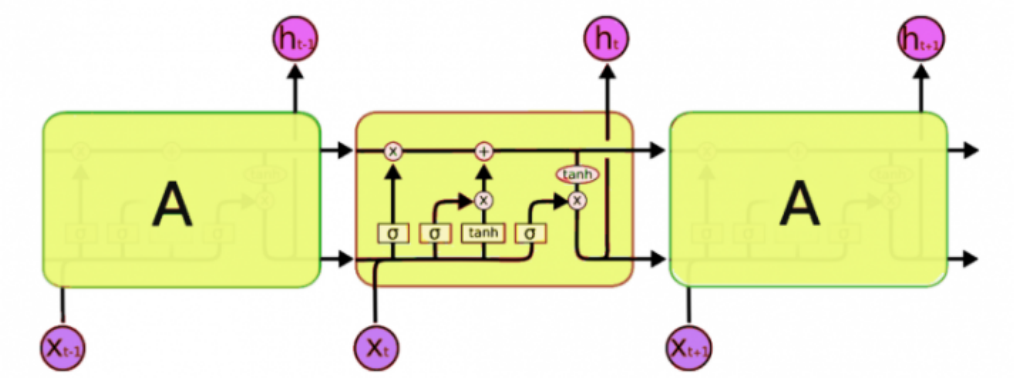


FIGURE 2.4: Architecture of Long Short-Term Memory Network.

#### 2.2.0.1 Components of a Long Short-Term Memory unit

- **Forget gate:**

  A forget gate is assigned the task of removing information from the cell state. Only the information that is not required or is of less importance in the direction of understanding the context of a text is removed by this gate.

  For example: Sam is a nice person. Jason on the other hand is evil.

  In this sentence the LSTMs at first assign the subject to be focussed as Sam. But as soon as they encounter a stop-word, they detect a possible change in subject and hence they forget the initial assignment of subject, hence vacating the space for another possible subject, which in this case is Jason.

- **Input gate:**

  The input gate is responsible for adding some new information into the cell state. This process takes place via a three-way process:

  - Filtering information: this involves the addition of relevant information to the cell state via a sigmoid function.

  - Creating a vectorized representation using tanh activation function, of all the possible values that can be added in the cell state.

- Performing vector multiplication of the value of the sigmoid function and the tanh function. This information is added by the input gate into the cell state.

- **Output gate:**

  This gate is assigned with the task to extract useful information from the current cell state and returning is as the output.

#### 2.2.0.2 Architecture and working of a system designed using LSTMs

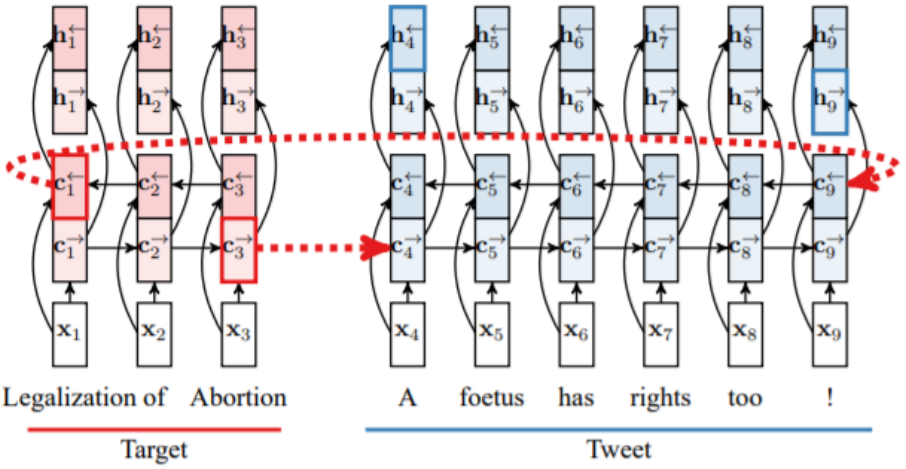


FIGURE 2.5: Architecture of bi-directional conditional encoded LSTM.Augenstein et al. [1]

- Pre-trained word embedding from Stanford GloVe dataset helps in capturing the token semantics.

- The architecture consists of 4 LSTM layers, with the input to each LSTM at every step being a 50-dimensional vector representation of the current token based on the GloVe word embeddings.

- The 4 LSTM layers in the proposed system are connected in the following manner:
  -

  LSTM (headline-forward) is connected to LSTM (article-forward).

  LSTM (headline-backward) is connected to LSTM (article-backward), these layers are supplied with same inputs as the forward LSTMs but in a reverse order.

- In the final step the outputs generated by LSTM (Article-forward) and LSTM (Article-backward) are averaged and fed to a softmax layer, which predicts the stance of the article in question.

# Chapter 3

# Design and Implementation

The motive of this chapter is to provide an overview of the various component architectures and their configurations and how they were involved in the implementation of experiments.

## 3.1 Problem Definition

A major component of this study investigates on how effective the Bi-directional LSTM units are in terms of determining the stance of the author of an article towards a specified target. The intent is to leverage the bi-directional nature of LSTM units to learn stronger textual relationships in sentences.

## 3.2 Data

### 3.2.1 A Dataset for stance detection

The dataset of SemEval-2016 competition for stance detection was selected for training and testing the proposed model. It was introduced in the competition by Mohammad et al. [5] and has been approved by the NRC Research Ethics Board (NRC-REB) as suitable for research purpose under the protocol number 2015-08.

