Dissertation Type: Research

DEPARTMENT OF ENGINEERING MATHEMATICS

Mining Argument Relations using Transformers

Cyrus Dobbs

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Science in the Faculty of Engineering.

Monday 13th September, 2021

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisor, Dr Edwin Simpson, for his excellent guidance throughout this project.

Abstract

This thesis explores the use of Transformer-based architectures for argument mining (AM), choosing to adapt the latest AM Transformer-based model [19] for use with the Consumer Debt Collection Practices (CDCP) dataset [25].

The thesis begins by continuing work on an existing end-to-end sequential pipeline architecture [19] by first experimenting with concatenating additional encoded information to the Transformer's vector representation of pairs of argument components. This leads to a slight improvement over using the argument component representation alone.

In order to find out what architectural features of the state-of-the-art ResAttArg [8] model are responsible for it's unmatched performance some of it's features are replicated in isolation, with the exception of it's use of static GloVe [28] embeddings, which is replaced with a Transformer-based embedding method. Previous work, including ResAttArg, has often jointly learnt the tasks of Relation Classification (RC) and Argument Component Classification. Joint learning is compared to the existing end-to-end sequential pipeline architecture [19] by also adapting the RC part of the Transformer based architecture to support joint learning.

Modifications are made to RBERT [40], for the CDCP dataset and AM task. RBERT is an existing model that uses a whole document, instead of a pair of sequences as input to the Transformer. This provides the language model with much more contextual information to learn from and make predictions. RBERT also utilises additional Transformer hidden states in order to make classifications, which my analysis suggests is a big factor in its performance improvement on more standard sequence classification techniques. My research provides some insight into how RBERT copes with dealing with longer range relations; a point that is put forward for future work in the original RBERT paper [40]. Joint learning is incorporated into the RBERT architecture which results in a new state-of-the-art model for AM, RBERT-JL.

The final RBERT-JL model is trained and evaluated using the CDCP dataset where it achieves 46.46% F1-score for relation classification and 81.43% F1-score for component classification. This is an improvement on the current state-of-the-art [8] for both relation and component classification by 3.51% and 2.72% F1-score, respectively. Quantitative and qualitative analysis on the test set results is carried out and supported by ideas for future extensions. The project proves that, like many other NLP tasks, AM can also benefit hugely from modern pre-trained language models such as BERT.

A summary of main achievements:

- 120 hours was spent collecting material and researching about the current state of AM and several methods that could lead to improvements, namely Transformers, MTL and RL.
- 2300 lines of *Python* code was written.
- The work of Mayer et al. [19] was developed further by experimenting with adding additional information to the Transformers vector representation.
- Parts of the ResAttArg architecture was dissected and reproduced to try to find out what made it work so well.
- RBERT was modified for AM and improved through joint learning.
- Evaluation of the completed modifications was carried out.
- Final results were compared to existing methods, resulting in state-of-the-art results on the CDCP dataset.

Supporting Technologies

- Python was the primary language used throughout this project. The following Python packages were used:
  - PyTorch for creating and training neural networks
  - Transformers for its BERT and various other implementations
  - Scikit-Learn for metric calculation
  - Tensorboard for viewing model metrics
  - Pandas, PyPlot, NumPy and Jupyter were also used for exploration of result data
- This work was carried out using the computational facilities of the Advanced Computing Research Centre, University of Bristol[1]
- Mayer et al.'s [19] Python repository was built upon during this work and much of it still remains in the final submission.
- Monologg's R-BERT Python implementation[2] was built upon during this work
- LaTeX was used to format the thesis, via the online service *Overleaf*.

[1] http://www.bristol.ac.uk/acrc/
[2] https://github.com/monologg/R-BERT

Notation and Acronyms

AM	:	Argument Mining
RC	:	Relation Classification
RL	:	Reinforcement Learning
SVM	:	Support Vector Machine
RNN	:	Recurrent Neural Network
DQN	:	Deep Q-Network
ATS	:	Automatic Text Summarization
NLP	:	Natural Language Processing
CDCP	:	Consumer Debt Collection Practices
LSTM	:	Long Short-Term Memory
GRU	:	Gated Recurrent Unit
CRF	:	Condition Random Fields
BERT	:	Bidirectional Encoder Representations from Transformers
	$\vdots$	
$\mathbb{U}$	:	Universal Set
$\cap$	:	Union
$\neg$	:	Not

Chapter 1

Introduction

1.1 Argument Mining Overview

With vast amounts of unstructured textual data now being collected and stored, it is increasingly valuable to be able to automatically extract structured information from text. The ability to automatically extract argumentative information, in the form of *claims* and *evidence* from documents of various domains and form summaries of arguments by connecting argument components via *support* and *attack* relations in a graph structure 1.1 could have a plethora of applications.

The motivation for AM is to help users understand the reasoning behind controversial claims, especially when expressed by multiple people across many different documents, comments or posts. Some domains could include the use of political debate and court proceeding transcripts as well as digital collections of news articles and opinion pieces. It is also a task that is heavily linked with other popular topics in AI such as fact checking and misinformation detection and could even serve as a method of explaining how machines reason in order to make decisions.

The task of AM continues to be a growing area of research spanning artificial intelligence, machine and deep learning, NLP, and computation linguistics and can be split up into 4 subtasks:

1. Component Segmentation: The detection and extraction of non-intersecting token sequences that make up argument components. [27], [35]

2. Component Classification: The classification of argument components. (e.g *claims* and *evidence*) [25], [7], [8]

3. Link Prediction: The prediction of links between argument components that make up the resulting argument structure. [25], [7], [8]

