4-19-2022 11:00 AM

A Unified Representation and Deep Learning Architecture for Persuasive Essays in English

Muhammad Tawsif Sazid, *The University of Western Ontario*

Supervisor: Mercer, Robert E., *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Computer Science

Abstract

We develop a novel unified representation for the argumentation mining task facilitating the extracting from text and the labelling of the non-argumentative units and argumentation components—premises, claims, and major claims—and the argumentative relations—premise to claim or premise in a support or attack relation, and claim to major claim in a for or against relation—in an end-to-end machine learning pipeline. This tightly integrated representation combines the component and relation identification sub-problems and enables a unitary solution for detecting argumentation structures. This new representation together with a new deep learning architecture composed of a mixed embedding method, a multi-head attention layer, two biLSTM layers, and a final linear layer obtains state-of-the-art accuracy and F1 scores on the Persuasive Essays (PE) dataset. Furthermore, the augmentation of the PE corpus by including copies of major claims substituting the n-gram tokens that occur right before the major claim tokens with other major claim-introducing n-grams has aided in this increased performance on the PE dataset.

Keywords: word embeddings, argumenation mining sub-tasks, unified-representation, data augmentation, natural language processing

Summary for Lay Audience

Argumentation Mining is a research area in the field of Natural Language Processing. To detect, extract and identify argumentative structures from natural text we have to consider several sub-tasks which define the whole argumentation mining problem. Previous research works consider the argumentation detection problem as a set of many different sub-tasks. The sub-tasks are: 1) Separating non-argumentative units (sentences, words) from the argumentative units. 2) Predicting different types of argumentative components. 3) Identifying relations (support or attack) between the argumentative components. 4) Predicting the distance (textual distance measured by the number of sentences before or after) between the detected argumentative components. For example, the following sentence contains argumentative elements: "We should not get a long-haired cat (**Claim**) because cats with long hair shed all over the house (**Premise supporting** the **Claim**)." Previous research works have solved the argumentation detection task in a de-coupled way. They first detect the argumentative components, then identify stance or other relational attributes between each of the detected components. Some of the previous research works have worked with fewer sub-tasks and have obtained significant improvement regarding the detection of argumentation components. They assume that the spans of the argumentative elements (sentences, words) have been given and try to predict the correct type of argumentative components and relations from them. We have introduced a novel representation to jointly solve all four sub-tasks mentioned above. We have developed a novel neural network architecture to detect and solve all the sub-tasks related to argumentation mining. With our novel representation of the argumentation problem and the deep learning architecture, we have achieved state of the art results on the Persuasive Essays (PE) Corpus.

Acknowledgements

I would like to express my sincere gratitude and admiration to my thesis supervisor Dr. Robert E. Mercer. Dr. Robert E. Mercer's expertise was invaluable in formulating the research questions and methodology. Without his significant contributions, continuous guidance, and support this work would not have been possible.

I would like to thank the Department of Computer Science of The University of Western Ontario for providing me with the Graduate Research Scholarship.

Last but not the least, I would like to thank my parents and brother for believing in me and for providing me with continuous support and encouragement throughout my graduate study.

Contents

Abstract i

Summary for Lay Audience ii

Acknowledgements ii

List of Figures vi

List of Tables vii

1 Introduction 1

2 RelatedWork 4

2.1 Word Embeddings . . . 4

2.1.1 Word2Vec . . . 4

2.1.2 GloVe . . . 5

2.1.3 FastText Embedding . . . 5

2.1.4 Byte-pair Embedding . . . 6

2.1.5 Stacked Embedding . . . 7

2.2 Sequence Architecture . . . 7

2.2.1 LSTM Architecture . . . 7

2.3 Rise of Transformer and Multi-Head-Attention . . . 9

2.3.1 Sequence-to-Sequence Models . . . 9

2.3.2 Sequence-to-Sequence Models with Attention . . . 9

2.4 Transformer . . . 11

2.5 The Argumentation Mining Task . . . 12

3 Methodology 16

3.1 Data Set Preparation . . . 16

3.2 New Problem Formulation . . . 19

3.3 Neural Architecture and Hyper-Parameters . . . 20

3.4 Interpretation Function for the Multi-label Outputs of the Model 22
3.5 Post-Processing 22
3.6 Evaluation Metrics 24

4 Experiments and Analysis of Results **26**
4.1 Experiment with Base architecture 26
4.2 Addition of Stacked Embedding 27
4.3 Additional Linguistic Information 27
4.4 Addition of Multi-Head Attention 29
4.5 Data Augmentation Experiment on the Paragraph Version of the PE Corpus . . 33
4.5.1 Augmentation with New Paragraphs 33
4.5.2 Augmentation of New Sentence as Paragraphs 34
4.5.3 Augmented Corpus Statistics and Results Comparison 34
4.6 Error Analysis 37

5 Conclusions and Future Work **41**
5.1 Conclusions 41
5.2 Future Works 43
5.2.1 Decoupled Solution Only to Detect Distance 43
5.2.2 Introduction of Graph Neural Network to Represent Distance 43
5.2.3 New Methodology for N-grams 43
5.2.4 Future Research on Other Available Argumentation Mining Corpus . . 44
5.2.5 Contextual Embedding 44
5.2.6 Curriculum Learning 44

Bibliography **45**

A An Example of an Annotated Essay **50**

B The collected n-gram list **63**

Curriculum Vitae **67**

List of Figures

2.1 Word2Vec Model Architectures (from Mikolov et al. [24]) 5
2.2 Unfolded Version of LSTM (from Colah [27]) 8
2.3 Information bottle-neck (from Lihala [21]) 10
2.4 Context Vector at each Time-step of the Decoder (from Lihala [21]) 10
2.5 Alignment Model e_{ij} (from Lihala [21]) 11
2.6 (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel. (from Vaswani et al. [43]) 11
2.7 Transformer Full Architecture (from Vaswani et al. [43]) 12

3.1 Corpus Structure (from Eger et al. [10]) 18
3.2 Example of the New Compact Representation of the Argumentation Problem (without the 23 distance items). O: Non-Argumentative Token, B: Beginning of Argument Component, I: Continuation of Argument Component, MC: Major Claim Component, Cl: Claim Component, P: Premise Component, Sup: Support Relation Identifier, For: For Relation Identifier, At: Attack Relation Identifier, Ag: Against Relation Identifier 21
3.3 Unified-AM: Final Argumentation Model Architecture 22
3.4 Interpretation Function for Our Model Output Values 23
3.5 Post Processing Steps for Finalizing Spans 24

4.1 Comparison of Relation Classification Accuracy with Distance |d| 37

List of Tables

3.1 PE Corpus Statistics 19

4.1 Results of the Experiments on Paragraph Level with the Base Architecture and the Stacked Pre-trained Embedding 27
4.2 Experiment on Joint Architecture and Introduction of N-gram Prediction as Subtasks 29
4.3 Induced Methodology For Increasing Major Claim Prediction Accuracy 30
4.4 Experiment on Paragraph Level with the Integration of Multi-head Attention (Unified-AM) Compared with LSTM-ER [10] 31
4.5 Experiment on Essay Level with the Integration of Multi-head Attention (Unified-AM) Compared with LSTM-ER [10] 32
4.6 Precision, Recall and F1-score for the Argumentation Mining Classes for Unified-AM (Paragraph Level) 32
4.7 Precision, Recall and F1-score for the Argumentation Mining Classes for Unified-AM (Essay Level) 33
4.8 Augmented Corpus Statistics 35
4.9 Experiment on the Augmented Corpus with the Integration of Multi-head Attention (Unified-AM) 36
4.10 Token level Comparison between the Original and the Augmented Datasets . . 36
4.11 Error Analysis and Comparison Between Our Three Models (False Positives + False Negatives) on Original Paragraph Version of the Corpus 38
4.12 Comparison between Unified-AMs with other models (which do not consider subtask 1) on Original Paragraph Version of the Corpus 39
4.13 F1 scores on the BIO labeling task 39
4.14 Individual F1 scores on the BIO labeling task 40

5.1 Summary of the State-of-the-art Results Obtain by the Unified-AM Model Architecture and Comparison with the LSTM-ER Model of Eger et al. [10] 42

Chapter 1

Introduction

Argumentation is the method of constructing and handling arguments. It is an essential part of human intelligence. Humans need the ability to engage in argumentation in order to comprehend new issues, conduct scientific reasoning, and articulate, explain, and defend their viewpoints in their everyday lives. Arguments consist of claims and premises and the relationships among them. Argumentation Mining is a recent research topic in the field of Natural Language Processing. The aim of argumentation mining is to identify the arguments in a text document and the internal structure of each argument. There are four subtasks of the problem. Since we are using the Persuasive Essay (PE) dataset [42] these subtasks can be made more precise:

1. Segmenting the argument components: separate the argumentative text from the non-argumentative text;

2. Labelling each argument component: whether the argumentative text is a Major Claim, Claim, or Premise;

3. Determining which argumentation components are in a relationship: this is represented as the text distance (the number of sentences before or after) between two related components; and

4. Classifying the stance of the relations between argument components: as "support" or "attack" between premises and claims or between premises and premises; and between claims and major claims as "for" or "against".

Previous research has approached the development of a computational argumentation mining method from two distinct viewpoints. The first approach views the input as text and searches for a method to solve all four of the subtasks mentioned above. Stab and Gurevych [42] and Eger et al. [10] are noteworthy examples of this strategy. [42] provide the PE dataset

(a detailed description of this dataset can be found in Chapter 3), which we use in the development of our method. Eger et al. [10] provide (what was up until the work described in this thesis) the state-of-the-art method to which we compare our new method. Both of these works approach argumentation mining as a sequence tagging problem, that is, with a sequence as an input, for each item of that sequence, an appropriate tag needs to be found. Both works first detect entities and then predict the argument structure on top of that. Recently, Persing and Ng [35] have developed an unsupervised machine learning method that provides all but the stance information for the relations. They also use the PE dataset.

The second view of the argumentation mining problem assumes the first subtask, the segmenting of the text into argumentative and non-argumentative components, has been done, and the input to the method are the argumentative components. Peldszus [31], Peldszus and Stede [32], Stab and Gurevych [42], Niculae et al. [26], Potash et al. [37], Kuribayashi et al. [18], and Bao et al. [4] are noteworthy examples. They have produced the best results when considering only the last three subtasks of the argumentation mining problem. We compare some of our results with the last five of these works (The first two works are focused on a different corpus and we could not compare our results with those research works).

The method proposed here takes the first approach, solving all four subtasks. As there are subtasks, previous argumentation mining works have decoupled various subtasks, solved them separately, and then combined the solutions. The end-to-end learning method proposed here differentiates itself from these previous works by approaching the problem in a unified manner. Our research objectives and contributions are summarized as follows:

1. Argumentation mining is formulated as a single problem by integrating all of its subtasks: separating the non-argumentative tokens from the argumentative tokens, labelling the argument components, identifying the related components, and classifying the stance of the relation. We show that combining all the subtasks results in improved performance.

2. By constructing a dense representation of the problem we are able to achieve a better than previous performance with a model comprising two biLSTM layers, a fully connected layer, and stacked embedding for creating a better representation of paragraphs. We complete the model by including a multi-head attention layer, giving the best performance.

3. We have developed some augmentation techniques (this experiment has been done on the paragraph version of the PE corpus) based on the n-gram tokens that indicate the starting of the major claim tokens. We have identified 108 n-gram tokens of different sizes that frequently occur before the major claims of the authors in the texts. With the augmented datasets we have further improved the results on the PE corpus.

With the new formulation of the problem, our model, Unified-AM, reaches state-of-the-art argument mining performance on detecting and labelling argument components and relations for the PE corpus.

This thesis has five chapters. Chapter 2 discusses previous research work in the argumentation mining area. It also contains the literature review of the deep learning architecture components that we have used to build our model. Chapter 3 describes the methodologies that we have used to develop the novel unified representation. The detailed description of the proposed model and the PE corpus can be found in this chapter. Chapter 4 shows the state-of-the-art results that we have obtained after applying the proposed model to the task of argumentation mining of the PE corpus. Chapter 5 concludes the thesis. It summarizes the contributions and comments on the limitations of this work. In addition, it suggests a number of directions that can be investigated to advance the work provided here.

Chapter 2

RelatedWork

2.1 Word Embeddings

Word embeddings is the technique to represent words or characters in the multi-dimensional vector spaces. Word embeddings are more powerful than a simple bag of words representation of words. Unlike the sparse representation of the word token generated by the bag of words method, word-embedding methods produce dense vectors for each of the word tokens available in the vocabulary. Word embedding models have become so impressive that these word vectors can group words in such a way that if we make equations with the learned word vectors, it will give highly accurate results regarding the relationships. For example, if we represent the word king, queen, man, and woman with their corresponding word vectors K, Q, M, and W, we can have an equation where: (K - M + W) will give us the vector values which will be close to Q's vector values.

2.1.1 Word2Vec

Researchers at Google invented 2 types of model architectures (collectively known as the Word2Vec models) for the distributed representation of words [24]. They are: CBOW (Continuous Bag of Words) and the Skip-Gram model. Both models use shallow two-layer neural networks that are trained in two different ways. The CBOW model predicts the target word given the fixed size context words surrounding the target word. On the other hand, the Skip-Gram model identifies context words given the corresponding target word. Figure 2.1 shows these two types of Word2Vec models.

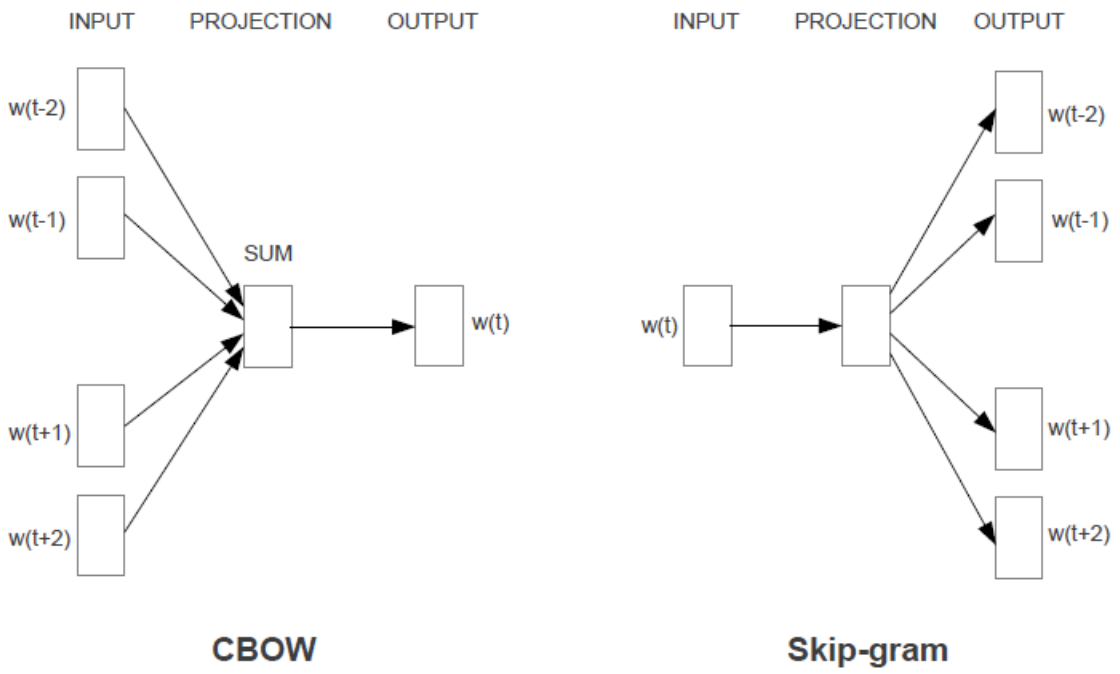

Figure 2.1: Word2Vec Model Architectures (from Mikolov et al. [24])

2.1.2 GloVe

Pennington et al. [33] developed GloVe, an unsupervised approach of learning word vectors. In the Word2Vec model, the generated embedding focusses only on the context window surrounding the target word which has the local positional and contextual information but does not have global statistical information. Contrarily, the other method of learning word vectors, known as the global matrix factorization method [8], has a limitation in the representation of the word vectors because it is more biased to the globally most frequent word tokens. This new model addresses both the global factorization and the local context window features in a combined fashion.