The stance dataset in Figure 3.1 has the following properties:

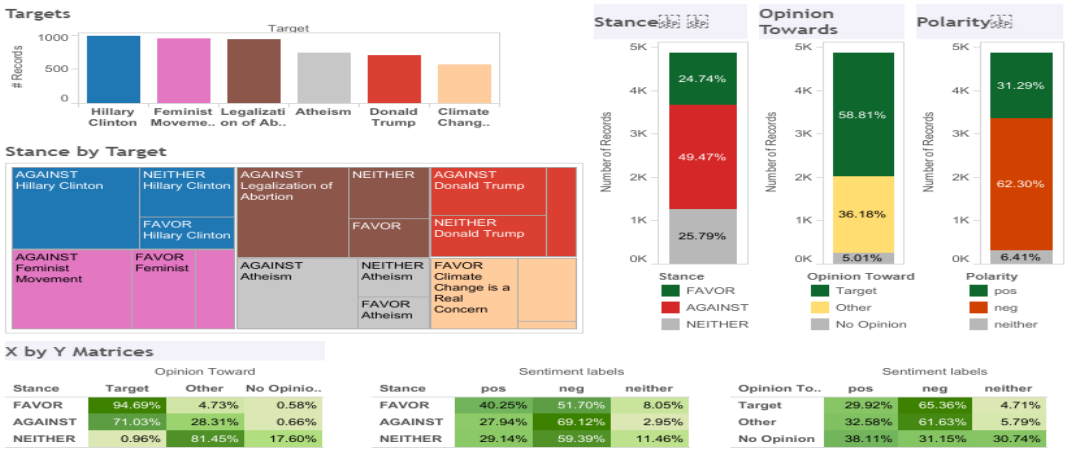


FIGURE 3.1: SemEval-2016 stance dataset visualization. Mohammad et al. [5]

- The dataset comprised of 6 targets, which were a compiled list of entities that were commonly known in the United States: Atheism, Climate change is a real concern, Donald Trump, Feminist Movement, Hillary Clinton and Legalization of abortion.
- Every target had a list of tweets and an annotated stance for the individual tweets attached with it.
- A total of 4163 instances were available in the dataset, out of which 2914 were available for training purpose and 1249 for test.
- The training instances had about 25.8% of tweets in favor of their targets, 47.9% against their targets and 26.3% falling in neither of the categories.
- The test instances had about 23.1% of tweets in favor of their targets, 51.8% against their targets and 25.1% falling in neither of the categories.
- The dataset comprises of 5 columns: Target, Tweet, Stance, Opinion towards and sentiment. A layout for the dataset can be seen in the figure 3.2

### 3.2.2 Pre-processing module

The dataset of SemEval-2016 competition for stance detection was selected for training and testing the proposed model. It was introduced in the competition by Mohammad et al. [5] and has been approved by the NRC Research Ethics Board (NRC-REB) as suitable for research purpose under the protocol number 2015-08. This dataset needs to be pre-processed before it can be used as a base for training or testing the proposed

| Tweet | Target | Train/Te.. | Stance | Opinion T.. | Sentim.. |
|---|---|---|---|---|---|
| If abortion is not wrong, then nothing is wrong. Powerful words from Blessed Mother.. | Legalization o.. | Train | AGAINST | Target | pos |
| Mary, Help of Christians persecuted everywhere, pray for us! #HolyLove #UnitedHear.. | Legalization o.. | Train | AGAINST | Other | pos |
| TY Michael @ASavageNation "I'd do anything to help him; he's right, he's telling the tr.. | Donald Trump | Test | FAVOR | Target | pos |
| 1 Cor 15:58 ...stand firm...Always give yourselves fully to the work of the #Lord...your l.. | Atheism | Train | AGAINST | Other | pos |
| 1 Corinthians 13:13 "So now faith, hope, and love abide, these three; but the greatest .. | Atheism | Train | AGAINST | Target | pos |
| 1 Holy Mary Mother of God, pray for us sinners now and at the hour of our death. Ame.. | Atheism | Train | AGAINST | Target | neg |
| 1 thing I learned from my job: doors leading 2 opportunity have a cover fee only the #p.. | Feminist Mov.. | Test | AGAINST | Other | neg |
| 1.2% of abortions happen at 21 weeks+ (these are only exceptions) #letstalkabortion #.. | Legalization o.. | Test | NEITHER | No Opinion | neither |
| 1/5"And the heaven We created with might, and indeed We are (its) expander." (Quran .. | Atheism | Train | AGAINST | Other | pos |
| 1of 2 #Never be #afraid to and never #apologise for using the #word of #GOD as #anal.. | Atheism | Test | AGAINST | Target | pos |
| 2 million bogus followers on Twitter @HillaryClinton #WhyImNotVotingForHillary | Hillary Clinton | Train | AGAINST | Target | neg |

FIGURE 3.2: A layout of the SemEval-2016 dataset. Mohammad et al. [5]

model in-order to make sure that the model learns stronger textual relationships in the given input.

Following steps were been taken in-order to make the dataset suitable for the model:

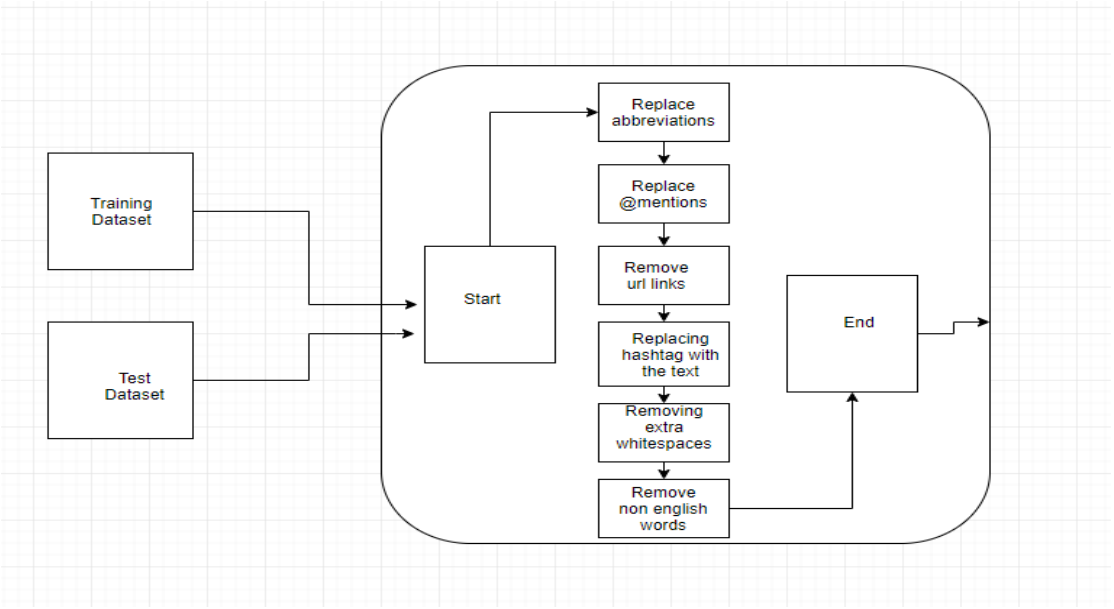


FIGURE 3.3: Pre-processing steps on SemEval 2016 dataset.