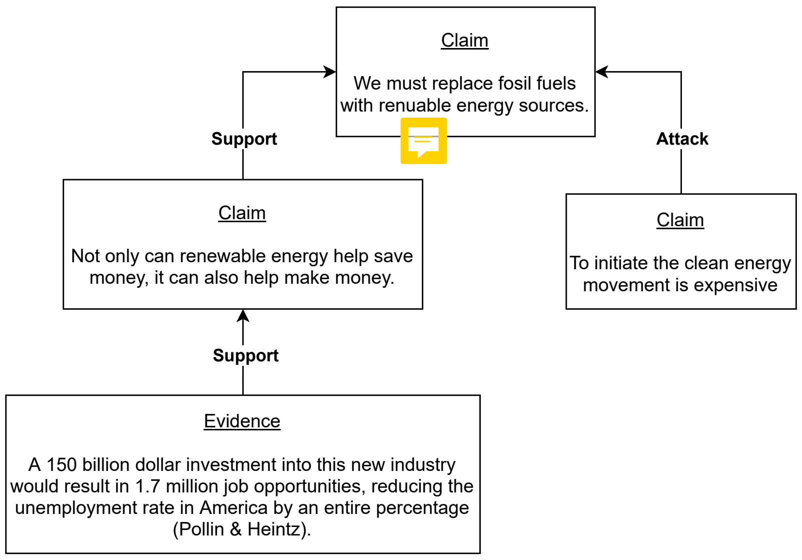

Claim
We must replace fosil fuels with renuable energy sources.
Support
Attack
Claim
Not only can renewable energy help save money, it can also help make money.
Claim
To initiate the clean energy movement is expensive
Support
Evidence
A 150 billion dollar investment into this new industry would result in 1.7 million job opportunities, reducing the unemployment rate in America by an entire percentage (Pollin & Heintz).

Figure 1.1: Example of argument structure with argument components and **relations** [3]

	Neoplasm	Glaucoma	Mixed
Approach			
Transformer-based			
BioBERT	64	58	61
SciBERT	68	62	**69**
RoBERTa	67	66	67
Residual network			
ResAttArg+Ensemble	**70.92**	**68.40**	67.66

Table 1.1: Latest relation classification macro averaged F_1 results on AbstRCT

4. Relation Classification: The classification of existing links between argument components (e.g *support* and *attack*).

AM has been researched using argumentative texts from a range of domains such as persuasive essays [35], medical trials [19], online comments [25], legal texts [23], [10] and Wikipedia articles [17].

1.2 The State of Argument Mining

Much of the work of argument component classification and relation classification was previously carried out using very domain specific models that use highly engineered and domain specific feature sets. Methods have included the use of Naive Bayes classifiers [23], [26], LDA [24], SVMs [25], [22] and RNNs [35] and various other classifiers [36]. However, recent advances and widespread adoption of deep learning has resulted in more general purpose applications [19], [7], [8] that make use of static word embeddings such as GloVe [28], ELMo [29] and contextualised embeddings such as BERT [4].

Many NLP tasks have benefited from the mechanism of Attention [38], allowing models to automatically weigh up the relevance of different regions of a sequence, and AM is no exception. The current state-of-the-art method from Galassi [8] uses attention in combination with a bi-LSTM and a residual network. They also make use of ensemble techniques as well as using a single internal representation for multiple classifiers to jointly classify both the argument components and the types of relations. In separate work, Mayer [19] evaluates various BERT Transformers in an argument mining pipeline consisting of two sequential learning tasks: i) argument segmentation and component classification via sequence labelling and ii) RC via sequence classification. Both methods are evaluated and compared on the Randomized Control Trials (AbstRCT) [19] dataset[1], with 3 test sets, Neoplasm, Glaucoma and Mixed. A sample of the best results for the two papers is presented in 1.1. However, Galassi also evaluates on the UKP and CDCP datasets, which will be revisited.

1.3 Argument Mining Challenges

There are many challenges within the field of AM. The most immediate is the lack of consistently annotated, general purpose, sizeable datasets. Currently, they are typically domain specific and often use conflicting conventions for component and relation classes. The need for a common argumentative framework is clear and research into crowd-sourcing [34] and automatic extension of existing datasets is ongoing.

Another major challenge is the inherent difficulty of the task [32] suggest that world, common-sense and domain knowledge play a pivotal role in being effective at AM and this knowledge is difficult to instil in machines for specific domains and currently impossible in a general world scale. A Transformer that is employed through this thesis name BERT (which stands for Bidirectional Encoder Representations from Transformers) helps to combat this issue in part by being pre-trained in an unsupervised setting on the Wikipedia and BooksCorpus [41] datasets that total over 3 million words.

1.4 The CDCP Dataset

Mayer et al.'s paper is the only existing research using BERT for AM, which this thesis builds upon and evaluates using the more frequently used Consumer Debt Collection Practices (CDCP) dataset. This

[1]https://gitlab.com/tomaye/abstrct/

corpus is a collection of user generated comments from an eRulemaking website[2]. CDCP does not contain as much specialist language and is also more widely used in previous papers, including the current state-of-the-art [8]. CDCP contains annotated spans and categories for each component and previous work has performed component and relation classification [8, 7, 25]).

Link types consist of REASON (97%) and EVIDENCE (3%). There are five argument component classes: POLICY (17%) and VALUE (45%) contain subjective information in contrast to objectiveTESTIMONY (21%) and FACT (16%). A small number of components are referred to as REFERENCE (1%) and contain URLs and citations.

Because CDCP is made of user-generated comments from an online setting the argument examples frequently deviate from common argument structure convention seen in more formal settings [11]. This is likely to make working with this data harder than working with, for example, legal texts which often follow similar patterns in argument structure. Additionally, only 3% of possible relations are present (from over 43,000 possible pairs of components) and 28% of documents do not contain a single link, making the task of relation classification even more difficult due to the sparse nature of the data.

1.5 Project Aims and Contributions

The thesis begins by adapting the existing Transformer-based method by Mayer *et al.*[19] by adding additional information to the Transformer's vector representation of the pair of argument component token sequences. The motivation for this is to bring Transformer-based methods in line with Gallasi's state of the art method, which utilises a 10-bit binary encoding of the distance between components. Mayer suggests that positional information is not relevant for the AbstRCT dataset. However, Galassi [7] contradicts this when using the CDCP dataset, providing evidence that position is an important consideration in the classification of relations. While Gallasi suggests that distance is an important piece of information for an AM model due to 70% of relations being between adjacent components, they do not go into detail about the effect distance information has on model performance. This thesis aims to investigate this by using the same distance encoding method with a Transformer-based model.