2.1.3 FastText Embedding

The FastText embedding [7] was developed by Facebook AI researchers. It is based on the Skip-Gram model, described previously. Predicting context words can be formulated by a set of independent binary-classification tasks. If a word is in position t, all context words (words occurring just before or after position t) are treated as positive examples for that position and the negative examples (words that are not occurring near position t) are sampled from the

vocabulary. If a word 'w' is in position t and the context word for the word is in position c, we can use the binary logistic loss for the chosen context position c. The negative log-likelihood formula is given below. In equation 2.1, $N_{t,c}$ is a set of negative examples sampled from the vocabulary.

$$\log(1 + e^{-s(w_t, w_c)}) + \sum_{n \in N_{t,c}} \log(1 + e^{s(w_t, n)}) \tag{2.1}$$

For the scoring function between the context word and a word, the dot product of their respective vectors are used. In particular, if we have one word w_t and a context word w_c the scoring function s between their corresponding vectors u_{wt}, v_{wc} can be defined as:

$$s(w_t, w_c) = u_{wt}^T v_{wc} \tag{2.2}$$

Equations 2.1 and 2.2 are formulas related to the Skip-Gram model with negative sampling [7]. However, during the generation of the word vectors for the FastText embedding, the scoring function has been modified which takes into account the internal structure of words. It does not learn vectors for words. Instead, it represents each word as an n-gram of characters. For example, take the word, 'Everything' with $n = 3$, the FastText representation of this word is "$\langle Ev, Eve, ver, ery, ryt, yth, thi, hin, ng \rangle$", where the angle brackets indicate the beginning and end of the word. This enables the embedding to learn and capture the meaning of shorter words. This n-gram technique also enables the embedding to learn suffixes and prefixes. In particular a word w can be denoted as the set of n-grams appearing in the word. Let, $G_w \subset (1, ..., G)$ be the set of n-grams representing the word. Each n-gram g is represented with a vector z_g. Then, the word w is represented by the sum of its n-gram vectors. The final scoring function is given below:

$$s(w, c) = \sum_{g \in G_w} z_g^T v_c \tag{2.3}$$

2.1.4 Byte-pair Embedding

Byte-pair embedding [12] uses the byte pair encoding (BPE) technique. It sees a series of texts as a sequence of symbols and merges the most frequent symbols iteratively. E.g., a text might consist of merging the frequent pair 't' and 'i' into 'ti'. Then 'ti' and 'm' merged into 'tim' and in the next iteration 'tim' and 'e' could be merged into 'time'. After creating symbols with the BPE algorithm, the pre-trained embedding GloVe is used for the symbols. Word level partitioning tries to solve the unknown words problem by presuming words can be reconstructed from their parts. As our corpus contains unknown words in the test set and the whole corpus heavily contains suffix and prefix dependent words, we have used both Fasttext and Byte-pair embeddings together with the help of a framework which will be described in

the next section.

2.1.5 Stacked Embedding

Alan et al. [2] introduced an NLP framework where we can mix different types of embedding and construct effective representations for a specific problem. In many tasks, a combination of different embeddings may work better than using only one embedding class. For example, Lample et al. [19] mixed classic word embeddings with character level features for their specific task and attained good results. For our task, we have used the stacked embedding class from the FLAIR framework for combining FastText and Byte-pair embedding.

2.2 Sequence Architecture

2.2.1 LSTM Architecture

A standard neural network does not have the mechanism to store previous information to predict new information from sequence data. For example, consider THE sentence: 'Australia cricket team won the 2020 world cup by defeating South Africa'. To predict the word 'defeating' from this sequential input sentence, a network should store the previous words and information about 'winning the world cup'. The feed-forward network lacks this mechanism. So for sequence data like sentences, where words and characters are interdependent and have some semantic relationships, the feed-forward network does not perform well as its weight matrix reasoning becomes ambiguous. It is unable to retain information from previous words and characters. It cannot infer semantic relationships between words from an input because it does not have a loop-back mechanism. To predict or infer accurately from sequential data, like texts, it is important to know the relationships between words in a sentence. Feed-forward networks are unable to learn these relationships from the sequential data.

As time-step and sequential data-input ideas have evolved, the recurrent neural network (RNN) comes into play. The RNN architecture has feedback loops and it has been designed to store previous information to predict future components. Though RNN promises to capture long-term dependencies theoretically, it struggles to retain long-term information which could be found in long sentences and paragraphs because of the vanishing and exploding gradient problem [6, 30]. When backpropagation algorithms perform their weight updates from the output layer towards the input layer, the gradients used to modify the weights can get very small. For this reason the weights of the initial layers remain unchanged and the gradient descent does not converge to the optimum. This issue is known as the vanishing gradient

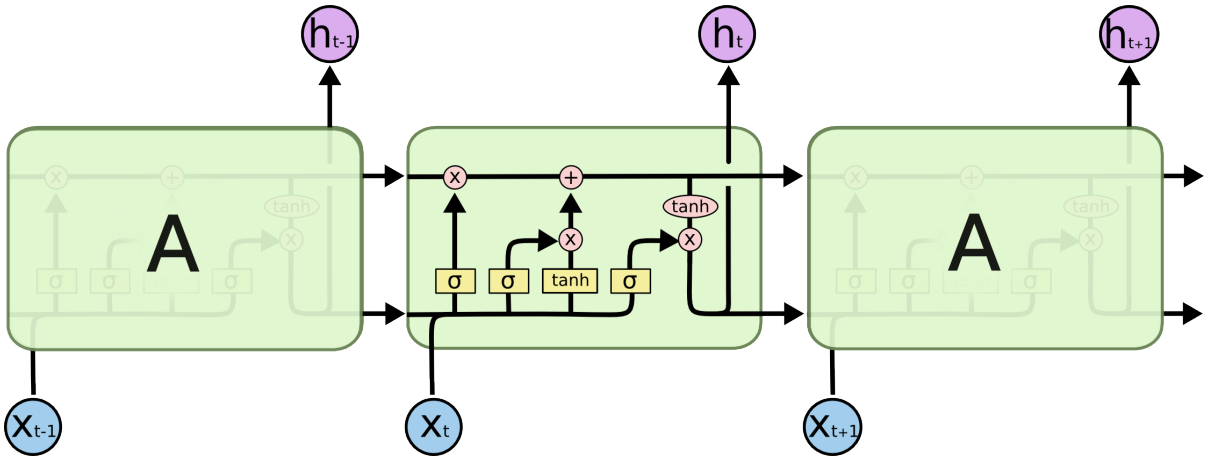

Figure 2.2: Unfolded Version of LSTM (from Colah [27])

problem. On the other hand, gradients can also become very large as the backpropagation algorithm runs. Called the exploding gradient problem, these large gradients lead to large weight updates which cause sub-optimal performance of the neural network.

LSTM [15] was created to solve the vanishing/exploding gradient problem as well as to capture long-distance dependencies in texts. In the recurrent unit, LSTM implements three types of gates allowing it to recall important details and dependencies from longer sequential data. It has a forget, input, and output gate that allows it to choose which information to carry and which to discard from memories. The vanishing and exploding gradient issues are resolved with this novel recurrent neural architecture. Figure 2.2 represents the unfolded version of the LSTM for viewing the internal structure and mechanism of a single LSTM cell:

Cell state is responsible for passing information through the LSTM chain. The gate mechanism of the LSTM enables it to pass information to the cell state and also remove previous information. Gates are composed of sigmoid neural networks. The "forget gate" takes a look in h_{t-1} and x_t to decide which unimportant information has to be removed. In equation (2.4), $[h_{t-1}, x_t]$ means a concatenation operation. Next, to add new information and update the current cell state, there are two layers involved. A sigmoid layer called the input gate layer (i_t) decides which values to be updated and the other layer is a tanh layer that constructs a vector of new $\bar{C}_t$ values which could be the potential addition to the cell state. To update the old cell state we first multiply f_t with C_{t-1} to forget unimportant information. Then we add ($i_t * \bar{C}_t$) to create the new cell state. The output gate finally outputs the results by filtering with the help of a sigmoid and tanh layer. The sigmoid layer is first applied to select the parts which will be the output. Then, the tanh layer is applied to the newly created cell state and then multiplies it with the output from the sigmoid layer. Below are the equations related to all of the gates described

above:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{2.4}$$

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{2.5}$$

$$\bar{C}_t = \tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{2.6}$$

$$C_t = f_t * C_{t-1} + i_t * \bar{C}_t \tag{2.7}$$

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \tag{2.8}$$

$$h_t = o_t * \tanh(C_t) \tag{2.9}$$

For its capability of retaining long-distance information from sequential texts, we used Sequence LSTM in both paragraph and sentence levels for the argumentation task.

2.3 Rise of Transformer and Multi-Head-Attention

2.3.1 Sequence-to-Sequence Models

A sequence to sequence model has an architecture composed of an encoder and a decoder. Both the encoder and decoder consist of recurrent neural network (LSTM) architectures. This architecture can be used in machine translation, question answering, and many other natural language problems. The encoder takes an input sentence and creates a representation of it. The final hidden state generated by the encoder is used as the input to the decoder side. Using the context information from the final state of the encoder, the decoder generates an output. The main drawback of this architecture is the information bottle-neck which has been created when producing one single vector to represent the entire context of the sentence for the decoder side. Figure 2.3 shows the issue related to the architecture.

2.3.2 Sequence-to-Sequence Models with Attention

To mitigate the bottle-neck issue of the encoder-decoder architecture Bahdanau et al. [3] and Luong et al. [23] introduced the technique named “Attention” which allows the decoder side of the architecture to attend to the hidden states generated at each time-step of the encoder. Unlike the previous methodology, this time instead of only passing the hidden state and the previous cell’s output state, a context vector is also provided to each time-step of the decoder. This context vector is the weighted sum of the encoder hidden states. Figure 2.4 shows the context vector c at each time-step. Here, c is the weighted sum of the encoder hidden states

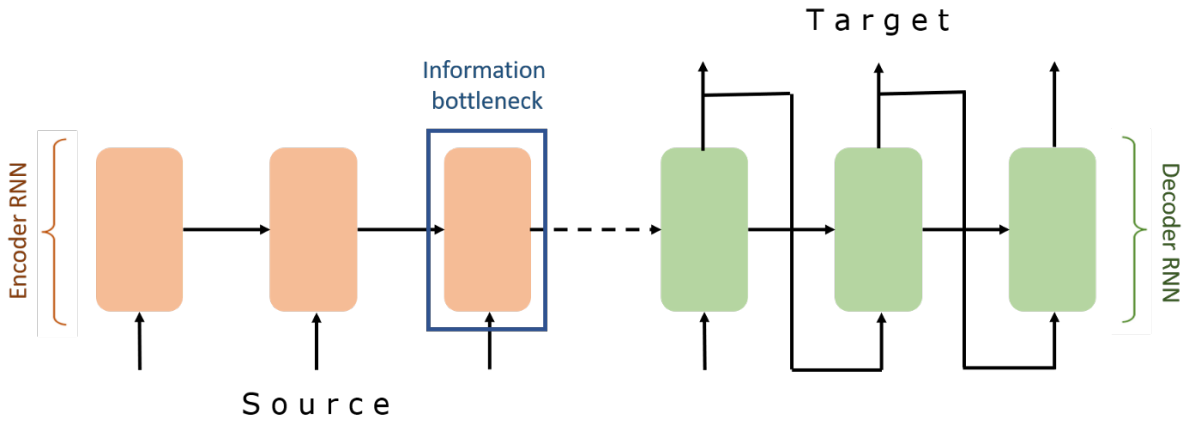

Figure 2.3: Information bottle-neck (from Lihala [21])

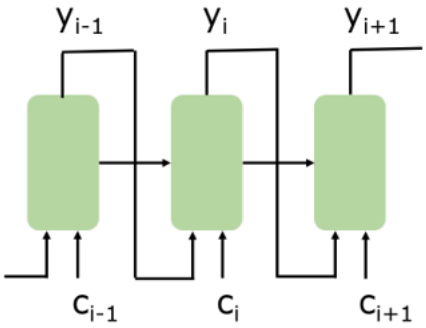

Figure 2.4: Context Vector at each Time-step of the Decoder (from Lihala [21])

and can be computed as:

$$c_i = \sum_{j=1}^{T_x} (a_{ij} h_j) \tag{2.10}$$

where a_{ij} is the attention value relating the ith output to the jth output and h_j is the encoder state for the jth input. This attention score a_{ij} is calculated as:

$$a_{ij} = \text{Softmax}(e_{ij}) = \frac{e_{ij}}{\sum_{k=1}^{T_x} \exp(e_{ik})} \tag{2.11}$$

where e_{ij} is produced by a small neural network f (see Figure 2.5) which is trained to detect how well the inputs around position j and the output at position i match where s_{i-1} represents the hidden state from the previous time-step. The formula for calculating e_{ij} is:

$$e_{ij} = f(S_{i-1}, h_j) \tag{2.12}$$

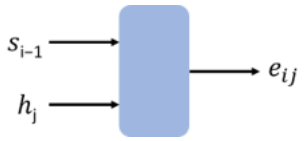

Figure 2.5: Alignment Model e_{ij} (from Lihala [21])

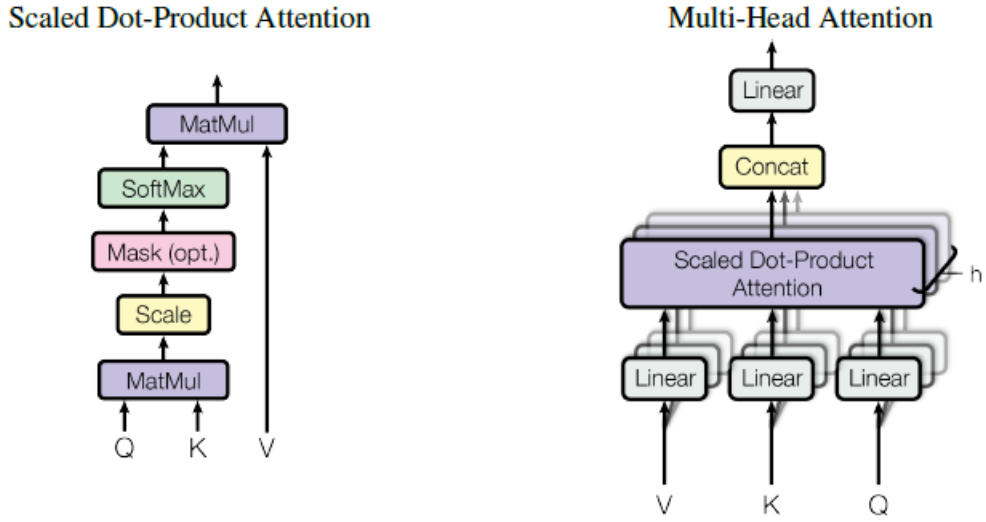

Figure 2.6: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel. (from Vaswani et al. [43])

2.4 Transformer

The invention of the transformer model [43] has been a major break-through in the field of deep-learning. It has improved the performance in many specific natural language tasks (machine translation, natural language inference, question-answering, text-generation, text-summarization, etc.) significantly. The attention function of the transformer uses query, key and value vectors for each input token. All of the query (q), key (k), and value (v) vectors corresponding to input tokens are packed together to compute them simultaneously as matrices (Q, K, V). The formula which combines the query with corresponding key and gives the weighted values is given below:

$$Attention(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.13}$$

First, the Q and K matrices are computed using the dot product. This dot product is scaled by a factor of $\frac{1}{\sqrt{d_k}}$. Then a Softmax function is applied to have weighted values V. Figure 2.6 shows the scaled dot-product attention function.