- **Replace abbreviations:** It has been observed that the tweets data contain abbreviated words that might have significant impact during the learning process of the model. To tackle this issue, the pre-processing module is provided with the most commonly used word abbreviations and their extended version. The module replaces the abbreviated word with its corresponding expanded version.

**examples of word abbreviations:** AFAIK: As Far As I Know, AFK: Away From Keyboard, ASAP: As Soon As Possible.

- **Replace @mentions:** @mentions in tweets are generally done in-order to point towards a entity which might be related to the context of the tweet. Loosing such valuable information might effect the performance of the stance detection model.The best idea is to keep the word and just remove the "@" character from it.

- **Remove URL links:** The URL links which might be found in tweets doesn't add value to the stance detection model, Therefore they can be ignored.

- **Replacing hashtags with the text:** Majority of the tweets data contain hashtags that might point towards a entity or an event. This information is very useful and needs to be preserved. Replacing the hashtags with the textual information that it carries might add some value to the stance detection model.

  **examples of hashtags:** #Trump, #GameOfThrones

- **Removing extra whitespaces:** The tweets data might contain extra whitespaces which does not carry any information. Therefore the extra whitespaces can be removed, leaving behind the informative text.

- **Remove non-english words:** Tweets might comprise of special symbols or non-english words which might have some information attached to it. This information can be extracted using neural machine translation techniques, but for the time being the non-english words are omitted.

## 3.3 Proposed Architecture

The proposed architecture, as depicted in figure 3.4, demonstrates the various components involved in the architecture and their work flow.

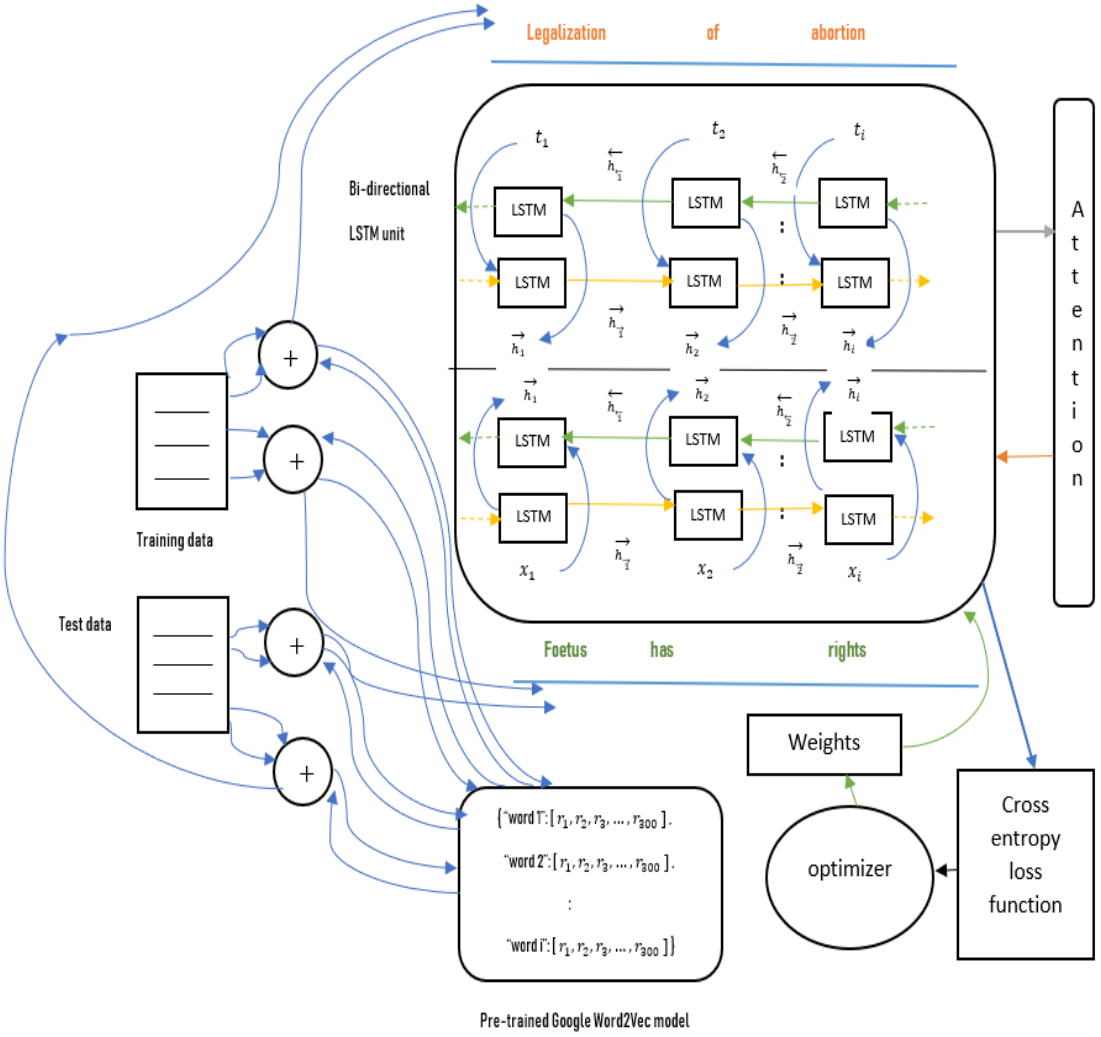


FIGURE 3.4: High-level architecture of the proposed model.