While adding the addition information to BERT's internal representation did improve performance, it was still considerably behind the state-of-the-art ResAttArg model. In order to make assumptions about what made ResAttArg work so well parts of it's architecture are reproduced in isolation with the exception of its encoding and embedding layers, which are replaced by a BERT Transformer.

The main research hypothesis of this research is that utilising additional contextual information outside of the argument components will allow the model to make more informed argument relation classifications. One benefit of large language models such as BERT is their ability to utilise distant contextual information via the use of self-attention. RBERT [40], a BERT based model that takes a

BERT for sequence classification input

[CLS] Requiring a validation notice from the owner of the debt should be optional on the creditorś part.[SEP]More notices from the creditor are useless and expensive.

RBERT input example

[Entity1] Requiring a validation notice from the owner of the debt should be optional on the creditorś part.[/Entity1] The FDCPA requires the collection agency to send their first notice after the debt is assigned. [Entity2] More notices from the creditor are useless and expensive.[/Entity2]

Relation

Entity 1: Requiring a validation notice from the owner of the debt should be optional on the creditorś part.

↑ REASON ↑

Entity 2: More notices from the creditor are useless and expensive.

Figure 1.2: BERT and RBERT input examples

[2]http://regulationroom.org/

whole document as input, is used to test the research hypothesis. The input is altered to include tags around each of the entities (argument components in the case of AM) in order to later extract additional hidden states from the final BERT embedding. A previous paper to RBERT by [16] introduced the use of entity tags, allowing attention mechanisms to alter their representation according to surrounding context. RBERT built upon this by replacing the Bi-LSTM network and attention layers with BERT.

From my understanding, this is the first piece of work that incorporates wider document context as input to Transformer-based model in an AM setting. However, the method of placing tags around entities within a sequence in order to classify relations between them has been used for other relation classification tasks. Previous work on relation classification has been developed and bench-marked using the SemEval-2010 Task 8 dataset, consisting of relations between words in a sentence. My research explores the results when the data is expanded to larger documents, with sentence long argument components as entities and relations that can span the distance of multiple components, as is the case for the majority of data from argumentative settings. This work aims to provide evidence of the effectiveness of RBERT at longer distance relationships, in line with suggested future work in the original RBERT paper [40].

The final research proposal is to reformat the RC task from classification of individual, independent pairs of components to a series of actions that each adds a new argument component pair to the argument summary for that document via reinforcement learning. This could be used to fine tune the Transformer model further by giving it a reward depending on how similar the produced summary is compared to the gold standard. This extra knowledge, fed back to the Transformer at the end of processing a single document, should add information that is unobtainable when treating the links independently, as in a supervised setting. When evaluating the Transformer-based method [19], Mayer mentions that the model tends to believe many connections are plausible, leading to components being linked to too many components. One hypothesis is that by further fine-tuning the model in an RL setting, over-estimation of links can be reduced.

The contributions of my work can be summarized as:

1. Evidence is provided that standard sequence classification is not an effective approach to the AM sub-task of relation classification.

2. RBERT, a Transformer-based model built for classifying semantic relations between small entities, is shown to be able to be extended to larger documents containing longer distance argument relations between much larger components. The model obtains 46.46% F1-score on the CDCP dataset, outperforming existing state-of-the-art models. This proves that Transformer-based models outperform previous deep learning architectures and that additional document context and use of additional BERT hidden states are useful when classifying relations between argument components.

3. Evidence is provided that joint learning benefits both relation and component classification for AM as RBERT-JL outperforms all existing methods on the CDCP dataset. The model achieves 81.43% F1-score for component classification, outperforming the existing state-of-the-art ResAttArg [8] model by 2.72% F1-score. It also outperforms the BERT for sequence tagging pipeline method documented by Mayer et al. [19] by 3.68% F1-score.

Chapter 2

Background

Parsing Argumentation Structures in Persuasive Essays
Stab *et al.* [35] present an end-to-end approach to AM that solves the four subtasks independently. It then enforces valid structure using a feature-based integer linear programming model. They present a new dataset of argumentative essays (UKP). Each essay is made up of argument components labelled as [MAJORCLAIM, CLAIM, PREMISE] and relations labelled as [SUPPORT, ATTACK]. The following characteristics of the dataset that make it less complex than the CDCP corpus: arguments are limited to a tree structure (an argument component can only have a single outgoing link but numerous incoming links) and each argument structure can only be contained in a single paragraph. The relevance of Stab's work is simply that it was the first complete end-to-end approach to the full AM task and many subsequent pieces of work build on and refer back to it.

Argument Mining with Structured SVMs and RNNs
Niculae et al [25], work with both the UKP dataset [35] as well as present their own CDCP dataset. They contribute a novel factor graph [13] model that can represent arguments with more complex structures than directed trees, can consider second order link structures (grandchildren etc), and also enforce arbitrary constraints such as transitivity. In a tree, a node can only have a single parent, but this is likely not the case in real-world argument structures. They use the framework of structured learning applied to Structured SVMs [37] and Recurrent Neural Networks [31] in a supervised setting. The models jointly learn component classification and RC. They test the models using factor graphs of varying numbers of components and achieve even state-of-the-art link prediction performance on the UKP dataset [35]. Their work is directly relevant because they propose the dataset that is used in this thesis while also designing an architecture that completes the same tasks and produces comparable results to the techniques explored in this work. This makes their work a good point of comparison to feature engineered, pre-Attention techniques, when evaluating the final results.

Here's My Point: Joint Pointer Architecture for Argument Mining
Potash *et al.* [30] present the first neural network-based approach to link prediction. They address the tasks of link prediction and component classification jointly using pointer networks [39] (and an RNN with attention). Like [35, 25], they also assume that the argument components have been identified and that the arguments follow a tree structure, i.e an argument component can only have a single outgoing link but numerous incoming links. Their results suggest that joint modelling of link prediction and relation classification is essential for competitive performance and outperform the existing heavily feature engineered and dataset specific models.