This attention function is used h times to linearly project the query, key, and values h times

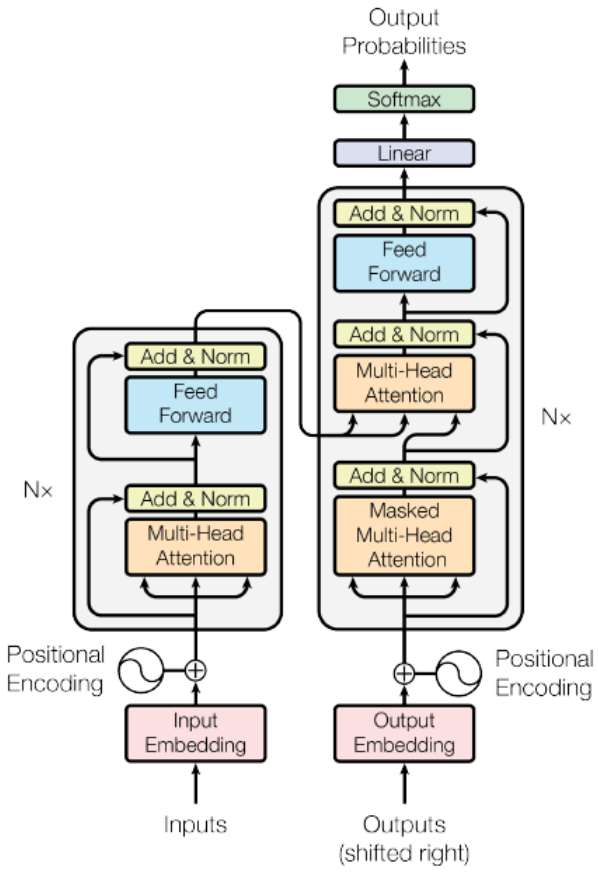

Figure 2.7: Transformer Full Architecture (from Vaswani et al. [43])

which leads to the idea of multi-head attention (see Figure 2.6). The multi-head attention block is the most important component of the transformer architecture. Besides the multi-head attention block, both the encoder and decoder side of the transformer have positional encoding, layer normalization and a final fully connected layer. Figure 2.7 shows the full architecture of a transformer model.

2.5 The Argumentation Mining Task

Argumentation mining deals with finding argumentation structures in text using computational methods. Extracting these structures requires the detection of claims and premises. In a sentence, a premise is a reason which supports or attacks the related claim. For example, if we look at a sentence from the PE corpus: “Through cooperation children can learn about interpersonal skills which are significant in the future life of all students **(claim)**. What we required from teamwork is not only how to achieve the same goal with others but more importantly how

to get along with others (**premise supporting the claim**)". Palau and Moens [28] looked at the key research questions surrounding argumentation mining, as well as the various methods that have been researched and built up to that time to overcome the challenges of argumentation mining in legal texts. Their landmark work established that argument mining would need to detect claims and premises and their relationships.

Stab and Gurevych [41, 42] provided the PE dataset, a corpus annotated with a scheme that includes claims, premises, and also attack or support relations. Stab and Gurevych [41] provided an annotation scheme that includes the annotation of claims, premises, and also attack or support relations. Inter-rater agreement of $\alpha_{\mathrm{U}} = 0.72$ for argument components and $\alpha_{\mathrm{U}} = 0.81$ for argumentative relations are the proof that this annotated corpus can be used for future research for detecting argumentation structures. The inter-rater agreement is a metric used for measuring how much the annotators have agreed while labelling a corpus. The inter-rater agreement score lies between 0 and 1. Stab and Gurevych [42] addressed the argumentation problem by training independent models for each of the subtasks and then combining them with an Integer Linear Programming Model for the end-to-end task. They identified argumentative components by sequence labeling and then argumentative relations using an integer linear programming model. This joint model architecture outperforms challenging heuristic baselines.

Eger et al. [10] got improvement by addressing the argumentation problem as a sequence tagging problem. They have the best accuracy of **61.67%** on the PE corpus by using the LSTM-ER model which has been introduced by Miwa and Bansal [25]. They did not follow the machine learning pipeline structures of previous research. They did joint learning with the help of deep learning rather than a pipeline approach of training different models in different subtasks and then combining them with integer linear programming (ILP). The LSTM-ER model identifies entities and then extracts relations between the detected entities. Their model improves over the feature-based model on end-to-end relation extraction. Miwa and Bansal [25] used a stacked architecture of Sequence and Tree LSTM. First, entities are detected with Sequence LSTM. Then this model detects relation candidates using all possible combinations of the detected entities or words where the words ended with **'L'** or **'U'** labels in the **BILOU** scheme.

Persing and Ng [34] state that a persuasive essay's argumentative structure requires discussing two issues. Identifying the elements of the essay's claim and the relationships that exist between them is a challenging task. They presented the first findings on end-to-end argument mining in student essays using a pipeline approach. This paper discusses error propagation inherent in the pipeline approach by performing joint inference using an Integer Linear Programming (ILP) framework. They introduced a new objective function that enables an ILP

solver to directly optimize F-score. They tested their combined approach with the specific objective function on a publicly available corpus of 90 writings, which yielded an 18.5 percent relative error reduction in the F-score over the pipeline scheme.

Ferrara et al. [11] introduced an unsupervised approach to detect a subtask of the argumentation mining. They examined whether topic modeling could be used to detect claims and premises in a sentence. Preliminary evaluation results are an indicator that unsupervised topic modeling could be a useful technique for detecting argumentative units. Persing and Ng [35] have also developed an unsupervised machine learning method that provides all but the stance information for the relations.

The discovery of the argument structure present in the argumentative text is one of the main goals of automatic argumentation mining. To establish this structure, one must first comprehend how the various individual parts of the overall statement are interconnected. The argument components form a hierarchy of persuasion, which manifests itself in a tree structure, according to common opinion in this area. Potash et al. [37] presented the first neural network-based approach to argumentation mining, focusing on extracting links between argument components and classifying types of argument components as a secondary goal. They suggested a modified Pointer Network structure to solve this problem. They built a joint model to increase the contribution by attempting to learn the form of argument components while also continuing to predict links between argument components. On two different evaluation datasets, the proposed model achieved state-of-the-art results. Furthermore, introducing a fully connected layer prior to the RNN unit could significantly improve performance.

A number of works have investigated approaches for subtasks 2, 3, and 4. Peldszus [31] uses a small German corpus and constructs different machine learning methods to detect argumentation structures where they obtain positive results. The experiment generates textual contents and shows that trained annotators can efficiently detect argumentation structure and invites further research for results and methods on detecting argumentation structure.

Peldszus and Stede [32] use an MST decoding algorithm and evidence graph where the edge contains the parameter or weight values for the different subtasks mentioned above. Their approach outperforms baseline and data-driven models in relation, function, and central claim identification and compares well with a complicated mstparser pipeline.

Niculae et al. [26] outperform the unstructured baseline in both web comments and argumentative essays dataset. They jointly approach unit type detections and relation predictions on their new CDCP dataset and the PE dataset. They have introduced parametrizations in SVM and RNN. Their model architecture can apply constraints and manifests dependencies between propositions and relations.

In their work, Kuribayashi et al. [18] focus on Argumentation Structure Parsing (ASP).

Their analysis of other works regarding the span representation led them to the development of a simple task-dependent addition for the ASP. A substantial amount of analysis indicates those representations achieve high performance and supply difficult types of instances for parsing.

Bao et al. [4] avoid previous inefficient enumeration operations for detecting relational attributes. For that, they introduce a transition-based methodology that follows an incremental procedure for building graphs based on argumentation. Secondly, their neural model can handle both tree and non-tree structures and is independent of structural constraints.

We investigated some neural architectures and how additional handcrafted features can help boost the accuracy on certain sequence tagging tasks such as Part of Speech (POS) tagging. Ahmed et al. [1] extracted additional features for POS tagging by using different methodology. They used character level embedding for words to capture morphological features, extract bigram (2 adjacent words) features, and also some handcrafted features (prefix, suffix, patterns) were extracted to feed the final network model. It gave, at that time, state-of-the-art performance in POS, NER, and Chunking tasks. Sense embedding has also been used besides word-level embedding. The idea of combining different learned vectors by different techniques has led to better performance.

We have developed a novel unified representation of the argumentation structures contained in the PE corpus that has been provided by Stab and Gurevych [41, 42]. We have experimented with our proposed neural model architecture together with the novel unified representation on this corpus. Our neural architecture is composed of a stacked embedding layer, two biLSTM layers, a multi-head attention layer, and a final linear layer. We have compared our results with those of Eger et al. [10] which obtained the previous state-of-the-art results on the PE corpus when considering all of the subtasks of the argumentation mining problem. The description of all of the subtasks can be found in Chapter 1. After comparing the results using the same metrics, we show that our methodology has achieved the new state-of-the-art results on the PE corpus.

Chapter 3

Methodology

In this chapter we present the method that we have developed to generate the argumentation structure for the Persuasive Essay (PE) data set. First, the data set is described. Then, we introduce the novel multi-label representation that allows us to consider argumentation mining as a single unified problem. After this, we describe our proposed model and its components and hyper-parameters that we have used to solve the argumentation task. Finally, we describe the methodology developed for interpreting multi-label outputs and a post-processing function to detect spans.

3.1 Data Set Preparation

The PE dataset was created by Stab and Gurevych [42] and used in Eger et al. [10]. The essays are written on controversial topics to let the authors make their opinions and take their stances. The corpus is tagged with the BIO scheme. Non-argumentative components are tagged as **'O'** in the corpus. Argumentative components have either a **'B' (Beginning of the component)** or an **'I' (Continuation of the component)** as a label. The dataset has 3 types of argumentative components: **MajorClaim, Claim, and Premise**. The 'Claim' component has either **'For'** or **'Against'** labels which indicate the stance type and how it is connected with the MajorClaim component. The 'Premise' component has a **'Support'** or **'Attack'** relation and it is connected to another 'Premise' or 'Claim' component with a relative distance value **(from -11 to +11)** which indicates how many sentences before or after the connected component is located in the corpus. There are essay and paragraph versions of the data set. We have worked with both versions of the corpus. Some words with the labels from the paragraph corpus are shown below. First we have the word number in a paragraph, then the word, and lastly the corresponding label of that particular word has been given. Appendix A contains one full essay as an example of the essay version of the corpus.

```
76 From O
77 this O
78 point O
79 of O
80 view O
81 , O
82 I O
83 firmly O
84 believe O
85 that O
86 we B-MajorClaim
87 should I-MajorClaim
88 attach I-MajorClaim
89 more I-MajorClaim
90 importance I-MajorClaim
91 to I-MajorClaim
92 cooperation I-MajorClaim
93 during I-MajorClaim
94 primary I-MajorClaim
95 education I-MajorClaim
96 . O

1 First O
2 of O
3 all O
4 , O
5 through B-Claim:For
6 cooperation I-Claim:For
7 , I-Claim:For
8 children I-Claim:For
9 can I-Claim:For
10 learn I-Claim:For
11 about I-Claim:For
12 interpersonal I-Claim:For
13 skills I-Claim:For
14 which I-Claim:For
15 are I-Claim:For
16 significant I-Claim:For
17 in I-Claim:For
18 the I-Claim:For
19 future I-Claim:For
20 life I-Claim:For
21 of I-Claim:For
22 all I-Claim:For
23 students I-Claim:For
24 . O
25 What B-Premise:-1:Support
26 we I-Premise:-1:Support
27 acquired I-Premise:-1:Support
28 from I-Premise:-1:Support
29 team I-Premise:-1:Support
30 work I-Premise:-1:Support
31 is I-Premise:-1:Support
32 not I-Premise:-1:Support
33 only I-Premise:-1:Support
34 how I-Premise:-1:Support
35 to I-Premise:-1:Support
36 achieve I-Premise:-1:Support
37 the I-Premise:-1:Support
38 same I-Premise:-1:Support
39 goal I-Premise:-1:Support
40 with I-Premise:-1:Support
41 others I-Premise:-1:Support
42 but I-Premise:-1:Support
43 more I-Premise:-1:Support
44 importantly I-Premise:-1:Support
45 , I-Premise:-1:Support
46 how I-Premise:-1:Support
47 to I-Premise:-1:Support
48 get I-Premise:-1:Support
49 along I-Premise:-1:Support
50 with I-Premise:-1:Support
51 others I-Premise:-1:Support
52 . O
```

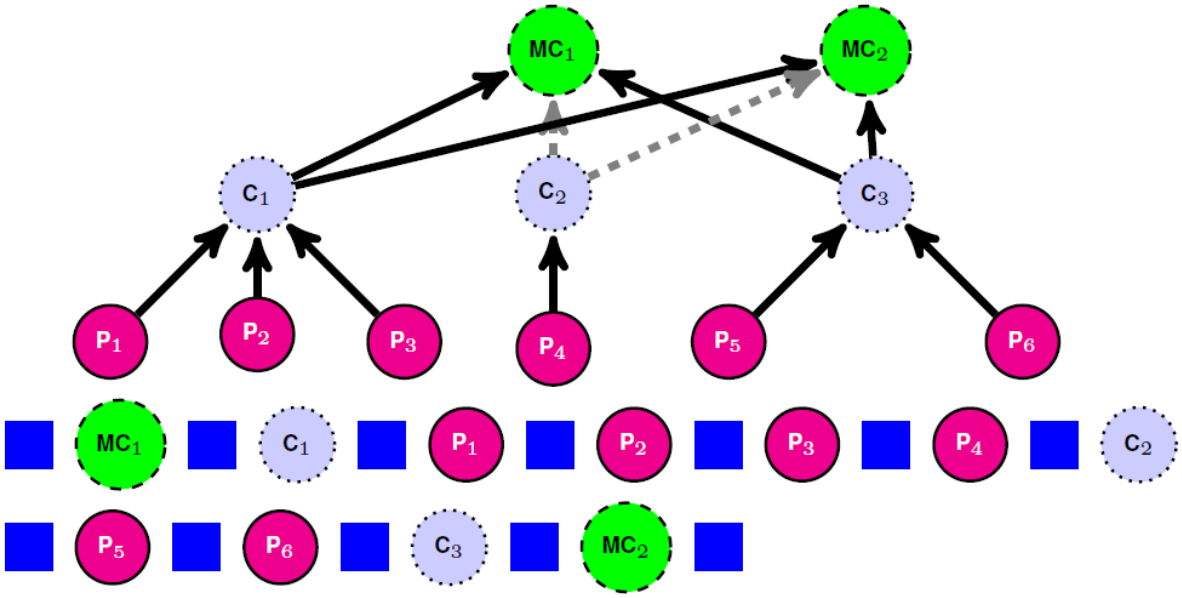

Figure 3.1: Corpus Structure (from Eger et al. [10])

Stab and Gurevych [42] has provided the dataset and the train-set, development-set, and test-set splits consisting of 1,587 paragraphs totalling 105,988 tokens in the train-set and 449 paragraphs with 29,537 tokens in the test-set[1]. The development set has 12,657 tokens available in 199 paragraphs. In the essay version of the corpus, there are 285 essays in the train-set. The development and the test set have 35 and 79 essays, respectively.