Proposed architecture has the following components:

- **Word-vector mapping:** A pre-trained Google word2vec word embedding model has been selected for producing vectorized representation of words for both target $(t_1, t_2, ..., t_i)$ and the tweet sentences $(x_1, x_2, ..., x_i)$. A detailed description of the word2vec model can be found in the section to follow.

- **Bi-directional LSTM:** In figure 3.4 Two LSTM units which are bi-directional in nature are been deployed separately on the target-tweet pairs for learning textual

relationships within the context of the sentence. Each Bi-LSTM unit gets two separate vectorized inputs $(x_1, x_2, ..., x_i)$ and $(t_1, t_2, ..., t_i)$ that represents tweet and target vectors respectively. Each Bi-LSTM unit learns the forward or reverse representation of words in the sentence through the word vectors.

- **Attention model:** A context aware attention model has been used between the two bi-LSTM layers. This is done in order to provide the LSTM layer operating on tweets with some context which was learned from the LSTM layer over the target sentences.

- **Cross entropy loss function:** This loss function is a combination of a softmax activation function and a cross-entropy loss function. It is used in special cases of Multi-class classification problems wherein the labels are one-hot encoded. The softmax layer in figure 3.5 receives the output of the Bi-LSTM network as inputs which is then converted to class probabilities. These class probabilities are feeded to a cross-entropy loss function which calculates the logarithmic loss value across the probability classes.

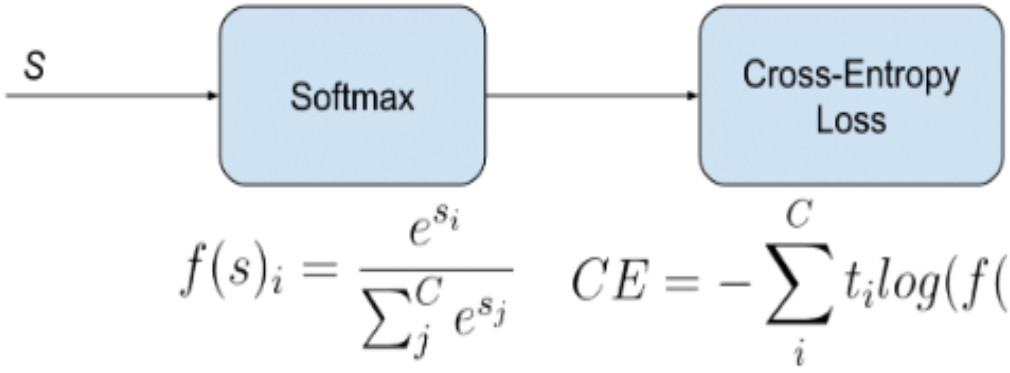


FIGURE 3.5: Cross entropy loss function Gmez [6]

- **Optimizer:** The proposed model uses a stochastic gradient optimizer (SGD) for performing an incremental gradient descent to reduce the loss. It is an iterative method for optimizing a differentiable objective function.

## 3.4 Word-vector mapping

The words in the target-tweet pairs needs to be converted in the form of vectors before feeding it to the bi-LSTM units.This word to vector conversion can be achieved with a

pre-trained Google word2vec model Mikolov et al. [7], which contains words and their vectorized representations.

The core idea behind developing a word2vec model was the assumption that the meaning of a word is affected by other words in a sentence. The more related are any two words in terms of their meaning, the more is the similarity between their corresponding vectors. Figure 3.6 depicts how strong the model is in determining the relationships between two words implicitly.

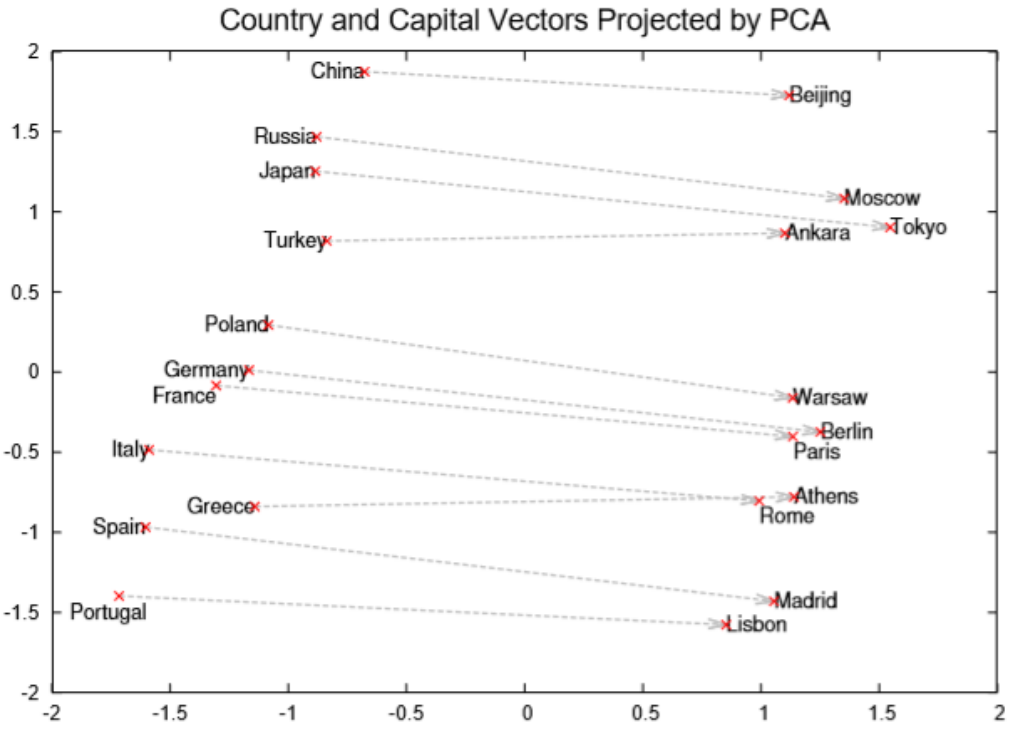


FIGURE 3.6: The figure demonstrates the ability of the word2vec model to organize concepts and learn implicitly the relationships that exists between them. Mikolov et al. [7]

In Figure 3.2 It can be observed that both target and tweet sentences derive their vectorized representation from the word2vec model, these vectors are given as inputs ($x_1$, $x_2$,..., $x_i$) and ($t_1$, $t_2$,..., $t_i$) to the two bi-LSTM units.