Neural End-to-End Learning for Computational Argumentation Mining
Egar et al [5] compare AM as token-based dependency parsing and as a token-based sequence tagging problem. They find that sequence tagging is superior to dependency parsing, especially at being able to catch long-range dependencies. They also conclude that joint learning subtasks in a multi-task architecture leads to improvements in performance.

Argumentative Link Prediction using Residual Networks and Multi-Objective Learning
Galassi et al [7] propose a residual network [9] architecture for AM. While He et al [9] focus on residual CNNs for image recognition, Galassi uses a fully connected architecture paired with an LSTM to jointly

classify argument components and predict argument links, with a focus on the latter. Residual networks differ from standard feed-forward networks by using shortcuts between distant neurons, where the standard only links subsequent layers. Galassi avoids the use of domain-specific features like those used by Niculae [25] and Stab [35], aiming to produce a model more generally applicable to other domains. While Niculae [25] predict link labels as [ON, OFF] and proposition labels as [REFERENCE, TESTIMONY, FACT, VALUE, POLICY], Galassi goes further by also predicting relation types [SUPPORT, ATTACK, SUPPORTEDBY, ATTACKEDBY] of active links. Each token within a proposition is transformed into an input sequence of 153 (number of tokens in the largest proposition) 300-dimensional embeddings via GloVe [28]. Dimensionality reduction is then carried out via deep embedders, resulting in a feature of size 50 for that proposition. The source, target and separation distance are then concatenated together and used as input to the residual network. Three independent softmax layers make up the final layers of the architecture, used to predict the relation, source and target labels. The model additionally makes use of dropout and batch-normalization as regularization techniques. Galassi also carries out useful preliminary analysis of the CDCP dataset and finds that 70% of links are between adjacent components - suggesting that component position could play an important factor in RC. They achieve then state-in-the-art results on the CDCP dataset.

Multi-Task Attentive Residual Networks for Argument Mining
Building on their previous work from 2018 [7], Galassi's main additions are an attention [38] module and ensemble learning that improve on their previous results by an average of ˜0.5% for RC and over 13% for component classification. They continue to use general-purpose embeddings with a focus on high generalization to different datasets. They test their model over four datasets, CDCP, AbstRCT, UKP and DrInventor with what they claim as similar state-of-the-art results but with smaller model size and therefor less environmental impact. Both of Gallassi's papers are highly relevant to this work and will be used to evaluate the results along side theirs. Their work provides useful insights that help guide the research objectives of this work, such as their analysis of positional information and insights into the nature of the CDCP dataset. They mention the need to create a hand-crafted parser that pre-processes out irregularities within documents (particularly present in the CDCP dataset of online user created comments) before embedding via GloVe that may have some relevance to how pre-processing of the data is carried out for use with BERT.

Transformer-based Argument Mining for Healthcare Applications
Mayer *et al.* [19] approach AM in the domain of health-care, suggesting that evidence analysis can support clinicians in providing the the best course of action in different scenarios. They extend the AbstRCT dataset, resulting in over 4000 argument components and 2600 relations on different diseases. They present a decoupled pipeline approach to end-to-end AM, in contrast to MTL via a single model with 3 softmax layers for classification in [8, 7]. They combine deep bidirectional transformers [4] with different neural network architectures such as LSTM, GRU and CRFs.

They merge component segmentation and classification as a sequence tagging task using the BIO-tagging scheme, tagging each token as either the **B**eginning (first token) of a compoennt, **I**nside a component or **O**utside of a component. They compare static embeddings GloVe [28] and extvec [12] with dynamically generated embeddings ELMo [29] and BERT [4]. Their results show that the best architecture for sequence tagging is the fine-tuning of a BERT model combined with a GRU+CRF. While this thesis does not focus on improving the component classification task from [19], its results directly feed into the RC task, for which this improvement is the main aim. It is expected that a similar sequence tagging set up is also likely to provide the best results for extension using the CDCP dataset.

Valid tag sequences are taken as components, with pairs used as input to a three class sequence classifier for RC [SUPPORT, ATTACK, NORELATION]. They evaluate two options for relation classification: i) classifying all possible argument component combinations or ii) predicting plausible pairs of components and then only classifying the relation of such pairs. They find that the first option performs better, with the best results using i) seeing average improvement of ˜8% over the best results using ii).

These results are used to reduce the scope of the extension of Mayer's work by solely focusing on jointly modeling the relations as in i). In summary, Mayer uses a pipeline of decoupled tasks that extract spans of components, classify these components and the relations between them, which this thesis aims to extend by adding document and full summary context to the learning process.

Human-level control through deep reinforcement learning
A team of Google Mind researchers developed a novel reinforcement learning agent, the deep Q-network

(DQN) [21], that uses a deep neural network to approximate the optimal action-value (Q) function. They also introduce two methods that combat the instabilities when using a neural network as the Q function. The Q-function's weights are updated for learning in mini-batches of experiences. Where an experience is the set of state, action, reward, and the resulting state after taking that action, at a single time step. Using Atari games as a test bed for the algorithm, they outperformed the best existing RL methods on almost all the games without the agent using any game-specific domain knowledge. Additionally, DQN achieved more than 75% of human level play in over half of the games. This is useful because the use of a DQN is a possible method to incorporate RL into RC.

Automatic Text Summarization Using Reinforcement Learning with Embedding Features
Lee and Lee [15] use a DQN-based architecture [21] for the task of Automatic Text Summarization (ATS). ATS systems are used to generate short summaries from documents while still retaining their overall meaning. Lee et al approach the problem from an extractive perspective, selecting existing sentences from the document directly. This thesis covers extractive approaches to AM, that extract whole spans of text (argument components) from the original document. This is opposed to abstractive, where a summary of the document would be made from a newly generated set of words and sentences. The state is represented as a set of 3 memories: i) Contains information on all sentences in the doc ii) contains information about sentences in the currently generated partial summary, iii) contains information about sentences that have not currently been selected for the summary. The actions within the system are the placement of new sentences into the summary. The sentences are a single unit of summarization and are represented by feature vectors containing information about their meaning (via GloVe encodings [28]) and their position within the document. The Q-learning model is a deep neural network, mentioned in the prior section, that takes the current state and chosen action as input and outputs the expected value to be gained from generating a new summary after inserting the sentence into the partial summary.