Figure 3.1, taken from Eger et al. [10], shows two representations of an essay. In the upper part of the figure, the argumentation structure can be viewed as a forest with each tree rooted by a green circle (MC_1, MC_2) representing the author's major claim. C_1, C_2, C_3 are the claims. They are connected to all of the major claims with either 'for' (solid line) or 'against' (dotted line) relations. P_1 to P_6 are all of the premises where each premise is related to exactly one claim or premise. Premises have either 'support' or 'attack' relations to the claims or other premises. The lower part of the figure represents the text that is the input to the model. All of the blue squares refer to non-argumentative components. One important piece of information is that the argumentation structure is completely contained in the paragraph except for some relations from claims to major claims which are not in the same paragraph. We have extracted the dataset at the essay, paragraph and sentence levels for our task.

We have observed that the PE corpus is imbalanced, both component and stance-wise. There are 16,332 claim, 47,683 premise, 34,124 non-argumentative tokens and only 7,849 major claim tokens in the corpus. We further look for stances and their number in the corpus.

[1]This differs slightly from what is detailed in Eger et al. [10]

Table 3.1: PE Corpus Statistics

Component	Quantity	%	Stance	Quantity	%
Major Claim	7,849	7.41			
Claim	16,332	15.41	For Against	13,536 2,796	82.88 17.12
Premise	47,683	44.99	Support Attack	45,162 2,521	94.71 5.29

BIO Label	Quantity	%
Beginning (B)	4302	4.06
Continuation (I)	67562	63.74
Non-argumentative (O)	34,124	32.20
Total	105,988	

Out of 47,683 premise tokens, 45,162 tokens have 'support' stances and only 2,521 tokens have 'attack' stances. Similarly, 13,536 claim tokens have a 'for' relation and only 2,796 tokens have an 'against' relation with major claims. A summary of these corpus statistics is provided in Table 3.1

3.2 New Problem Formulation

Sequence labeling is the task where, given a sequence of observations, we predict a class label for each observation. Some of the examples of the sequential data are textual streams, audio and video clips, time-series data, etc. As sentences, paragraphs, essays, or any other type of text can be represented and interpreted as a sequence of words, sequence tagging is very popular and common in the Natural Language Processing field. We are addressing the argumentation mining problem as a sequence labeling task, classifying each word or token as beginning argumentative / continuation argumentative / non-argumentative, premise / claim / major claim, support / attack, for / against, and the distance between the current component and the component it relates to. The maximum and minimum distances from premise to claim or premise suggested in Eger et al. [10] are +11 and -11, respectively.

For the sequential representation of the words, LSTM is a popular choice. After getting the representation from the word embedding layer, words are fed according to their position to the LSTM which processes each word sequentially (see Section 2.2). Thus, it generates a word sequence representation of a sentence, paragraph or even longer sequence such as essays by observing each previous word's sequence representation and generating a representation for the present word.

To integrate all of the sub-problems (argumentative and non-argumentative unit classifi-

cation; major claim, claim, and premise component classification; relation identification, and distance between 2 entities) into a single task, we construct a binary vector of size 33 for our target labels. Figure 3.2 is an example of this new problem representation where we combine component and relational units. Thus, we have constructed a dense unified representation of the argumentation mining problem. By formulating the argumentation mining task as a multi-label problem, we do not have to think of separated and decoupled solutions for each of the subtasks. Here the columns represent different components related to argumentative units. We took some sample tokens from one of the paragraphs in the PE dataset where the author is concluding the section with his/her major claims. The value **'1'** represents that the token belongs to that particular (non-)argumentative unit or in the case of argumentative components, the token is the continuation or beginning of that component; **'0'** indicates otherwise. Figure 3.2 only represents the first 10 columns. Some of the preliminary experiments where we have tested our unified representation for only the components and stance classifications (while not considering the distance) use this 10 labels (see Tables 4.1, 4.2, 4.3). However, all the final experiments have the distance metric and the binary vector has the size of 33 dimensions instead of 10. If the vector is representing a premise then the corresponding appropriate distance position will also contain '1' (between -11 to +11).

3.3 Neural Architecture and Hyper-Parameters

Figure 3.3 represents our final argumentation model architecture (Unified-AM) which we have created for detecting argumentation structures. The figure includes mixed embedding but in the experimental analysis we have also experimented without the pre-trained mixed embedding and trained a plain embedding layer instead. Here, we describe the final model. Our final model architecture includes: stacked embedding, axial positional embedding, multi-head attention layer, a 2-layered biLSTM and the final linear layer. The output of the model is optimized with BCEWithLogitsLoss. We have implemented an interpretation function to represent our multi-label outputs from the Unified-AM model. Section 3.4 describes this function. For its capability of retaining long-distance information from sequential texts, we use biLSTM in sentence, paragraph and essay levels for the argumentation mining task. We have determined the number of layers by using a trial and error methodology, i.e., we have tried two layers of biLSTM with one linear layer, one biLSTM layer with one linear layer, and so on. We have found one linear layer and two biLSTM layers achieve the best accuracy. Throughout all of our experiments (from base architecture to the final architecture) we use a consistent set of hyperparameter values for different parts of the architecture. After trying several hyperparameter values for each of the different components we have chosen the final values for maintaining

Token	O	B	I	MC	Cl	P	Sup	For	At	Ag
we	0	0	1	1	0	0	0	0	0	0
should	0	0	1	1	0	0	0	0	0	0
attach	0	0	1	1	0	0	0	0	0	0
more	0	0	1	1	0	0	0	0	0	0
importance	0	0	1	1	0	0	0	0	0	0
to	0	0	1	1	0	0	0	0	0	0
cooperation	0	0	1	1	0	0	0	0	0	0
during	0	0	1	1	0	0	0	0	0	0
primary	0	0	1	1	0	0	0	0	0	0
education	0	0	1	1	0	0	0	0	0	0
.	1	0	0	0	0	0	0	0	0	0

Figure 3.2: Example of the New Compact Representation of the Argumentation Problem (without the 23 distance items). O: Non-Argumentative Token, B: Beginning of Argument Component, I: Continuation of Argument Component, MC: Major Claim Component, Cl: Claim Component, P: Premise Component, Sup: Support Relation Identifier, For: For Relation Identifier, At: Attack Relation Identifier, Ag: Against Relation Identifier

consistency among all the experiments. We use dropout values of 0.5 for the linear layer, and 0.65 for the biLSTM layer of our architecture. We use the default dropout value (0.0) for the multi-head attention layer. We do not use any type of activation function in-between the layers. A learning rate of 0.001 has been used in all of the experimental design stages. The Adam optimizer is used throughout. During training, we have used random shuffling for all of the final experiments. We have trained our model around 1000-1100 epochs for all of the experiments except the data augmentation experiment (see Tables 4.9, 4.10). For determining the default training epochs (1000-1100) we have closely observed the development set accuracy value after every 5 epochs. If after some epochs the development set accuracy stops increasing or starts fluctuating somewhat between a small range of accuracy values, we have stopped the training procedure. We also observe the training loss and find that when it reaches around 0.0005 loss value, the model has the highest development set accuracy. If we further train and decrease the loss value, it does not help to improve the accuracy value of the development set. As we have also increased the original PE corpus by augmenting the data in our augmentation experiments (see Section 4.5), we also increase the training epochs to reach around the 0.0005 training loss which has given us improvements regarding the C-F1, R-F1 and F1 scores.

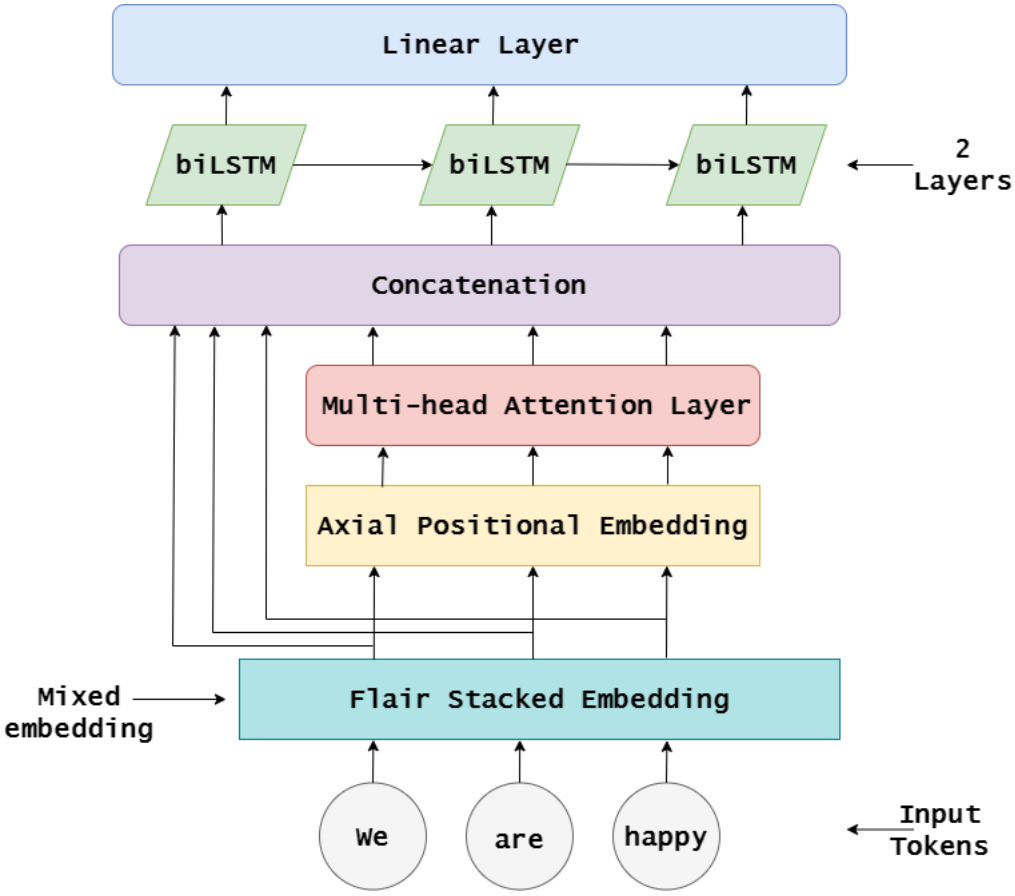

Figure 3.3: Unified-AM: Final Argumentation Model Architecture

3.4 Interpretation Function for the Multi-label Outputs of the Model

We have formulated the argumentation problem in a unified way. As a result, it has become a multi-class, multi-label problem. As it becomes a multi-label problem when we create a unified representation, we just want to choose the index for each of the categories that has the highest logit value in that specific category (components, stances, and distance). For this, we have created an interpretation function. Figure 3.4 represents our interpretation function steps applied to our model outputs.

3.5 Post-Processing

To calculate the argumentative spans located in the corpus we have done some post processing steps after applying the interpretation function on our model logits. Whenever the model has predicted one argumentative span, we have checked the entire span (each token or word