## 3.5 Bi-directional LSTM model with attention:

Recurrent neural networks with long short-term units, Hochreiter and Schmidhuber [13] have proved out to successful in many NLP tasks such as neural machine translation Mikolov et al. [7], language modelling Zaremba et al. [14] etc.

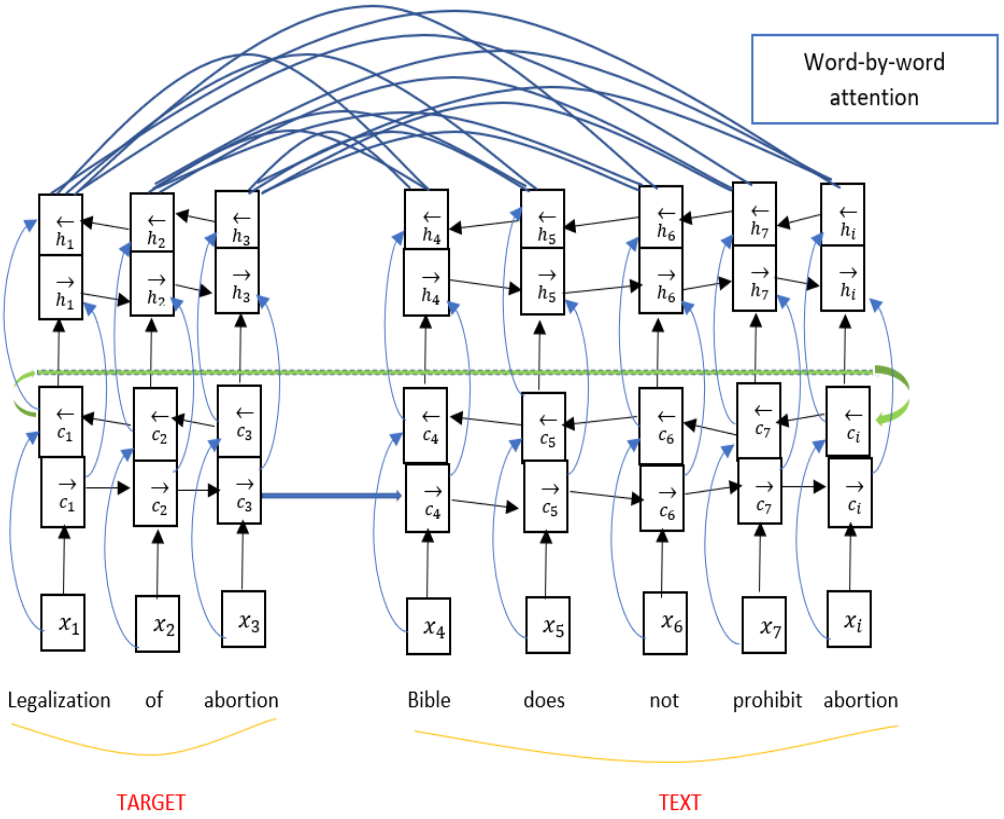


FIGURE 3.7: Bi-LSTM with context aware attention.

LSTMs have memory cells that can persist information for longer period of time. The flow of information in the LSTM cells are controlled by three gates: input gate, forget gate and output gate.

They can be used for recognizing textual entailment's by independently encoding target and tweet data as dense vectors and concatenating their input to a MLP (Multi-layered perceptron) classifier Bowman et al. [15].

In figure 3.7, $x_i$ denotes the individual vectorized representation of the words which are been referenced from pre-trained glove word embeddings. These word vectors of tweets

and target are provided to two different Bi-directional LSTM units. The memory state of the LSTM cells are denoted by $c_i$.

The memory units of the tweets are initialized by the memory state of the last LSTM unit of the target LSTM block. The following process is repeated but with the reverse encoded version of the tweet representations. This bi-directional encoded learning in LSTM allows the model to learn much stronger target-dependant relationships such that if a word is considered then both the left and the right context of the word are taken into account.

**Context aware Attention with LSTM:** An LSTM with attention network for recognizing textual entailment is very efficient in producing output vectors and accumulating representations of the target sentence in the cell state , which later on informs the second LSTM about which output vectors of the target it should attend over inorder to determine the stance of the tweet. The traditional approach of attention networks lacks in the quantity of contextual information

# Chapter 4

# Result and Evaluation

In this chapter results of the experiments performed are presented and analyzed. The first two sections represent different configurations and architectures that have been experimented and the last section will consist of the evaluation of the results.

## 4.1 Bi-Directional LSTM with attention

A Bi-Directional LSTM model with the configuration as depicted in figure 4.1 was tested on the SemEval 2016 Task-A test dataset.