While Lee develops a system for ATS, similarities can be drawn to the task of RC. In terms of AM, an argumentative summary could be built, made up of argument components and their relations, which may also benefit from the use of RL. RL is especially useful for ATS because the existing summary in full must be taken into account when deciding on the addition of more information. In contrast, the addition of an argument component may not be as dependent on an existing component that isn't directly linked. However, there could be value in taking into account the whole argument summary when predicting links and therefore incorporate this into RC via RL.

Improving Factual Consistency Between a Response and Persona Facts
Mesgar et al [20] present a response generating system that captures aspects of persona via a set of factual statements. It aims to produce appropriate responses that contain information from prior exchanges and facts from the speakers persona. Applications include digital assistants and automatic interactions in social media. They build upon existing methods that make use human-generated responses as labels for use in a supervised learning setting. An issue with learning via supervised neural networks is that the model's objective is to maximise the likely of human-written responses rather than generating responses that are viewed as semantically plausible and factually correct by humans. They use RL to maximise a score given by a reward function that assess the quality of the generated response according to three factors: i) factual consistency with persona facts, ii) topical coherence with dialogue history, and iii) language fluency. They use this reward function within their actor-critic RL method for Transformer-based model fine-tuning and evaluate their results using both human and automatic methods that show an increase in frequency of factually consistent responses compared to the previous purely supervised learning based method. This work proves that Transformer-based model fine-tuning is possible and can lead to substantial improvements on supervised learning alone, albeit in a different setting to AM. The methodology from this paper can be used to help aid the RL architecture design for AM.

Chapter 3

Approach

This chapter will cover the iterative approach that was taken towards improving models for AM. The first chapter will begin with the most basic architecture, BERT for sequence classification, that takes as input two sequences and outputs the relation classification between them. This leads on to several additions to the input feature set, namely distance and argument component type encodings. The subsequent section instead focuses on an architecture that is referred to throughout as RBERT. This model takes into account wider document context, taking the whole document (up to BERT's limit of 512 tokens) as input, using tags to highlight the pair of argument components. The penultimate section contains an exploration of joint learning in this setting that leads on to the final section, a discussion of the previous state of the art's use of joint learning in addition to residual networks and it's effect when implemented in the architecture.

3.1 BERTForSequenceClassification

The first step in the approach was to establish a baseline method on which to build upon and compare the subsequent methods back to. The natural choice for this was BERTForSequenceClassification (BERT-FSC), an existing method in the HuggingFace Transformers library. Mayer *et al.* had already implemented this within their code base on the AbstRCT dataset and this was used by building on top of their work, saving time that would have been spent designing and implementing surrounding infrastructure (data loading, training framework etc). However, some alterations and additions were necessary to add compatibility for the CDCP dataset. In addition to BERT-FSC, the effects of including an encoding of the distance between components within the document (covered in 3.1.2) as well as including encodings for the source and target component types (3.1.3), were explored.

3.1.1 BERT Encoding

BERT-FSC is a neural network architecture made up of a BERT transformer, drop out regularization, a fully connected (FC) layer and Softmax. The BERT tokenizer is used to map each token to an input ID and feed this set of input IDs into the BERT transformer. Segment IDs are also inputted to BERT to tell it which tokens are part of sequence one/two. Finally, an attention mask is included in order to tell BERT which tokens are part of the sequences and which are padding, allowing the transformer to only apply self attention to sequences. The addition of a drop out layer helps to combat overfitting and is kept at a constant rate of 10% throughout all of the tests. The FC layer takes the values from the final hidden layer within the BERT transformer and maps them to 3 neurons, one for each of the classes that the model predicts (NONE, REASON, EVIDENCE). These values are then fed into a Softmax layer from which the loss is calculated and subsequently backpropagate.

One of the difficulties associated with the CDCP dataset is the large class imbalance - 97% of relations are of type NONE, ˜3% are REASON, leaving EVIDENCE to make up less than 0.1% of the examples. The models often get stuck in a local optima, exclusively predicting the majority class, resulting in poor learning. The models would occasionally manage to find their way out of exclusively predicting NONE which led to seemingly illogical and sporadic results. To combat this problem, the weights are re-scaled (3.1) for each class in the cross entropy loss function (3.2). This reduces the proportion of the loss gained from majority class predictions and increases the proportional loss for predicting EVIDENCE/REASON examples. In other words, this makes the model more interested in how it performs at predicting the

	F1			REASON	
Approach	Macro	NONE	REASON	Precision	Recall
BERT-FSC	0.4092	0.9742	0.2535	0.22	0.29

Table 3.1: Best dev set results from BERT-FSC.

two smaller classes. This weight scaling lead to improvements with every subsequent approach that was tested and is therefore used in all of the proposed architecture's relation loss functions.

$$Scaling_i = \frac{\text{no. examples of } class_{min}}{\text{no. examples of } class_i} \quad \text{where } i = \text{NONE}, \text{REASON}, \text{EVIDENCE} \tag{3.1}$$

$$Scaling_{\text{NONE}} = \frac{35}{30435} = 0.00115 \qquad Scaling_{\text{REASON}} = \frac{35}{924} = 0.03788 \qquad Scaling_{\text{EVIDENCE}} = \frac{35}{35} = 1$$

$$loss(x, class) = weight[class]\left(-x[class] + log\left(\sum_j \exp x[j]\right)\right) \tag{3.2}$$

Results

This simple architecture works well for a range of GLUE tasks and was therefore anticipated to be well suited to the task of RC. The best result 3.1 (on the dev set) was recorded when training with maximum sequence length of 166 (the maximum token length of all pairs of components), for 30 epochs, with a learning rate of 2e-6. The EVIDENCE class scores are omitted from the tables of results due to an F1 score of 0.