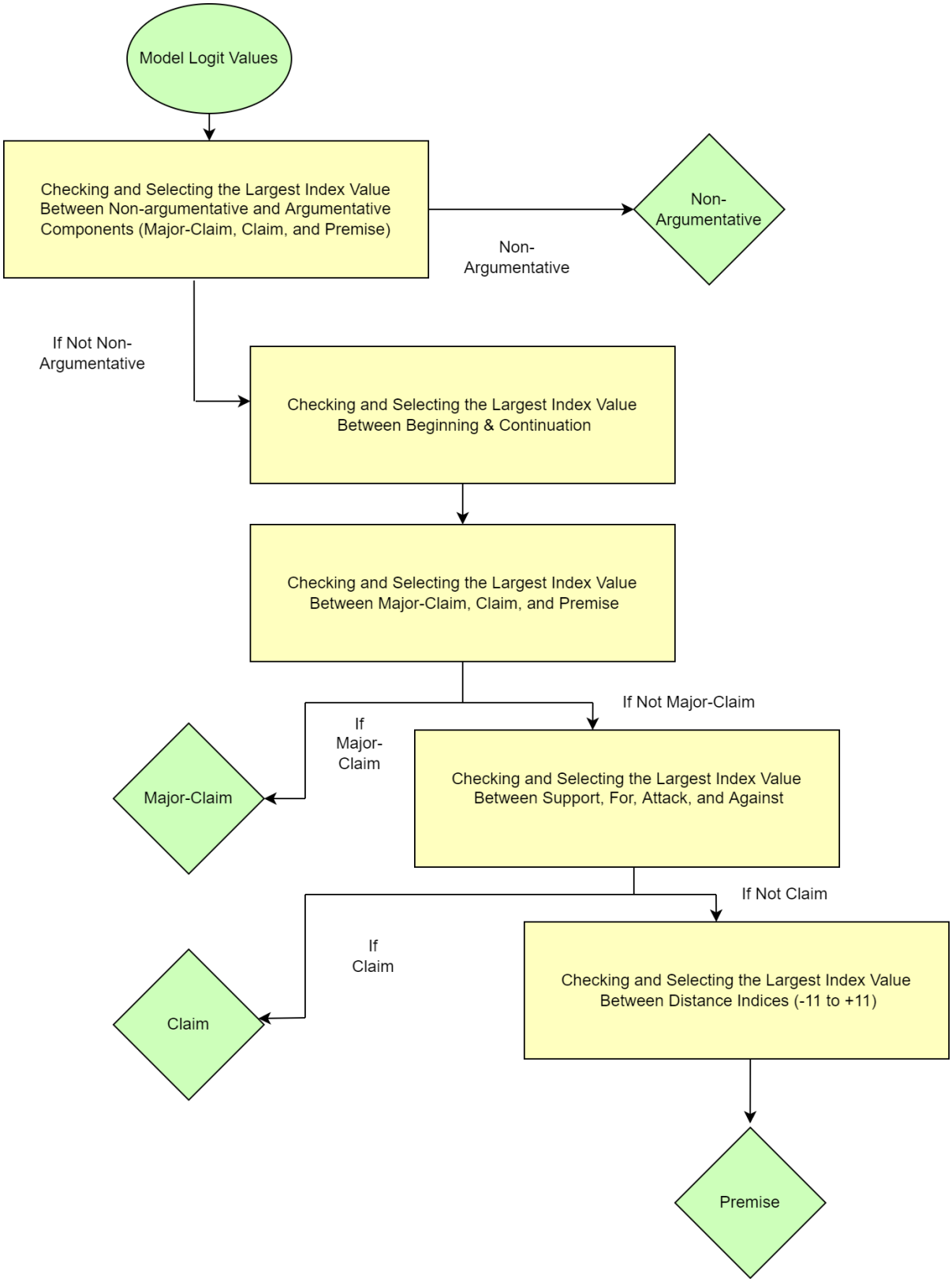

Figure 3.4: Interpretation Function for Our Model Output Values

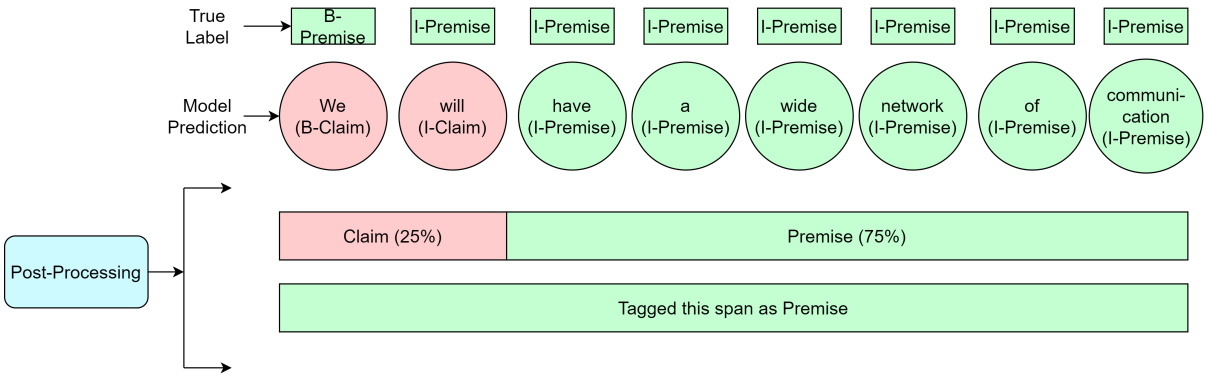

Figure 3.5: Post Processing Steps for Finalizing Spans

available in that particular span) and tagged that span as an argumentative component which has the highest number of predictions. Figure 3.5 illustrates the post processing steps for the argumentation detection task.

3.6 Evaluation Metrics

We have used the same evaluation metrics that have been used by Eger et al. [10] to compare our model's performance with their models. First, we have described the true positive (*TP*), false positive (*FP*), false negative (*FN*), and the true negative (*TN*) components.

True positive (*TP*): A true positive is the model's prediction where it correctly predicts the positive class.

True negative (*TN*): A true negative is the model's prediction where it correctly predicts the negative class.

False positive (*TN*): A false positive is the model's prediction where it incorrectly predicts the positive class.

False negative (*TN*): A false negative is the model's prediction where it incorrectly predicts the negative class.

Based on these components (*TP*, *FP*, *FN*), the evaluation metrics (accuracy, precision, recall, and *F1* score) that we have used to measure the performance of our model are given below.

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

$$F1 = \frac{2 * Precision \times Recall}{Precision + Recall} \tag{3.3}$$

Accuracy: We have also measured accuracy. Accuracy is defined as the number of correct predictions divided by the total number of predictions. We have calculated the accuracy score at the token or word level.

Chapter 4

Experiments and Analysis of Results

In this chapter, we present our results and experiments in a bottom-up style, starting with a base architecture to which we make a series of additions, since we want to discuss the motivation for these additions. We provide at each stage the performance increase given by that addition. We compare the final model's performance with that achieved by Eger et al. [10], the state of the art prior to this thesis.

4.1 Experiment with Base architecture

In the first experiment, we do not consider using any pre-trained word embedding and have trained an embedding layer initialized with random values with the newly created compact representation of the argumentation task. First, a few tables (Tables 4.1, 4.2, and 4.3) contain results where we do not consider the distance prediction problem and we have represented the argumentation problem with only the first 10 labels (see Figure 3.2). The base architecture is composed of the embedding layer, two layers of biLSTM, and one linear layer. We have measured accuracy at the token level. Our simple model architecture has achieved an accuracy of **72.58%** where we did not consider the distance aspect of the problem (see Figure 3.2).

The results related to the experiment indicates that when we use the unified representation of the problem where we have represented both components and relations, it does not hurt the token level accuracy. Rather, the novel unified representation motivates us to carry on further experiments that we have described in the next sections.

To further support our claim that the new problem representation is an important reason to solve the entity and relation classification subtasks in a unified manner, we were motivated to investigate the use of the new representation at the sentence level since this decouples the interaction that happens when solving at the paragraph level. The accuracy declines to 66.01%. The primary reason for the decline is argumentation structures are contained at the single para-

Table 4.1: Results of the Experiments on Paragraph Level with the Base Architecture and the Stacked Pre-trained Embedding

Paragraph Level		
Embedding	**Model**	**Token Accuracy**
Embedding Layer (No Pre-trained Embedding Used)	biLSTM (2 Layers) Linear (1 Layer)	**72.58%**
Flair Stacked Embedding (Standard FastText Embedding and Byte-Pair Embedding)	biLSTM (2 Layers) Linear (1 Layer)	**79.17%**

graph level (with the exception of some claim to major claim relations) and relations are only sometimes contained in a single sentence.

4.2 Addition of Stacked Embedding

In the second experiment, the embedding layer in the base architecture is replaced with a stacked embedding. Akbik et al. [2] has developed an NLP framework where we can combine different types of embeddings. For our task, we use the memory-efficient stacked embedding class from the Flair framework for combining FastText and Byte-pair embedding [2]. As our corpus contains unknown words in the test set and the whole corpus contains many suffix and prefix dependent words, we have used these two types of embedding together.

We obtain an accuracy of **79.17%**. The only difference here is the addition of the stacked embedding class and the mixing of two pre-trained embeddings. FastText and Byte-pair embeddings jointly handled the out of vocabulary words in our test set. Also, it has produced a richer representation for the tokens available in the PE dataset. We do not fine-tune these two pre-trained embedding classes for our argumentation task. The results of the first two experiments are summarized in Table 4.1.

4.3 Additional Linguistic Information

Having had the successes described above, we looked for other aspects of the problem that could be improved upon. Noting that the major claims are the roots of the argumentation tree structure and because the first experiment pointed to how important is the interaction of

learning the different aspects of the problem, it was thought that improving on the recognition of major claims would help to improve the recognition of claims because they are related. This would then further improve the recognition of premises. Taking inspiration from the addition of linguistic information done by Ahmed et al. [1] in their work, we have considered adding this type of knowledge. We have observed (as did Kuribayashi et al. [18], and Persing and Ng [35]) that many major claims, claims, and premises are prefaced by a reasonably small set of n-grams. An n-gram is a continuous sequence of *n* words. Some examples of the n-grams that are found in the PE corpus are: 'I firmly believe that', 'In conclusion ,', 'Hence ,', and 'Firstly ,'. We have presented all of the 108 n-grams that we have collected from the train-set for all the experiments related to n-grams in Appendix B.

We have experimented with a joint architecture where the first model attempts to detect only the n-gram, major claim, and non-argumentative tokens. We have hypothesized that, if we detect the major claim tokens together with the collected n-gram tokens and separate the prediction of the two other argumentative components (premise and claim) by using a second model, it would help the model distinguish between the root (major claim tokens) and the other argumentative components observed in the PE corpus. We have collected the n-gram features for major claims, premises, and claims and added the subtasks of predicting n-grams to improve accuracy on entity identification. For this experiment, the target labels for our first model become: Y = [*N-gram for major claim, Non-argumentative, Major Claim, Premise/Claim flag, N-gram for premise and claim*] (total 5 target labels). This model, a simple biLSTM, easily learns the n-grams: accuracy for major claim n-gram is 98.61%, for premise and claim (one label for a merged class) the accuracy is 99.74%. In this first model, a 'premise-claim flag' label is used to separate both premise and claim tokens from major claim tokens. We reintroduce premise and claim labels separately in a second model. Instead of labelling the n-gram tokens as non-argumentative tokens, we introduce n-gram tokens as a separate class.

The target labels for this second model are: Y = [*N-gram for major claim, Non-argumentative, Major Claim, Premise/Claim flag, N-gram for premise and claim, Claim, Premise, Support, For, Attack, Against*] (11 target labels). We create the joint model in two ways. First, we take the first model's biLSTM outputs and concatenate it with the second model's biLSTM output prior to the linear layer. The accuracy score is **73.98%**. This low accuracy may be due to information loss since concatenation doubles the size of the vector which is then projected into a lower dimensional space by the linear layer. To overcome this information loss, we replace concatenation with a summation operation in a second joint model. The accuracy is increased to **77.64%**, but is still below that obtained by our basic architecture with stacked embedding (see Table 4.1).

Although it is expected that introducing information about the n-grams for major claims

Table 4.2: Experiment on Joint Architecture and Introduction of N-gram Prediction as Subtasks

Joint Architecture on Paragraph Level		
Combination method	**Token Accuracy**	**Wrong MC Predictions**
Concatenation	73.98%	788
Summation	**77.64%**	658

would improve the major claim token prediction accuracy, it actually decreases. The best wrong major claim predictions is for the second model: 658. The basic architecture with stacked embedding (see Section 4.2) achieves better results for major claim tokens—618 wrong predictions for major claims (**1516** correct out of 2134 major claim tokens)—and overall accuracy. Table 4.2 provides a summary of this information for these two joint models.

Although the joint model idea does not result in the anticipated improvements, we are still committed to incorporate the n-gram information (next sections have addressed this problem by working on the model architecture by introducing attention module). We now move in the next sections to the last addition to the model, which gives the final architecture (see Figure 3.3), and the analysis of the performance of this final model.

4.4 Addition of Multi-Head Attention

In our next experiment we introduce a multi-head attention module (see Section 2.4). Experiments with a full Transformer layer produced poorer results. We use axial positional embedding for the positional information [13, 17]. With the multi-head attention module we have returned to using our single level architecture scheme of stacked embedding, biLSTM, and a final fully connected linear layer (see Figure 3.3) rather than using the just discussed joint model in the previous section. Attention mechanism has proven to be very effective for solving many types of natural language tasks (machine translation, natural language inference, question-answering, text-generation, etc.). We have tried experimenting with the attention module to represent specifically the n-gram features which in turn will help to increase the accuracy of detecting the major claim tokens. We have experimented to construct different representations for our word vectors to feed the query, key and value matrices of our attention module (see Section 2.4 for the description of the transformer). Our goal in this particular case is to induce the model with n-gram features specifically for better prediction on the detection of the major claim tokens. We only consider the stacked embedding representation of the word tokens which are contained in n-grams. For other word tokens we use 0 valued vectors. We feed these representations to our attention module to persuade the model to focus on the word

Table 4.3: Induced Methodology For Increasing Major Claim Prediction Accuracy

Operation difference	**Extra biLSTM used**	**Multi-head attention integration**	**Major Claim correct predictions**	**Overall token level accuracy**	**Training loss**
Summation	No	After biLSTM	1445	74.68	0.0004
Concatenation	No	After biLSTM	**1495**	77.07	0.0004
Concatenation	No	After biLSTM	1371	**77.24**	0.0002
Summation	Yes	Before biLSTM	1490	76.25	0.0003
Concatenation	Yes	Before biLSTM	1491	75.21	0.0002

tokens which represent our hand-crafted n-gram features only. Table 4.3 contains accuracy information for different operations related to the above experiment.

We speculate that the reason we do not have any significant improvements on the major claim detection tasks by this induced methodology is because major claim tokens are relatively rare in the whole corpus. About only 7.40% of the whole corpus represent the major claim tokens which make the prediction task difficult for this particular token even after injecting extra relevant n-gram information that has been considered useful. The more we train our argumentation model it directs itself toward gaining higher overall accuracy. We address this relatively low frequency of the major claim tokens later in Section 4.5. But before doing that we now discuss the experiments done on the final argumentation model architecture.

In the next experimental setup, we have used the multi-head attention module and have fed the stacked embedding representation for all tokens to the query, key and value matrices that we are using for solving the argumentation mining task instead of considering the persuasive methodology for the major claim task discussed previously (i.e., only using the stacked representation for the hand-crafted features and representing other words as 0-valued vectors). For our 400-dimension embedding class we use four heads for the multi-head attention layer for this experiment. We have called this final model architecture, Unified-AM (see Figure 3.3). Regarding this problem our target label vector becomes: Y = [*Non-Argumentative, Beginning, Continuation, Major Claim, Claim, Premise, Support, For, Attack, Against, (-11 to +11)*] (33 labels). We have used the distance values from -11 to +11 that were observed by Eger et al. [10] in the PE data set.

With this final architecture, we are in a position to begin comparing our results with those

Table 4.4: Experiment on Paragraph Level with the Integration of Multi-head Attention (Unified-AM) Compared with LSTM-ER [10]

Paragraph Level							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	**66.79%**	68.88	**78.22**	**51.14**	**56.41**	**60.00**	**67.32**
LSTM-ER	61.67%	**70.83**	77.19	45.52	50.05	55.42	60.72

given by Eger et al. [10] which until now have been state-of-the-art. Before presenting the results of this experiment, an important difference between the results presented in the remainder of the thesis and those given earlier needs to be pointed out. Previously, "token level accuracy" meant the correct prediction of the 10 component labels only, i.e., were the words in the text given the correct major claim, claim, or premise label, and the correct stance labels for claim ('for' and 'against') and premise ('support' and 'attack'). This labelling was considered sufficient to make some decisions in the early architecture design development. Henceforth, "token level accuracy" will mean the correct prediction of the complete labels as annotated in the corpus. This now includes the distance metric. In addition, new scores are also computed: C-F1 100% and R-F1 100%. These scores represent the F1-scores of predicted spans of text that perfectly match the test-set spans for component labels (Beginning, Continuation, Major Claim, Claim, and Premise) and relation labels (stance and distance), respectively, where true positives are defined as the set of predicted spans which perfectly match the true spans. C-F1 50% and R-F1 50% are the respective scores for predicted spans that have at least a 50% overlap with the test-set spans, that is, true positives are defined as the set of predicted spans which overlap the true spans at least 50%. The meaning of these scores correspond to that given by Eger et al. [10].

This final architecture, Unified-AM, achieves token level accuracy of **66.79%** for the argumentation mining task on the paragraph-level. Table 4.4 summarizes the result for this experiment, including the F1 score for the component and relation tasks, and a global F1 score. The results from Eger et al. [10] have been included for comparison. Now, compared to the Eger et al. decoupled method for computing the relation identification [10], the unified representation of the problem in Unified-AM couples this task with the component identification task and leads to the better performance. Table 4.4 shows the results.