### 4.1.1 Model configuration

The model consists of the following layers:

- 2 Input layers which represent the target and tweet sentences.
- 2 LSTM layers consisting of LSTM units equals to the the size of the word embedding dimension. One of the LSTM layer is for the encoding the target in forward direction and the other is for encoding the target in backward direction.
- 1 concatenate layer for merging the output of the 2 LSTM units for the target.
- 2 LSTM layers consisting of LSTM units equals to the the size of the word embedding dimension for the tweets. One of the LSTM layer is for the encoding the

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_1 (InputLayer) | (32, 50, 300) | 0 | |
| input_2 (InputLayer) | (32, 50, 300) | 0 | |
| lstm (LSTM) | [(32, 50, 300), (32, | 721200 | input_1[0][0] |
| lstm_1 (LSTM) | [(32, 50, 300), (32, | 721200 | input_2[0][0] |
| lstm_2 (LSTM) | [(32, 50, 300), (32, | 721200 | input_2[0][0]<br>lstm[0][1]<br>lstm[0][2] |
| lstm_3 (LSTM) | [(32, 50, 300), (32, | 721200 | input_2[0][0]<br>lstm_1[0][1]<br>lstm_1[0][2] |
| concatenate (Concatenate) | (32, 50, 600) | 0 | lstm[0][0]<br>lstm_1[0][0] |
| concatenate_1 (Concatenate) | (32, 50, 600) | 0 | lstm_2[0][0]<br>lstm_3[0][0] |
| time_distributed (TimeDistribut | (32, 50, 300) | 180300 | concatenate[0][0] |
| time_distributed_1 (TimeDistrib | (32, 50, 300) | 180300 | concatenate_1[0][0] |
| attention_with_context (Attenti | (32, 300) | 90600 | time_distributed[0][0] |
| attention_with_context_1 (Atten | (32, 300) | 90600 | time_distributed_1[0][0] |
| concatenate_2 (Concatenate) | (32, 600) | 0 | attention_with_context[0][0]<br>attention_with_context_1[0][0] |
| flatten (Flatten) | (32, 600) | 0 | concatenate_2[0][0] |
| dense_2 (Dense) | (32, 100) | 60100 | flatten[0][0] |
| dense_3 (Dense) | (32, 3) | 303 | dense_2[0][0] |

Total params: 3,487,003
Trainable params: 3,487,003
Non-trainable params: 0

FIGURE 4.1: Model summary for the Bi-Directional LSTM with attention.

tweets in forward direction and the other is for encoding the tweets in backward direction.

- 1 concatenate layer for merging the output of the 2 LSTM units for the tweet.
- 2 Time distributed layers one for each target and tweet sentences.
- 2 context aware attention layers.
- 1 concatenate layer to merge the output sequences generated from the attention units.
- A multi-layered perceptron unit with 100 neurons in the first hidden layer and a softmax function in the output layer.
- stochastic gradient descent optimizer with a learning rate of 0.01.

- categorical cross entropy loss calculation.

### 4.1.2 Result

| Number of epochs | Training accuracy | Training loss | Validation loss | Validation accuracy |
|---|---|---|---|---|
| 10 | 0.47 | 1.0571 | 1.0115 | 0.5721 |

TABLE 4.1: Results obtained by testing the model on SemEval 2016 Task-A test dataset.

The model generated an overall validation accuracy of **0.5721** on the test dataset.

It was observed that the model had a consistent training and validation accuracies after the very first epoch of the training and validation process, which can be seen in figure 4.2.

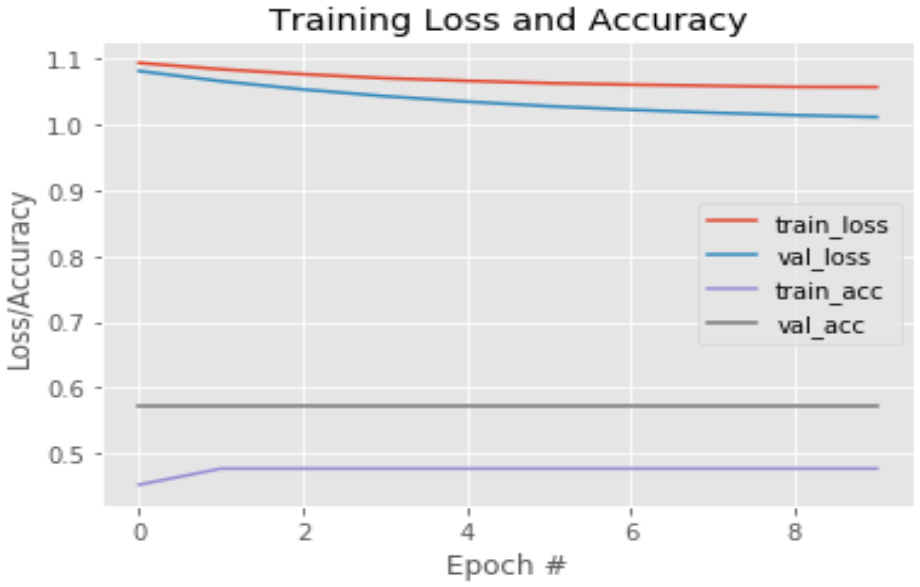


FIGURE 4.2: Visualization for the result generated.

Also a very slow rate of decrease can be noticed in the training and the validation losses in figure 4.2 across the time frame of 10 epochs.

## 4.2 Convolutional Neural Networks

The effectiveness of the CNN's in detecting relevant features in data make them a perfect choice for the purpose of stance detection.

### 4.2.1 Configurations

Two different configurations of the convolutional neural networks were tested on the SemEval 2016 Task-A dataset.

**Configuration 1:**

A detailed visualization for the configuration can be seen in figure 4.3

- Two input layers, one for the target sentence and the other for the tweets.
- One convolution layer and one max-pooling layer for the target.
- One convolution layer and one max-pooling layer for the tweets.
- One concatenate layer to merge the feature sequences generated by the target and tweet max-pooling layers.

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_1 (InputLayer) | (None, 50, 300) | 0 | |
| input_2 (InputLayer) | (None, 50, 300) | 0 | |
| conv1d (Conv1D) | (None, 49, 300) | 180300 | input_1[0][0] |
| conv1d_1 (Conv1D) | (None, 49, 300) | 180300 | input_2[0][0] |
| max_pooling1d (MaxPooling1D) | (None, 24, 300) | 0 | conv1d[0][0] |
| max_pooling1d_1 (MaxPooling1D) | (None, 24, 300) | 0 | conv1d_1[0][0] |
| concatenate (Concatenate) | (None, 24, 600) | 0 | max_pooling1d[0][0]<br>max_pooling1d_1[0][0] |
| flatten (Flatten) | (None, 14400) | 0 | concatenate[0][0] |
| dense (Dense) | (None, 300) | 4320300 | flatten[0][0] |
| dropout (Dropout) | (None, 300) | 0 | dense[0][0] |
| dense_1 (Dense) | (None, 100) | 30100 | dropout[0][0] |
| dropout_1 (Dropout) | (None, 100) | 0 | dense_1[0][0] |
| dense_2 (Dense) | (None, 3) | 303 | dropout_1[0][0] |

Total params: 4,711,303
Trainable params: 4,711,303
Non-trainable params: 0

FIGURE 4.3: Model summary for configuration 1.