3.1.2 Distance Encoding

During their preliminary analysis, Galassi *et al.* found that argument components located closer together in a document are more frequently linked than distant components, with 70% of links being between adjacent components. This led them to encode the distance between components as a 10bit array, where the first 5 bits are used if the source precedes the target and the latter 5 bits if the target precedes the source. For example, if the source component preceded the target component by 4 sentences, the resulting encoding would be 0111100000. They based this off previous work, such as AlphaGo [33], where scalars have been encoded in binary form. This method of distance encoding is also adopted in this work by concatenating the 10 digit feature vector to the output of BERT 3.1, before passing it through a neural network that uses a Softmax activation function to produce a classification.

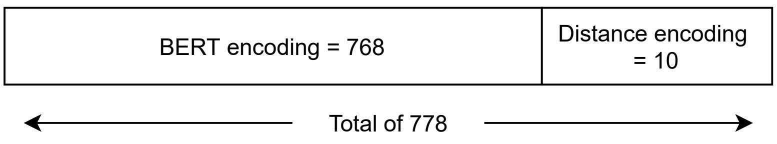


Figure 3.1: Feature vector with BERT and distance encodings

When BERT-FSC was adapted to include the addition distance information the results were actually worse 3.2. One reason for this could be that FC layer after encoding concatenation did not contain enough parameters or layers, to effectively learn from the extra contextual information. Several different neural network architectures were tested that sit between BERT and the Softmax layer 3.2 to try and improve on this. The 10 digit distance encoding has been omitted from the diagrams, however the two encodings would be concatenated between the dropout and first FC layer in all of the architectures. The most basic, a single FC layer 3.2b, was used as a baseline. More layers were then added, giving the

	F1			REASON	
Approach	Macro	NONE	REASON	Precision	Recall
BERT					
FC	40.31	98.24	22.70	33	17
2FC+RelU	38.25	96.50	16.54	27	12
BERTDist					
FC	40.08	95.66	22.47	23	22
2FC	39.65	97.72	21.24	22	21
3FC	40.18	97.80	22.73	24	22
2FC-RelU	**41.66**	**97.75**	**27.23**	**26**	**28**
2FC-Tanh	39.84	98.00	21.52	26	18
ResAttArg *testset*	42.95	98.32	30.56	-	-

Table 3.2: BERT-FSC + distance dev set results. The previous state-of-the-art method results on the test set are also presented for comparison.

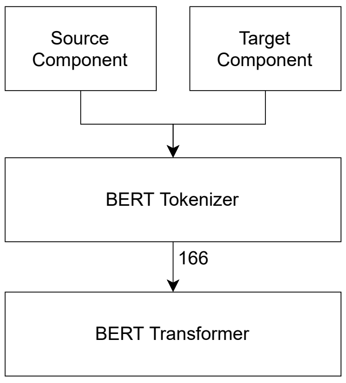
model more parameters to train. 2FC (3.2c) and 3FC (3.2d) add extra FC layers that gradually reduce the feature vector from 768(778) → 256 → 5 and 768(778) → 256 → 64 → 5 respectively. Non-linear activation functions, RelU 3.2f and Tanh 3.2e are also added to the 2FC model. This allows the model to learn more complex, non-linear relationships that may be present in the data. The four additional configurations are not exhaustive but are designed to give signs that the model either requires extra layers, some kind of non-linearity on top of Softmax, or both in order to learn the most from the data. Worse results would suggest the need for a more fundamental rethink of the architecture.

Results

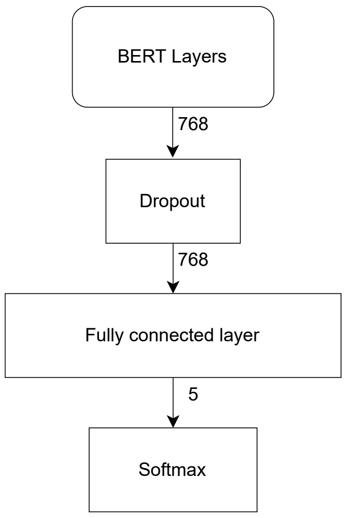
The results (3.2) show that any additional layers contribute to better NONE relation predictions. However, this often comes at a cost of recall of the REASON class, which subsequently causes a drop in Macro F1. One reason for this could be that the larger model overfits to the heavy class imbalance in the training set, hence the improvement in the NONE predictions. One counter measure to this could be to implement more regularization - however preliminary testing of higher dropout and the addition of weight decay did not result in any improvement. One exception to this is the use of the RelU activation function, where best result and an improvement on the baseline FC architecture of both the NONE and REASON classes is observed, leading to an improvement of 1.58% Macro F1 score. Once again, the EVIDENCE class has not been predicted at all due to it class being the smallest class in the dataset, the models still struggle learn an accurate representation of that type of relation.

As previously mentioned, when using distance encodings and the basic FC design the model shows no improvement on BERT encodings alone. After testing more complex model structures with the distance encoding, the 2FC-RelU architecture was then applied to just the BERT encodings alone (without the distance encoding), for comparison. However, it was found to perform considerably worse than all other architectures. This confirmed that the distance encodings are leading to improvements but that to be able to utilise the extra contextual information properly, the model needed extra layers. Additionally, it shows that extra layers and activations in conjunction with the BERT encodings alone adds unnecessary complexity to the model, resulting in a drop in performance. Under closer inspection of the metrics collected throughout training, the 2FC-RelU model using only BERT encodings learns the training set to the same capacity as its FC counterpart, but the extra model complexity causes it to overfit, losing it's ability generalize to the dev set.

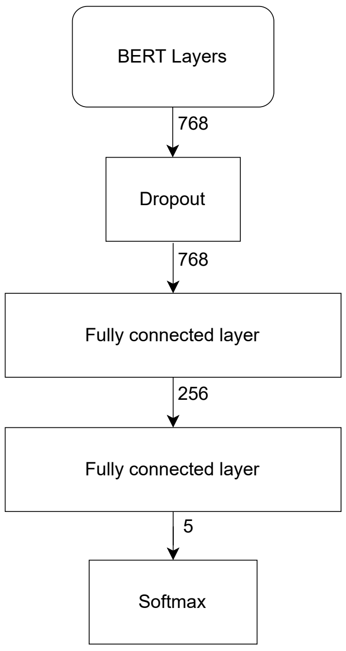
The ResAttArg method still outperforms the best results that were collected despite both methods utilizing the same sets of information (source, target, distance). There are still a number of reasons remaining as to why ResAttArg could be outperforming BERT-based methods. It could perhaps be that their encoding/embedding layers are more effective at this specific task, although this seems unlikely due to BERT's unmatched ability in many other NLP tasks. Other reasons could include ResAttArg's more complex residual network architecture that contains many more layers and parameters or it's implementation of multitask learning that leads to better performance.