We have also experimented on the essay-level of the argumentation corpus and our Unified-AM model has achieved the highest token level accuracy, C-F1, and R-F1 scores. The experiments on the essay version of the corpus show the robustness of the unified representation of all the subtasks with our model. When we experiment on the essay version of the corpus, our scores and results have not decreased like the LSTM-ER model and its decoupled solution.

Table 4.5: Experiment on Essay Level with the Integration of Multi-head Attention (Unified-AM) Compared with LSTM-ER [10]

Essay Level							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	**62.88%**	**67.78**	**76.20**	**48.24**	**52.49**	**58.01**	**64.35**
LSTM-ER	54.17%	66.21	73.02	29.56	32.72	40.87	45.19

Table 4.6: Precision, Recall and F1-score for the Argumentation Mining Classes for Unified-AM (Paragraph Level)

Class	**Precision**	**Recall**	**F1 Score**	**Token Percentage (Approx)**
Non-Argumentative	88.38	88.27	88.33	32.20
Major Claim	73.87	74.18	74.02	7.41
Claim	65.37	58.05	61.48	15.41
Premise	88.01	90.87	89.42	44.99
Support	86.79	89.69	88.22	42.61
For	60.96	57.05	58.94	12.77
Attack	32.52	26.77	29.37	2.38
Against	60.81	29.97	40.15	2.64

Table 4.5 summarizes the results for this experiment.

In Table 4.6, we present individual precision, recall and F1 scores for the unified representation of the components and relations that are available in the PE corpus. We observe low precision and recall scores for the claim tokens even though the class is not the least frequent one in the PE corpus. A similar phenomenon was noted by the creators of the corpus: the lowest agreement score among the human annotators for the argument component types (Major Claim, Claim, Premise) is also for Claim [42]. We have observed the lowest recall value for the 'attack' component. This component is only about 2% of the whole training dataset. During the training procedure, the model has found very few example sentences where this component occurs and as a result the false negative rate is high for the 'attack' component. Similarly Table 4.7 shows individual precision, recall and F1 score for the essay version of the corpus.

We now turn to the final argumentation model performance improvement. Recalling that the previous modifications to the model architecture to induce detection of the major claim tokens did not work, we now turn to augmenting the corpus to increase the frequency of this component type.

Table 4.7: Precision, Recall and F1-score for the Argumentation Mining Classes for Unified-AM (Essay Level)

Class	Precision	Recall	F1 Score	Token Percentage (Approx)
Non-Argumentative	89.26	91.27	90.25	32.20
Major Claim	70.34	72.05	71.19	7.41
Claim	56.11	49.34	52.51	15.41
Premise	87.18	88.32	87.74	44.99
Support	85.09	88.35	86.69	42.61
For	56.41	50.76	53.43	12.77
Attack	27.08	7.10	11.25	2.38
Against	21.01	10.23	13.76	2.64

4.5 Data Augmentation Experiment on the Paragraph Version of the PE Corpus

In this experimental setup, we have augmented the PE dataset (which consists of paragraphs) in two ways. Below, we describe the two different augmentation techniques that we have used to augment the PE corpus. We also compare the performance of Unified-AM on both of the augmented and original corpora.

4.5.1 Augmentation with New Paragraphs

We have augmented the paragraph-level corpus with new paragraphs. These new paragraphs are copies of those paragraphs that contain one of the 108 n-gram tokens that occur immediately before the major claim tokens (see Appendix B) but have had the n-gram randomly swapped with a same size n-gram token. This augmentation increases the number of major claim tokens in the whole corpus but with different introductory n-grams. We have hypothesized that if we increase the root element, i.e., the major claim components of the corpus, by swapping frequently occurring n-gram tokens that appear immediately before the component, it would help the model to accurately detect this type of component and differentiate between the three types of components that are available in the PE corpus. We have shown below an example of the original paragraph and the augmented paragraph after applying the described augmentation method:

Original Paragraph: "It is always said that competition can effectively promote the development of economy . In order to survive in the competition , companies continue to improve their products and service , and as a result , the whole society prospers . However , when we discuss the issue of competition or cooperation , what we are concerned about is not the whole

society , but the development of an individual's whole life . I firmly believe that we should attach more importance to cooperation during primary education."

Augmented Paragraph: "It is always said that competition can effectively promote the development of economy . In order to survive in the competition , companies continue to improve their products and service , and as a result , the whole society prospers . However , when we discuss the issue of competition or cooperation , what we are concerned about is not the whole society , but the development of an individual's whole life . I truly believe that we should attach more importance to cooperation during primary education."

Description of the Augmentation Process: In this particular example we have substituted the n-gram "I firmly believe that" with an equal size randomly chosen n-gram "I truly believe that" from our collected n-gram list. The words following in that particular sentence are major claim tokens. Here, the n-grams consist of 4 words.

4.5.2 Augmentation of New Sentence as Paragraphs

In this method, we only take the particular sentence which contains the n-gram tokens as well as the major claim tokens to create a new sentence which will be treated as a separate paragraph during the training phase. One example is shown below:

Original Paragraph: "It is always said that competition can effectively promote the development of economy . In order to survive in the competition , companies continue to improve their products and service , and as a result , the whole society prospers . However , when we discuss the issue of competition or cooperation , what we are concerned about is not the whole society , but the development of an individual's whole life . I firmly believe that we should attach more importance to cooperation during primary education."

Augmented Sentence as Paragraph: "I truly believe that we should attach more importance to cooperation during primary education."

Description of the Augmentation Process: Here, we have swapped the n-gram "I firmly believe that" with the randomly chosen "I truly believe that" and created a paragraph of only one sentence that contains major claim tokens after the n-gram. Both n-grams have 4 words.

4.5.3 Augmented Corpus Statistics and Results Comparison

Overall corpus statistics after augmentation are shown in Table 4.8. After creating the augmented corpus, we have trained our Unified-AM model (see Figure 3.3) on both of the augmented corpora. We have achieved highest token level accuracy on the paragraph-level argumentation corpus. Previously, without augmentation, we have achieved 66.79% token level accuracy on the PE dataset (see Table 4.4) and after applying the augmentation methodology

Table 4.8: Augmented Corpus Statistics

Original Corpus (Paragraph level)				
Total Major Claim Tokens	Total Claim Tokens	Total Premise Tokens	Total Non-Argumentative Tokens	Total New Paragraphs
7849	16332	47683	34124	-
Augmented Corpus (Addition of New Paragraphs)				
Total Major Claim Tokens	Total Claim Tokens	Total Premise Tokens	Total Non-Argumentative Tokens	Total New Paragraphs
11794	18480	48168	42862	265
Augmented Corpus (Addition of New Sentence as Paragraphs)				
Total Major Claim Tokens	Total Claim Tokens	Total Premise Tokens	Total Non-Argumentative Tokens	Total New Paragraphs
11650	16788	47683	35635	265

we have achieved the highest token level accuracy of **68.02%**. Also, all other performance measures have been improved significantly. We further worked on the paragraph-level augmentation and trained it more (1500-1600 epochs). Table 4.9 shows the results related to the augmented datasets. If we compare Unified-AM's performance between the augmented corpus and the original corpus (see Table 4.9), the model has much higher token level accuracy, C-F1, R-F1, and F1 scores when we apply augmentation techniques on the training corpus. We have reached the highest component C-F1(100%) score of **71.35%** where Eger et al. [10] has obtained 70.83%.

We present in Table 4.10, the token level improvements and compare them with the original PE corpus results. In the test set, we have in total 2,134 major claim tokens, 4,238 claim tokens, 13,728 premise tokens, and 9,437 non-argumentative tokens. Our goal is to increase the major claim tokens which can be considered as the root of the argumentation structure. The results provided in Table 4.10 show the overall token level improvements that we get compared to the original paragraph version of the PE corpus. These results indicate that both augmentation techniques have significantly improved the previous predictions regarding the major claim, claim, and premise tokens.

Table 4.9: Experiment on the Augmented Corpus with the Integration of Multi-head Attention (Unified-AM)

Original Paragraph Corpus							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	66.79%	68.88	78.22	51.14	56.41	60.00	67.32
Augmented Corpus (Addition of New Paragraphs)							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	67.02%	71.03	79.82	52.50	58.25	61.77	69.04
Augmented Corpus (Addition of New Paragraphs) Training Epochs: 1500-1600							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	**68.03%**	**71.35**	80.21	**54.27**	**59.46**	**62.81**	**69.83**
Augmented Corpus (Addition of New Sentence as Paragraphs)							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	66.85%	69.74	**80.72**	52.26	58.92	61.00	69.82

Table 4.10: Token level Comparison between the Original and the Augmented Datasets

Original Corpus (Paragraph level)			
Correct Major Claim Tokens	Correct Claim Tokens (with Stance: For, Against)	Correct Premise Tokens (with Stance: Support, Attack and Distance -11 to + 11)	Correct Non-Argumentative Tokens
1542	2057	7329	8217
Augmented Corpus (Addition of New Paragraphs)			
Correct Major Claim Tokens	Correct Claim Tokens (with Stance: For, Against)	Correct Premise Tokens (with Stance: Support, Attack and Distance -11 to + 11)	Correct Non-Argumentative Tokens
1633	2287	7612	**8266**
Augmented Corpus (Addition of New Paragraphs) Training Epochs: 1500-1600			
Correct Major Claim Tokens	Correct Claim Tokens (with Stance: For, Against)	Correct Premise Tokens (with Stance: Support, Attack and Distance -11 to + 11)	Correct Non-Argumentative Tokens
1597	**2344**	**7956**	8196
Augmented Corpus (Addition of New Sentence as Paragraphs)			
Correct Majo Claim Tokens	Correct Claim Tokens (with Stance: For, Against)	Correct Premise Tokens (with Stance: Support, Attack and Distance -11 to + 11)	Correct Non-Argumentative Tokens
1567	2120	7275	8153

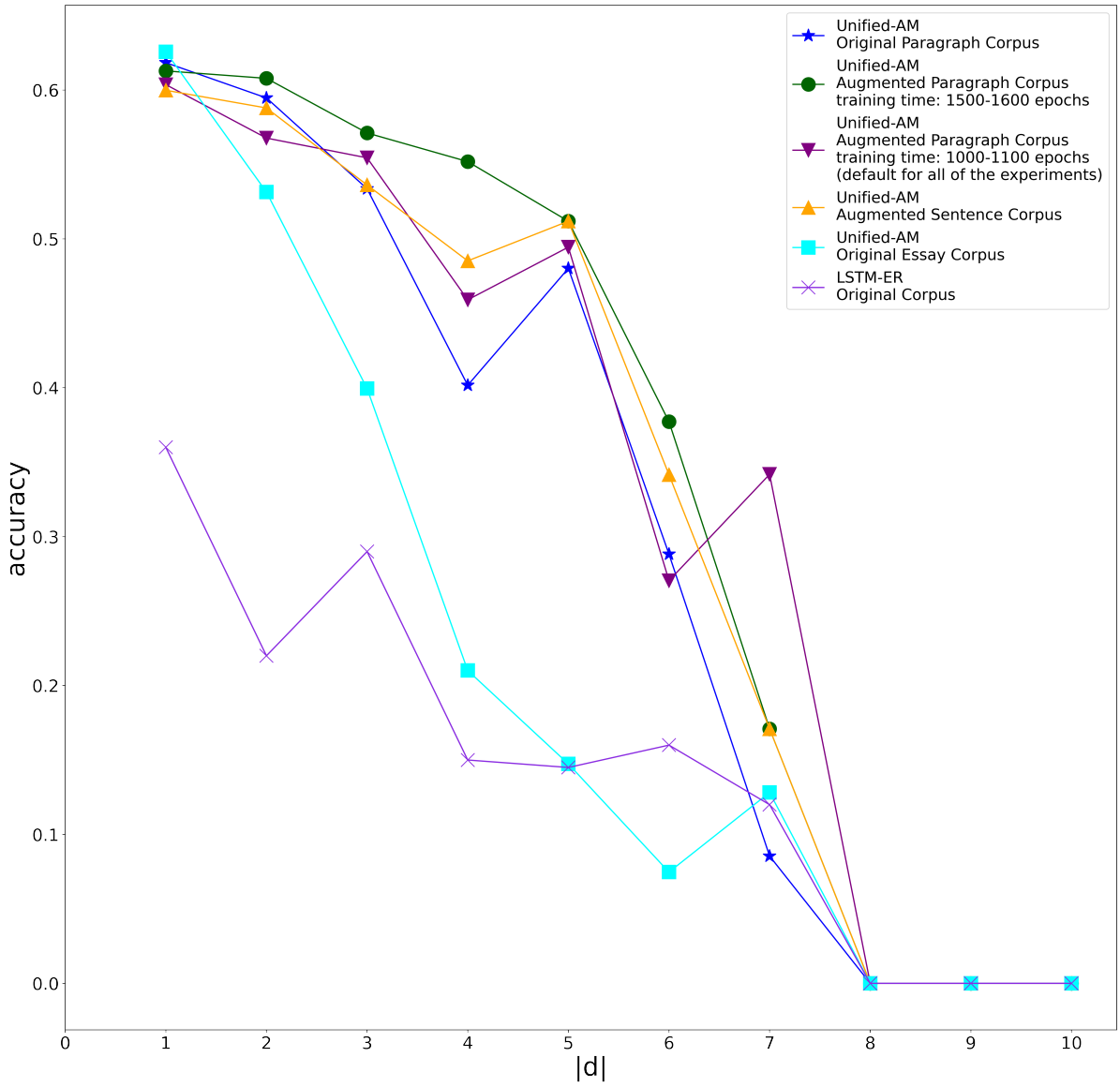

Figure 4.1: Comparison of Relation Classification Accuracy with Distance |d|

4.6 Error Analysis

We have done some error analysis and comparison among our neural architectures. The results obtained show how all the models perform on the argumentation task. The argumentation model which consider subtasks of distance learning (total 33 labels), we observe a higher accuracy of predicting longer distance in the paragraphs compared to the essays. Figure 4.1 shows the distance accuracy information. One of the key strategies that we have followed for all of these experimental setups: We ensure the models share all of their learned parameters while solving any particular subtask (component detection and labelling, relation classification, or accurate distance prediction) of the main Argumentation Mining problem. This denser

Table 4.11: Error Analysis and Comparison Between Our Three Models (False Positives + False Negatives) on Original Paragraph Version of the Corpus

	Number of Wrong Predictions				
	Major Claim	**Claim**	**Premise**	**Relations**	**Non Argumentative**
Trained Embedding + biLSTM + LINEAR	1306	4011	4787	10004	3255
Pretrained Stacked Embedding (FastText + Byte-pair) + biLSTM + LINEAR	1176	3215	3122	7653	2120
Unified-AM	1111	3082	2953	7301	2202

representation of the whole argumentation task enables our neural models to share all of the parameters while making predictions for each of the subtasks which has led to a high performance. According to Eger et al. [10], around two-thirds of the relation distances have values of -2, -1, and 1. Figure 4.1 compares the accuracy obtained by Eger et al. [10] with ours. We observe that the LSTM-ER model's probability of correctness given true distance is below 40% (no details regarding which version of the corpus they have used to compare relation classification accuracy with distance [10]) and it is below 20% when we observe distances which are larger than 3. But in our case, we see above 50% accuracy for distances 1, 2, and 3 (on the original paragraph corpus), and distances 1 and 2 (on the original essay corpus). Our final model (see Figure 3.3) has higher accuracy regarding smaller distances but its prediction accuracy declines as we observe larger distance values in the PE corpus. For our augmented data experiment, we observe the highest accuracy on the distance prediction task over a longer range. Our Unified-AM model maintains prediction accuracy of above **50%** for distances 1, 2, 3, 4, and 5. The performance drops when we consider predicting distance values in the essay version of the corpus. The reason is essays are much longer text sequences and that could affect the learning of our Unified-AM model which is based on biLSTM. Longer sequence lengths can cause the vanishing gradient problem [14] which makes it difficult for the model to learn the specific subtask.