- A multi-layered perceptron consisting of 2 hidden units. The first hidden unit consists of 300 neurons while the second hidden unit has 100 neurons. This is followed by a softmax layer which annotates stance to the intsances in the dataset.

**Configuration 2:**

A detailed visualization for the configuration can be seen in figure 4.4

- Two input layers, one for the target sentence and the other for the tweets.
- Two convolution layer and one max-pooling layer for the target.
- Two convolution layer and one max-pooling layer for the tweets.
- One concatenate layer to merge the feature sequences generated by the target and tweet max-pooling layers.

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_1 (InputLayer) | (None, 50, 300) | 0 | |
| input_2 (InputLayer) | (None, 50, 300) | 0 | |
| conv1d (Conv1D) | (None, 49, 300) | 180300 | input_1[0][0] |
| conv1d_2 (Conv1D) | (None, 49, 300) | 180300 | input_2[0][0] |
| conv1d_1 (Conv1D) | (None, 48, 300) | 180300 | conv1d[0][0] |
| conv1d_3 (Conv1D) | (None, 48, 300) | 180300 | conv1d_2[0][0] |
| max_pooling1d (MaxPooling1D) | (None, 24, 300) | 0 | conv1d_1[0][0] |
| max_pooling1d_1 (MaxPooling1D) | (None, 24, 300) | 0 | conv1d_3[0][0] |
| concatenate (Concatenate) | (None, 24, 600) | 0 | max_pooling1d[0][0]<br>max_pooling1d_1[0][0] |
| flatten (Flatten) | (None, 14400) | 0 | concatenate[0][0] |
| dense (Dense) | (None, 300) | 4320300 | flatten[0][0] |
| dropout (Dropout) | (None, 300) | 0 | dense[0][0] |
| dense_1 (Dense) | (None, 100) | 30100 | dropout[0][0] |
| dropout_1 (Dropout) | (None, 100) | 0 | dense_1[0][0] |
| dense_2 (Dense) | (None, 3) | 303 | dropout_1[0][0] |

Total params: 5,071,903
Trainable params: 5,071,903
Non-trainable params: 0

FIGURE 4.4: Model summary for configuration 2.

- A multi-layered perceptron consisting of 2 hidden units. The first hidden unit consists of 300 neurons while the second hidden unit has 100 neurons. This is followed by a softmax layer which annotates stance to the intsances in the dataset.

### 4.2.2 Results

The results for the two configurations can be found in Table 4.2

| Number of epochs | Training accuracy | Training loss | Validation loss | Validation accuracy |
|---|---|---|---|---|
| 20 | 0.5430 | 0.9947 | 0.9204 | 0.6621 |
| 30 | 0.5483 | 0.9944 | 0.9186 | 0.6589 |

TABLE 4.2: Results obtained by testing the model on SemEval 2016 Task-A test dataset.

It can be observed from the readings in table 4.2, the training accuracy for the CNN model for configuration 1 is **0.5430** and for configuration 2 is **0.5483**. It was also observed that the CNN models for both the configurations gave stable validation and training accuracy curves after the 15th epoch.This can be seen in figure 4.5 and figure 4.6.

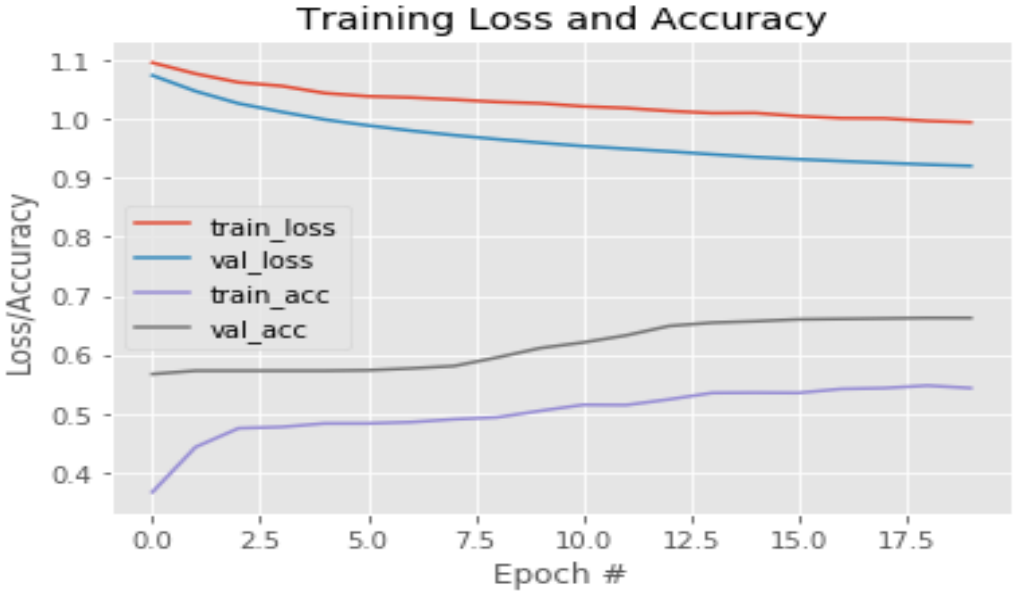


FIGURE 4.5: Visualization for the result generated for configuration 1.

The final validation accuracies recorded over the test dataset for configuration 1 was **0.6621** and for configuration 2 it was **0.6589**. This can be observed in table 4.2.