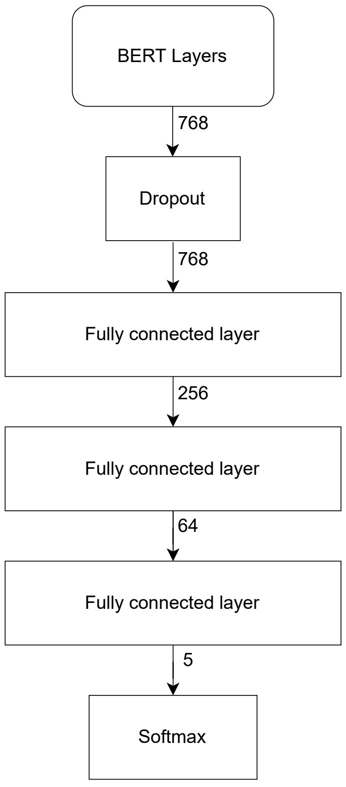
(a) BERT Layers



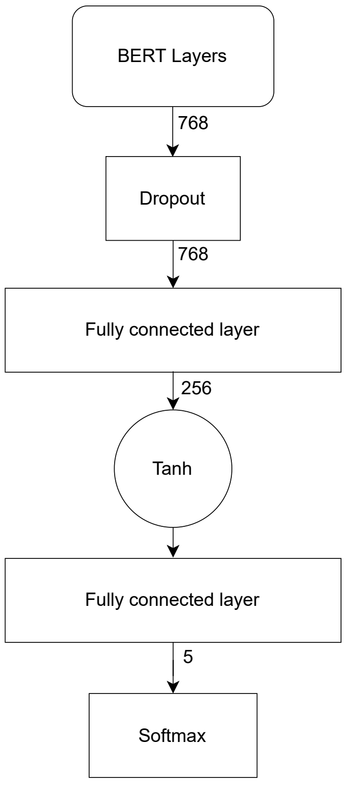
(b) FC



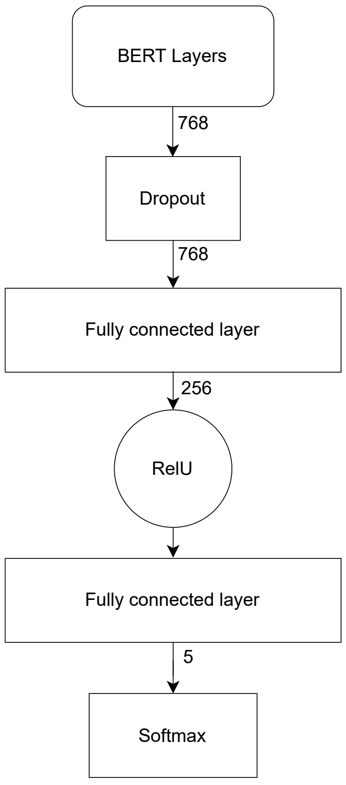
(c) 2FC



(d) 3FC



(e) 2FC-Tanh



(f) 2FC-RelU

Figure 3.2: Post-BERT architecture configurations

	F1			REASON	
Approach	Macro	NONE	REASON	Precision	Recall
BERT					
FC	40.31	98.24	22.70	33	17
2FC+RelU	38.25	96.50	16.54	27	12
BERTDist					
2FC-RelU	41.66	97.75	27.23	26	28
BERTDistComp					
FC	41.50	98.19	26.32	33	22
2FC	39.98	97.83	22.09	24	21
3FC	40.57	97.85	24.00	25	23
2FC-RelU	**41.78**	**97.78**	**27.51**	**27**	**28**
2FC-Tanh	41.02	98.06	25.00	29	22

Table 3.3: BERT-FSC + distance + components dev set results

3.1.3 Component Encoding

A further addition to the BERT encoding feature set is to add the source and target argument component type labels. It seems intuitively beneficial to have this information when trying to decide if there is a relation between two components. For example, specific pairs of component types could be more likely to be linked than others. One consideration with this approach is deciding which component labels to use. One option is to use the gold standard labels, although this would make the results directly incomparable with previous work as it would omit the component classification part of the AM task. Another option is to use the component span and type predictions from the BERTForSequenceTagging method th[illegible]used by Mayer in the first part of their pipeline approach. The two tasks can be run as a pipeline, pr[illegible]ving the option of comparison with similar methods, while possibly not being as accurate as using the gold standard. To compromise, the gold standard is used, with the option of testing the pipeline method if the results are promising. A third option that is explored later is to utilise the gold standard components labels as labels in a joint learning environment, rather than within the input feature vectors.

The common and well documented one-hot encoding method was chosen as a way of encoding the categorical component type data into the feature vector. Both the source and target component [illegible] assigned 4 bits (1000 for policy, 0100 value, 0010 fact, 0001 testimony and 0000 reference). The two 4 [illegible] arrays were then concatenated together to form 8 total additional bits and then concatenated on to the BERT and distance encodings 3.3.

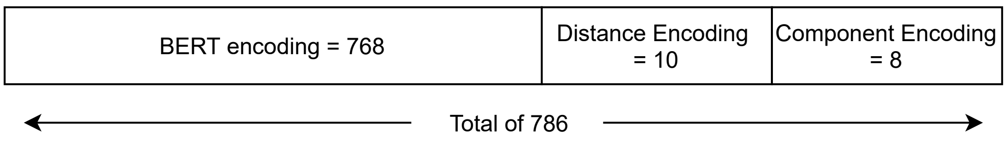


Figure 3.3: Feature vector with BERT, distance and component encodings

Results

The same architectures outlined in the previous section were tested over a range of hyperparameters, with the addition of the component encodings. The results show only a very slight improvement overall. This suggests that using predicted (as opposed to gold standard) component labels would likely perform worse than using just distance encodings and that joint learning may be a better use of component labels.