We compare the number of wrong-predictions between our pre-trained stacked embedding model without multi-head attention (see Section 4.2), non-pre-trained embedding model (see Section 4.1), and the final Unified-AM model (seeFigure 3.3). Table 4.11 represents error analysis results for the non-argumentative units, argumentative components, and relations. For

Table 4.12: Comparison between Unified-AMs with other models (which do not consider subtask 1) on Original Paragraph Version of the Corpus

Method	**ACTC**			
	Macro	MC	Claim	Premise
Joint-ILP	82.6	89.1	68.2	90.3
St-SVM-full	77.6	78.2	64.5	90.2
Joint-PN	84.9	89.4	73.2	92.1
Span-LSTM	87.3	-	-	-
Span-LSTM-Trans	87.5	**93.8**	76.4	92.2
BERT-Trans	88.4	93.2	**78.8**	93.1
Unified-AM	**89.18**	92.30	78.25	**96.98**

Table 4.13: F1 scores on the BIO labeling task

	STag_BLCC	**LSTM-ER**	**ILP**	**HUB**	**Unified AM Original Corpus**	**Unified AM Augmented Paragraph Training Time (1500-1600 epochs)**	**Unified AM Augmented Paragraph**	**Unified AM Augmented Sentence**
Essay	90.04	**90.57**	-	-	90.52	-	-	-
Paragraph	88.32	**90.84**	86.67	88.60	89.69	89.88	90.40	89.46

each of the mentioned argumentative units we present the total number of errors (false negatives + false positives). For relations (support, attack, for, and against), we have combined the errors from each class and report this combined value. There are somewhat fewer wrong predictions when the stacked embedding is incorporated into the model. Without stacked embedding, the total number of wrong predictions for all of the classes on the paragraph level is **23,363**. With the addition of stacked embedding the total number of wrong predictions becomes **17,286**. After using this pre-trained embedding, the error rate is reduced by **26.01%**. The total number of errors for the Unified-AM model is **16,649**. The Unified-AM model further reduces the error rate by **3.69%**.

Recalling that subtask 1 of the argumentation mining problem is the separation of the argumentative text from the non-argumentative text (see Chapter 1), we compare Unified-AM's performance with some of the recent works where the output of subtask 1 of the argumentation problem has already been obtained. We look for 100% accuracy of span detection (successful segmentation of the argumentative text from the non-argumentative text) by Unified-AM and only on those spans do we measure F1 values for individual component type identification to compare our results with the other models (which assume subtask 1 is given). Lastly we calculate the macro-F1 score. Table 4.12 contains the results. We obtain the highest macro-F1 score of **89.18%** when we do not consider subtask 1. We have the highest individual F1 score for

Table 4.14: Individual F1 scores on the BIO labeling task

	Beginning (B) F1 score	**Continuation (I) F1 score**	**Non-Argumentative (O) F1 score**	**Macro F1 Score**
Augmented Corpus (Addition of New Paragraphs)	87.19	94.81	89.20	90.40
Augmented Corpus (Addition of New Paragraphs) Training Epochs: 1500-1600	85.90	94.78	88.97	89.88
Augmented Corpus (Addition of New Sentences as Paragraphs)	85.21	94.55	88.62	89.46

the premise tokens (**96.98%**) which boosts the macro F1 score considerably.

For major claim, premise, and claim, there are two different tags in the PE corpus, B: Beginning of a component and I: Continuation of a component. Non-Argumentative tokens are tagged as 'O' in the BIO scheme. We compare the component segmentation task (subtask 1) results with other works that have been mentioned in Eger et al. [10]. Table 4.13 shows the results between the models. We see that LSTM-ER has the highest macro-F1 score when we consider only the BIO labeling task. We obtain a similar macro-F1 score for the essay version of the PE corpus. In the original paragraph version of the PE corpus, we did not get the highest score when we consider this subtask. When we analyze the BIO labeling score on the augmented corpus, we find that if we train it for 1500-1600 epochs for which we get the highest C-F1 and R-F1 scores (see Table 4.9), the BIO F1 score decreases. We individually observe the F1 scores related to 'B' (Beginning), 'I' (Continuation) and 'O' (Non-Argumentative) and find that training for more epochs actually decreases the F1 score related to 'B' significantly. The other two components' F1 score do not change much. For the default training epochs (1000-1100), we obtain 90.40% macro-F1 score and the individual F1 score related to 'B' is 87.19%. On the other hand, this F1 score related to the 'B' component decreases to 85.90% when we train it for 1500-1600 epochs. The PE corpus does not contain that many 'B' components. Only about 4.05% of the whole corpus (see Table 3.1) has this component and that could be one of the reasons for the model to not get a high F1 score for this class while training for more epochs. Because we are trying to solve all of the class labels together with our unified representation and during training time, the model could get biased to reduce the loss and increase the accuracy for the highly available components. Table 4.14 shows the individual BIO labelling F1 scores on the paragraph version of the corpus.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this research work, we show that rather than using a complex stacked architecture, as that used by Eger et al. [10], for a problem which has different subtasks (where all the subtasks are related to each other), we can have a compact and unified representation of all of the sub-problems and can tackle it as a single problem with less complicated architectures. These sub-tasks are: segmenting the argument components; labelling each argument component as Major Claim, Claim, or Premise; determining which argumentation components are in a relationship which has been given by a distance value in this corpus; and classifying the stance of the relations between argument components. We have introduced a novel multi-label representation for solving the argumentation mining task for one linguistic genre, persuasive essays, which is exemplified by the Persuasive Essays (PE) corpus [42]. We have created a full representation of all the sub-tasks with a 33-dimensional vector.

We obtain an improved performance over Eger et al. [10] in recognizing the argument components and relations by jointly solving all of the subtasks. This improvement is achieved by introducing the Flair stacked embedding [2] to represent the text input. In addition to the biLSTM layers in the model, we introduce a multi-head attention layer to our final neural architecture which leads us to the highest accuracy on both the paragraph-level and essay-level versions of the PE corpus.

On the paragraph version of the PE corpus, we have the highest F1 score of **60%** where Eger et al. [10] has achieved 55% (see Table 4.4). We have obtained **58%** F1 score on the essay version of the corpus which has improved the previous result by **8** percentage points (see Table 4.5).

Observing that the imbalanced corpus may be creating problems for this model to learn certain underrepresented features of the corpus, we have used the standard technique of data

Table 5.1: Summary of the State-of-the-art Results Obtain by the Unified-AM Model Architecture and Comparison with the LSTM-ER Model of Eger et al. [10]

Paragraph Corpus							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	**68.03%**	**71.35**	**80.21**	**54.27**	**59.46**	**62.81**	**69.83**
LSTM-ER	61.67%	70.83	77.19	45.52	50.05	55.42	60.72
Essay Corpus							
Model	Token Accuracy	C-F1 (100%)	C-F1 (50%)	R-F1 (100%)	R-F1 (50%)	F1 (100%)	F1 (50%)
Unified-AM	**62.88%**	**67.78**	**76.20**	**48.24**	**52.49**	**58.01**	**64.35**
LSTM-ER	54.17%	66.21	73.02	29.56	32.72	40.87	45.19

augmentation to achieve further gains in performance. We have created two augmented versions of the PE training corpus by using different combinations of the n-grams that occur immediately before approximately two-thirds of the major claim components (see Section 4.5). The augmentations have been done on the paragraph version of the corpus. By using the augmentation methodology, we further improve the Unified-AM model's performance on the test set. We have obtained the highest token level accuracy, C-F1, R-F1, and the global F1 score (which is the combination of both C-F1 and R-F1 scores) on the paragraph version of the PE corpus by applying the augmentation techniques.

There are some limitations with regards to the model and the unified representation proposed here. In this thesis work, the argumentation structure prediction task is done in a supervised way where we need to have an annotated corpus. Without any labelling, we would be unable to train our model. Also, our novel unified representation is created specifically for the PE corpus. Although this representation can be modified based on other corpora, we have not tried our unified representation with other argumentation mining corpora that have become available. We have introduced some of the new corpora related to argumentation mining in Section 5.2.

We summarize the results our model architecture in Table 5.1. These new results are compared with the performance of that of Eger et al. [10] showing that our model has obtained new state-of-the-art results for all of the measures on both the paragraph and essay versions of the PE corpus. Shared parameter values across different subtasks enhanced the accuracy score and also the model's capability for accurate detection of components, relations and distance in the argumentation mining task.

5.2 Future Works

In this section, we describe some of the future research works and ideas that can be implemented to solve and detect argumentation structure contained in the texts.

5.2.1 Decoupled Solution Only to Detect Distance

We have introduced a novel multi-label representation for solving the argumentation task. Future work can decouple this 33-length vector in a unique way to solve the distance prediction task separately with the introduction of a second model. In this joint architecture, the first model could be used and trained for detecting argumentative components and relations between them, the second model (can be a graph neural network [39], or a convolutional neural network [16]) will be trained by extracting information of components and relations from the first model to detect only the distance between the detected components.

5.2.2 Introduction of Graph Neural Network to Represent Distance

Introduction of graph neural networks to represent the textual distance [38, 44] can improve the results further for detecting argumentation structure. We have observed relative distance values from -11 to +11 in the PE corpus between premise to claim through a supporting or attacking relation. If we can represent the whole argumentation problem separately as a graph structure where premise to claim will have an edge with a particular type (support or attack) during the training time, it can positively affect the learned parameter or weight values of the main neural architecture. While evaluating our model on the test set, we will not use the graph information (as it contains prediction labels as distance values). We will only let the main neural architecture model, which has been trained with the extra graph-like label information during the training step, to detect argumentation structure in the test corpus.

5.2.3 New Methodology for N-grams

We have tried several ways to incorporate information about certain n-grams (we have found a total of 108 n-grams that occur frequently before the major claim spans) into the model. The augmentation of the PE corpus on the paragraph level has led to the highest accuracy obtained on the test set of the PE corpus. However, we also tried several other approaches 4.2 regarding injecting the n-gram information into the model to boost the performance but it did not work well. In future, research works and experiments can be carried out based on the n-gram tokens.

The future research work can be based on collecting new n-gram tokens that occur before claim and premise spans or injecting n-gram information in a different and more efficient way.

5.2.4 Future Research on Other Available Argumentation Mining Corpus

Recently, Park and Cardie [29], Levy et al. [20], and Shnarch et al. [40] have published new benchmark argumentation mining corpora which will help researchers in this particular field to carry out further research and findings. Future work can include analysis, experiments, and evaluation of the unified representation (or a slightly modified representation for the different classes available in the new datasets) to solve the argumentation mining problem in these new standardized corpora.

5.2.5 Contextual Embedding

Many robust and efficient transformer models like BERT [9], RoBERTa [22], and other contextual embeddings [36] can be used to represent the textual inputs for the argumentation task. These bigger models may need fine-tuning for this downstream argumentation structure detection task.

5.2.6 Curriculum Learning

Curriculum learning (CL) [5] is a technique that can be used in the future research works to train the existing (Unified-AM) or new models. CL learns from the easier to label data samples progressing to the harder ones. CL has proven to be efficient and powerful to increase the generalization and convergence rate of different types of models in different machine learning fields (for example, in computer vision and natural language processing).

Bibliography

[1] Mahtab Ahmed, Muhammad Rifayat Samee, and Robert E. Mercer. Improving neural sequence labelling using additional linguistic information. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 650–657. IEEE, 2018.

[2] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, 2019.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[4] Jianzhu Bao, Chuang Fan, Jipeng Wu, Yixue Dang, Jiachen Du, and Ruifeng Xu. A neural transition-based model for argumentation mining. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6354–6364, 2021.

[5] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 41–48, 2009.

[6] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[7] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[8] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. Neural end-to-end learning for computational argumentation mining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11–22, 2017.

[11] Alfio Ferrara, Stefano Montanelli, and Georgios Petasis. Unsupervised detection of argumentative units through topic modeling techniques. In *Proceedings of the 4th Workshop on Argument Mining*, pages 97–107, 2017.

[12] Benjamin Heinzerling and Michael Strube. BPEmb: Tokenization-free pre-trained subword embeddings in 275 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[13] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

[14] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[17] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2020.

[18] Tatsuki Kuribayashi, Hiroki Ouchi, Naoya Inoue, Paul Reisert, Toshinori Miyoshi, Jun Suzuki, and Kentaro Inui. An empirical study of span representations in argumentation structure parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4691–4698, 2019.

[19] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, 2016.

[20] Ran Levy, Ben Bogin, Shai Gretz, Ranit Aharonov, and Noam Slonim. Towards an argumentative content search engine using weak supervision. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2066–2081, 2018.

[21] Anusha Lihala. attention-and-its-different-forms, March 2019.

[22] Yinhan Liu, Myle Ott, Naman Goyal, J Du, M Joshi, D Chen, O Levy, M Lewis, L Zettlemoyer, and V Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[23] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[25] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, 2016.

[26] Vlad Niculae, Joonsuk Park, and Claire Cardie. Argument mining with structured SVMs and RNNs. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 985–995, 2017.

[27] Christopher Olah. Understanding LSTM networks, August 2015.

[28] Raquel Mochales Palau and Marie-Francine Moens. Argumentation mining: the detection, classification and structure of arguments in text. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, pages 98–107, 2009.

[29] Joonsuk Park and Claire Cardie. A corpus of erulemaking user comments for measuring evaluability of arguments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 1623–1628, 2018.

[30] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318. PMLR, 2013.

[31] Andreas Peldszus. Towards segment-based recognition of argumentation structure in short texts. In *Proceedings of the First Workshop on Argumentation Mining*, pages 88–97, 2014.

[32] Andreas Peldszus and Manfred Stede. Joint prediction in MST-style discourse parsing for argumentation mining. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 938–948, 2015.

[33] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[34] Isaac Persing and Vincent Ng. End-to-end argumentation mining in student essays. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1384–1394, 2016.

[35] Isaac Persing and Vincent Ng. Unsupervised argumentation mining in student essays. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6795–6803, 2020.

[36] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[37] Peter Potash, Alexey Romanov, and Anna Rumshisky. Here's my point: Joint pointer architecture for argument mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1364–1373, 2017.

[38] Federico Ruggeri, Marco Lippi, and Paolo Torroni. Tree-constrained graph neural networks for argument mining. *arXiv preprint arXiv:2110.00124*, 2021.

[39] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[40] Eyal Shnarch, Carlos Alzate, Lena Dankin, Martin Gleize, Yufang Hou, Leshem Choshen, Ranit Aharonov, and Noam Slonim. Will it blend? Blending weak and strong labeled data in a neural network for argumentation mining. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 599–605, 2018.

[41] Christian Stab and Iryna Gurevych. Annotating argument components and relations in persuasive essays. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1501–1510, 2014.