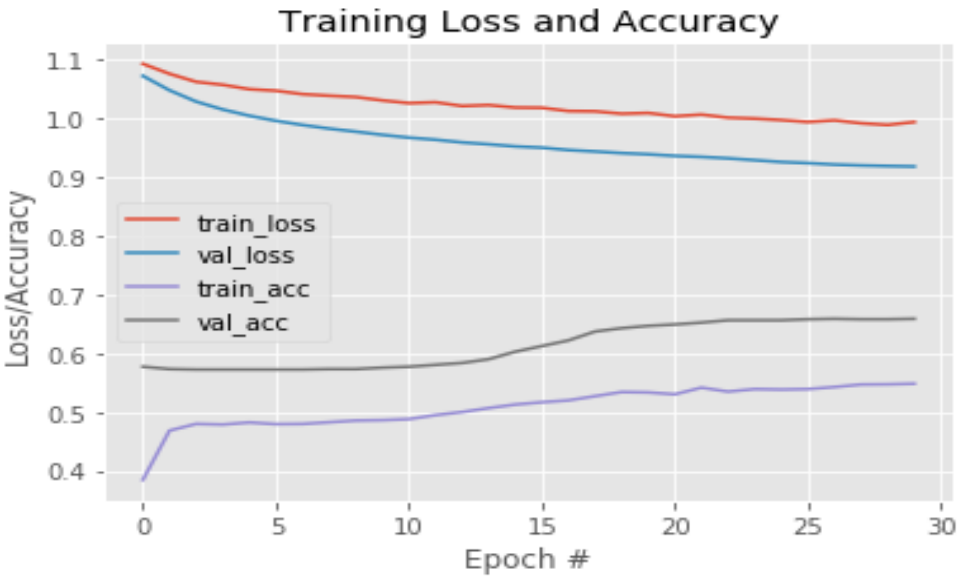


FIGURE 4.6: Visualization for the result generated for configuration 2.

The model generated for both the configurations shows no signs of overfitting, this phenomena can be observed by comparing the training and validation loss curves in figure 4.5 and figure 4.6, which do not overlap in both the configurations.

## 4.3 Comparative Analysis

It can be observed by comparing the table 4.1 and table 4.2 that the CNN model yields better train and validation accuracies than the Bi-directional LSTM model.

| Model name | macro F1-score | micro F1-score |
|---|---|---|
| Bi-LSTM with attention | 0.2426 | 0.5721 |
| CNN (config 1) | 0.4348 | 0.6621 |
| CNN (config 2) | 0.4461 | 0.6589 |

TABLE 4.3: F1-score comparisons for Bi-LSTM model and CNN model.

The F1-scores generated for all the configurations can be seen in table 4.3, On comparing the F1-scores it can be observed that the CNN model with a macro F1-score of **0.4461** and **0.4348** clearly out performs the Bi-LSTM model which has a macro F1-score of **0.2426**.

The results when compared with the model proposed by Augenstein et al. [1] depict that the proposed model under-performed.

# Chapter 5

# Conclusions and Future Work

## 5.1 Discussion

There were a series of problem which were encountered while making the Bi-conditional LSTM with attention model. The problems encountered are enlisted below:

- The stateful behaviour of the LSTM units which can be set by stateful=True parameter requires certain fixed batch size for the inputs. The batch size should be the highest common factorial of the number of instances in the training dataset and the number of instances in the test dataset. This results in a very slow and under-performing training process.
- There is only one certified tweet dataset available for stance detection, SemEval 2016. The datapoints in this dataset are very less and hence that explains the under-performing nature of the Bi-LSTM model.

## 5.2 Conclusion

The experiments performed on the SemEval 2016 dataset Mohammad et al. [5] depict how efficient the convolutional neural networks are in detecting textual relationships on comparing it with the proposed Bi-Directional LSTM unit with attention model. The CNN model yields an exceptional validation accuracy score of **0.6621** on the SemEval 2016 Task A test dataset.

## 5.3 Future Work

The goal of this study was to determine how efficient the Bi-directional LSTM models are in the process of stance classification. The challenges enlisted in the discussion section 5.1 restrain the model to achieve good performance. Hence for the future work it will be investigated on how to overcome these challenges and achieve a better performing model configuration.

# Bibliography


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

[2] W. Wei, X. Zhang, X. Liu, W. Chen, and T. Wang, "pkudblab at semeval-2016 task 6: A specific convolutional neural network system for effective stance detection," in *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 2016, pp. 384–388.

[3] M. Venkatachalam and M. Venkatachalam, "Recurrent neural networks," Mar 2019. [Online]. Available: https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce

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

[5] S. M. Mohammad, P. Sobhani, and S. Kiritchenko, "Stance and sentiment in tweets," *Special Section of the ACM Transactions on Internet Technology on Argumentation in Social Media*, vol. 17, no. 3, 2017.

[6] R. Gmez, May 2018. [Online]. Available: https://gombru.github.io/2018/05/23/cross_entropy_loss/

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[8] S. Baird, D. Sibley, and Y. Pan, "Talos targets disinformation with fake news challenge victory," *Fake News Challenge*, 2017.

[9] A. Hanselowski, A. PVS, B. Schiller, F. Caspelherr, D. Chaudhuri, C. M. Meyer, and I. Gurevych, "A retrospective analysis of the fake news challenge stance detection task," *arXiv preprint arXiv:1806.05180*, 2018.

[10] A. K. Chaudhry, D. Baker, and P. Thun-Hohenstein, "Stance detection for the fake news challenge: identifying textual relationships with deep neural nets," *CS224n: Natural Language Processing with Deep Learning*, 2017.

[11] A. Bessi and E. Ferrara, "Social bots distort the 2016 us presidential election online discussion," 2016.

[12] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[15] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," *arXiv preprint arXiv:1508.05326*, 2015.

# Appendix A

# Code Snippets

The code for this project can be found on the link below: `https://github.com/pankajsingh1131/research_thesis_project`