[42] Christian Stab and Iryna Gurevych. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659, 2017.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, 2017.

[44] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5171–5181, 2018.

Appendix A

An Example of an Annotated Essay

```
1 Should  O
2 students  O
3 be  O
4 taught  O
5 to  O
6 compete  O
7 or  O
8 to  O
9 cooperate  O
10 ?  O
11 It  O
12 is  O
13 always  O
14 said  O
15 that  O
16 competition  O
17 can  O
18 effectively  O
19 promote  O
20 the  O
21 development  O
22 of  O
23 economy  O
24 .  O
```

```
25 In  O
26 order  O
27 to  O
28 survive  O
29 in  O
30 the  O
31 competition  O
32 ,  O
33 companies  O
34 continue  O
35 to  O
36 improve  O
37 their  O
38 products  O
39 and  O
40 service  O
41 ,  O
42 and  O
43 as  O
44 a  O
45 result  O
46 ,  O
47 the  O
48 whole  O
49 society  O
50 prospers  O
51 .  O
52 However  O
53 ,  O
54 when  O
55 we  O
56 discuss  O
57 the  O
58 issue  O
59 of  O
60 competition  O
```

```
61 or  O
62 cooperation  O
63 ,  O
64 what  O
65 we  O
66 are  O
67 concerned  O
68 about  O
69 is  O
70 not  O
71 the  O
72 whole  O
73 society  O
74 ,  O
75 but  O
76 the  O
77 development  O
78 of  O
79 an  O
80 individual  O
81 ’  O
82 s  O
83 whole  O
84 life  O
85 .  O
86 From  O
87 this  O
88 point  O
89 of  O
90 view  O
91 ,  O
92 I  O
93 firmly  O
94 believe  O
95 that  O
96 we  B-MajorClaim
```

```
97 should  I-MajorClaim
98 attach  I-MajorClaim
99 more  I-MajorClaim
100 importance  I-MajorClaim
101 to  I-MajorClaim
102 cooperation  I-MajorClaim
103 during  I-MajorClaim
104 primary  I-MajorClaim
105 education  I-MajorClaim
106 .  O
107 First  O
108 of  O
109 all  O
110 ,  O
111 through  B-Claim:For
112 cooperation  I-Claim:For
113 ,  I-Claim:For
114 children  I-Claim:For
115 can  I-Claim:For
116 learn  I-Claim:For
117 about  I-Claim:For
118 interpersonal  I-Claim:For
119 skills  I-Claim:For
120 which  I-Claim:For
121 are  I-Claim:For
122 significant  I-Claim:For
123 in  I-Claim:For
124 the  I-Claim:For
125 future  I-Claim:For
126 life  I-Claim:For
127 of  I-Claim:For
128 all  I-Claim:For
129 students  I-Claim:For
130 .  O
131 What  B-Premise:-1:Support
132 we  I-Premise:-1:Support
```

```
133 acquired  I-Premise:-1:Support
134 from  I-Premise:-1:Support
135 team  I-Premise:-1:Support
136 work  I-Premise:-1:Support
137 is  I-Premise:-1:Support
138 not  I-Premise:-1:Support
139 only  I-Premise:-1:Support
140 how  I-Premise:-1:Support
141 to  I-Premise:-1:Support
142 achieve  I-Premise:-1:Support
143 the  I-Premise:-1:Support
144 same  I-Premise:-1:Support
145 goal  I-Premise:-1:Support
146 with  I-Premise:-1:Support
147 others  I-Premise:-1:Support
148 but  I-Premise:-1:Support
149 more  I-Premise:-1:Support
150 importantly  I-Premise:-1:Support
151 ,  I-Premise:-1:Support
152 how  I-Premise:-1:Support
153 to  I-Premise:-1:Support
154 get  I-Premise:-1:Support
155 along  I-Premise:-1:Support
156 with  I-Premise:-1:Support
157 others  I-Premise:-1:Support
158 .  O
159 During  B-Premise:-2:Support
160 the  I-Premise:-2:Support
161 process  I-Premise:-2:Support
162 of  I-Premise:-2:Support
163 cooperation  I-Premise:-2:Support
164 ,  I-Premise:-2:Support
165 children  I-Premise:-2:Support
166 can  I-Premise:-2:Support
167 learn  I-Premise:-2:Support
168 about  I-Premise:-2:Support
```

```
169 how  I-Premise:-2:Support
170 to  I-Premise:-2:Support
171 listen  I-Premise:-2:Support
172 to  I-Premise:-2:Support
173 opinions  I-Premise:-2:Support
174 of  I-Premise:-2:Support
175 others  I-Premise:-2:Support
176 ,  I-Premise:-2:Support
177 how  I-Premise:-2:Support
178 to  I-Premise:-2:Support
179 communicate  I-Premise:-2:Support
180 with  I-Premise:-2:Support
181 others  I-Premise:-2:Support
182 ,  I-Premise:-2:Support
183 how  I-Premise:-2:Support
184 to  I-Premise:-2:Support
185 think  I-Premise:-2:Support
186 comprehensively  I-Premise:-2:Support
187 ,  I-Premise:-2:Support
188 and  I-Premise:-2:Support
189 even  I-Premise:-2:Support
190 how  I-Premise:-2:Support
191 to  I-Premise:-2:Support
192 compromise  I-Premise:-2:Support
193 with  I-Premise:-2:Support
194 other  I-Premise:-2:Support
195 team  I-Premise:-2:Support
196 members  I-Premise:-2:Support
197 when  I-Premise:-2:Support
198 conflicts  I-Premise:-2:Support
199 occurred  I-Premise:-2:Support
200 .  O
201 All  B-Premise:-3:Support
202 of  I-Premise:-3:Support
203 these  I-Premise:-3:Support
204 skills  I-Premise:-3:Support
```

```
205 help  I-Premise:-3:Support
206 them  I-Premise:-3:Support
207 to  I-Premise:-3:Support
208 get  I-Premise:-3:Support
209 on  I-Premise:-3:Support
210 well  I-Premise:-3:Support
211 with  I-Premise:-3:Support
212 other  I-Premise:-3:Support
213 people  I-Premise:-3:Support
214 and  I-Premise:-3:Support
215 will  I-Premise:-3:Support
216 benefit  I-Premise:-3:Support
217 them  I-Premise:-3:Support
218 for  I-Premise:-3:Support
219 the  I-Premise:-3:Support
220 whole  I-Premise:-3:Support
221 life  I-Premise:-3:Support
222 .  O
223 On  O
224 the  O
225 other  O
226 hand  O
227 ,  O
228 the  B-Premise:1:Support
229 significance  I-Premise:1:Support
230 of  I-Premise:1:Support
231 competition  I-Premise:1:Support
232 is  I-Premise:1:Support
233 that  I-Premise:1:Support
234 how  I-Premise:1:Support
235 to  I-Premise:1:Support
236 become  I-Premise:1:Support
237 more  I-Premise:1:Support
238 excellence  I-Premise:1:Support
239 to  I-Premise:1:Support
240 gain  I-Premise:1:Support
```

```
241 the  I-Premise:1:Support
242 victory  I-Premise:1:Support
243 .  O
244 Hence  O
245 it  O
246 is  O
247 always  O
248 said  O
249 that  O
250 competition  B-Claim:Against
251 makes  I-Claim:Against
252 the  I-Claim:Against
253 society  I-Claim:Against
254 more  I-Claim:Against
255 effective  I-Claim:Against
256 .  O
257 However  O
258 ,  O
259 when  B-Premise:2:Support
260 we  I-Premise:2:Support
261 consider  I-Premise:2:Support
262 about  I-Premise:2:Support
263 the  I-Premise:2:Support
264 question  I-Premise:2:Support
265 that  I-Premise:2:Support
266 how  I-Premise:2:Support
267 to  I-Premise:2:Support
268 win  I-Premise:2:Support
269 the  I-Premise:2:Support
270 game  I-Premise:2:Support
271 ,  I-Premise:2:Support
272 we  I-Premise:2:Support
273 always  I-Premise:2:Support
274 find  I-Premise:2:Support
275 that  I-Premise:2:Support
276 we  I-Premise:2:Support
```

```
277 need  I-Premise:2:Support
278 the  I-Premise:2:Support
279 cooperation  I-Premise:2:Support
280 .  O
281 The  O
282 greater  O
283 our  O
284 goal  O
285 is  O
286 ,  O
287 the  O
288 more  O
289 competition  O
290 we  O
291 need  O
292 .  O
293 Take  B-Premise:1:Support
294 Olympic  I-Premise:1:Support
295 games  I-Premise:1:Support
296 which  I-Premise:1:Support
297 is  I-Premise:1:Support
298 a  I-Premise:1:Support
299 form  I-Premise:1:Support
300 of  I-Premise:1:Support
301 competition  I-Premise:1:Support
302 for  I-Premise:1:Support
303 instance  I-Premise:1:Support
304 ,  I-Premise:1:Support
305 it  I-Premise:1:Support
306 is  I-Premise:1:Support
307 hard  I-Premise:1:Support
308 to  I-Premise:1:Support
309 imagine  I-Premise:1:Support
310 how  I-Premise:1:Support
311 an  I-Premise:1:Support
312 athlete  I-Premise:1:Support
```

```
313 could  I-Premise:1:Support
314 win  I-Premise:1:Support
315 the  I-Premise:1:Support
316 game  I-Premise:1:Support
317 without  I-Premise:1:Support
318 the  I-Premise:1:Support
319 training  I-Premise:1:Support
320 of  I-Premise:1:Support
321 his  I-Premise:1:Support
322 or  I-Premise:1:Support
323 her  I-Premise:1:Support
324 coach  I-Premise:1:Support
325 ,  I-Premise:1:Support
326 and  I-Premise:1:Support
327 the  I-Premise:1:Support
328 help  I-Premise:1:Support
329 of  I-Premise:1:Support
330 other  I-Premise:1:Support
331 professional  I-Premise:1:Support
332 staffs  I-Premise:1:Support
333 such  I-Premise:1:Support
334 as  I-Premise:1:Support
335 the  I-Premise:1:Support
336 people  I-Premise:1:Support
337 who  I-Premise:1:Support
338 take  I-Premise:1:Support
339 care  I-Premise:1:Support
340 of  I-Premise:1:Support
341 his  I-Premise:1:Support
342 diet  I-Premise:1:Support
343 ,  I-Premise:1:Support
344 and  I-Premise:1:Support
345 those  I-Premise:1:Support
346 who  I-Premise:1:Support
347 are  I-Premise:1:Support
348 in  I-Premise:1:Support
```

```
349 charge  I-Premise:1:Support
350 of  I-Premise:1:Support
351 the  I-Premise:1:Support
352 medical  I-Premise:1:Support
353 care  I-Premise:1:Support
354 .  O
355 The  O
356 winner  O
357 is  O
358 the  O
359 athlete  O
360 but  O
361 the  O
362 success  O
363 belongs  O
364 to  O
365 the  O
366 whole  O
367 team  O
368 .  O
369 Therefore  O
370 without  B-Claim:For
371 the  I-Claim:For
372 cooperation  I-Claim:For
373 ,  I-Claim:For
374 there  I-Claim:For
375 would  I-Claim:For
376 be  I-Claim:For
377 no  I-Claim:For
378 victory  I-Claim:For
379 of  I-Claim:For
380 competition  I-Claim:For
381 .  O
382 Consequently  O
383 ,  O
384 no  O
```

```
385 matter  O
386 from  O
387 the  O
388 view  O
389 of  O
390 individual  O
391 development  O
392 or  O
393 the  O
394 relationship  O
395 between  O
396 competition  O
397 and  O
398 cooperation  O
399 we  O
400 can  O
401 receive  O
402 the  O
403 same  O
404 conclusion  O
405 that  O
406 a  B-MajorClaim
407 more  I-MajorClaim
408 cooperative  I-MajorClaim
409 attitudes  I-MajorClaim
410 towards  I-MajorClaim
411 life  I-MajorClaim
412 is  I-MajorClaim
413 more  I-MajorClaim
414 profitable  I-MajorClaim
415 in  I-MajorClaim
416 one  I-MajorClaim
417 '  I-MajorClaim
418 s  I-MajorClaim
419 success  I-MajorClaim
420 .  O
```

Appendix B

The collected n-gram list

0. In conclusion ,
1. I think that
2. I believe that
3. I reckon that
4. I agree that
5. I support that
6. I feel that
7. I contend that
8. I suppose that
9. I convinced that
10. I firmly believe that
11. I strongly believe that
12. I would contend that
13. I truly believe that
14. I agree with that
15. I completely agree that
16. I strongly agree that
17. I would state that
18. I personally think that
19. I feel certain that
20. I do think that
21. I would conclude that
22. I will conclude that
23. I am convinced that
24. I always believe that

25. I would maintain that
26. I definitely feel that
27. I really believe that
28. I agree to that
29. I would argue that
30. I nevertheless believe that
31. I still believe that
32. I believe that that
33. I definitely believe that
34. I strongly feel that
35. I totally agree that
36. I would say that
37. I personally suppose that
38. I totally believe that
39. I completely argue that
40. I do believe that
41. I really believe that that
42. I believe that without that
43. I certainly can say that
44. I want to say that
45. I advocate the idea that
46. I would definitely agree that
47. I personally agree with that
48. I am strongly convinced that
49. I want to mentioned that
50. I hold the contention that
51. I strongly agree with that
52. I am utterly convinced that
53. I tend to think that
54. I would like to say that
55. I agree with that idea that
56. I strongly affirm the conclusion that
57. I agree with the view that
58. I favor the former ; that
59. I favor the latter ; that
60. I agree to that fact that

61. I can say for certain that
62. I ' m absolutely sure that
63. I am inclined to believe that
64. I strongly agree with notion that
65. I am of the view that
66. I admittedly and strongly agree that
67. I agree to the argument that
68. From my point of view ,
69. from my point of view ,
70. All in all ,
71. In my point of view ,
72. As far as I am concerned ,
73. To conclude ,
74. In my opinion ,
75. in my opinion ,
76. In my personal opinion ,
77. By way of conclusion ,
78. In my perspective ,
79. in my perspective ,
80. To my view ,
81. to my view ,
82. In my view ,
83. in my view ,
84. To sum up ,
85. From my perspective ,
86. from my perspective ,
87. I think ,
88. I believe ,
89. In sum ,
90. In summary ,
91. I still hold the firm view that
92. I once again restate my position that
93. Therefore ,
94. Thus ,
95. Hence ,
96. First of all ,

97. Moreover ,
98. To begin with ,
99. Last but not least ,
100. Firstly ,
101. Secondly ,
102. Thirdly ,
103. Lastly ,
104. First ,
105. Second ,
106. Admittedly ,
107. Furthermore ,

Curriculum Vitae

Name:	Muhammad Tawsif Sazid
Post-Secondary Education and Degrees:	The University of Western Ontario London, ON 2021 - 2022 M.Sc.
	Military Institute of Science and Technology-Bangladesh Dhaka, Bangladesh 2015 - 2018 B.Sc.
Honours and Awards:	Western Graduate Research Scholarship 2021-2022
Related Work Experience:	Teaching Assistant The University of Western Ontario 2021 - 2022
	Research Assistant The University of Western Ontario 2021-2022