3.2 ResAttArg replication

In this section, replicating part of the ResAttArg architecture is explored in an effort to identify what part of its architecture is enabling its improved learning. The use of the residual network, a deep neural network made famous for its use within computer vision when it eliminated the vanishing gradient problem, is explored first. Then multi-task learning is introduced, a learning approach that has been stated in many papers as essential for the best performance in AM tasks. Before finally touching on Galassi's use of inverse relation labels for use during training.

3.2.1 Residual Network

While the original residual network (ResNet) (explained previously 2) was made with convolution layers for use with images, they can be replaced with dense layers as in this 2d vector implementation. The motivation Gallasi gives for using a ResNet is that they result in faster training, with the potential to add many layers. However, it is surprising to see that they exploit only a single dense layer and two residual layers before making the classifications. One of the main advantages of the ResNet is its ability to make very large networks that do not suffer from the vanishing gradient problem, which makes Galassi's choice of a small ResNet surprising. By inspecting their code base, it's clear that they tested larger ResNet sizes. This could be interpreted to suggest that use of a ResNet is not particularly suitable in this case.

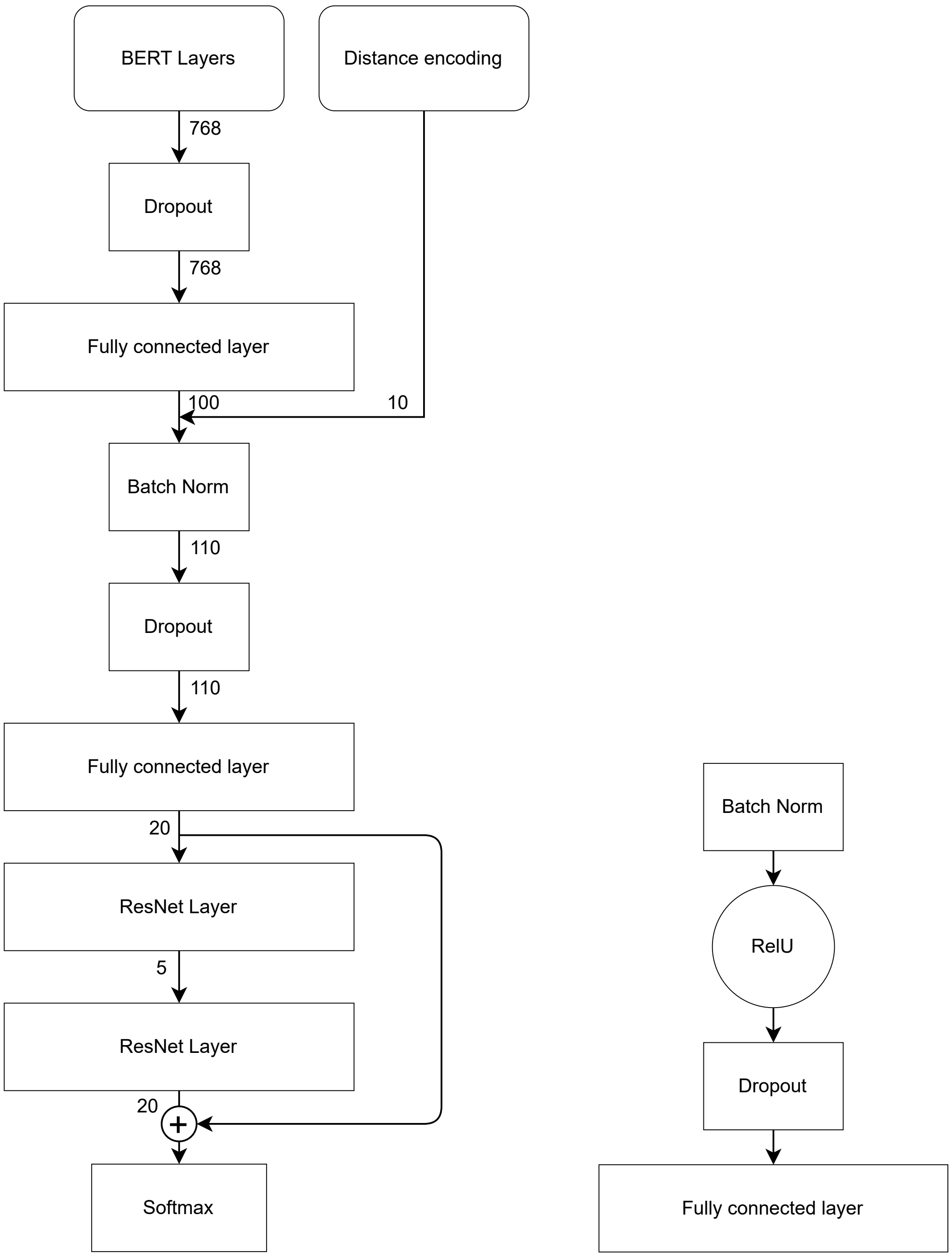
To replicate ResAttArg with a BERT based method, the embedding, encoding and LSTM layers are replaced with BERT but use the same latter part of the network made up of the residual network and classifiers. In order to use a network with the same dimensions, the output from BERT is reduced from 768 to 100 via a FC layer before concatenating the distance encoding.

It is expected that an improvement on the BERT+distance 2FC-RelU architecture will be observed due to the larger model size and the success that Gallasi using the architecture.

3.2.2 Joint Learning

The next iteration of envisioned improvements is the addition of joint learning. Multitask learning aims to leveraging addition information from similar tasks to improve the performance on a main task. Instead of taking a pipeline approach to a problem where each task is learned in sequence, joint learning combines the learning processes by sharing the internal representation of each task between all tasks. The method has lead to improved generalization, speed and precision of models.

In this project's joint learning set up, the model is trained on three prediction tasks: a) Relation between source and target components, b) Source component type, c) Target component type. Galassi also includes the binary link (on/off) task in their paper but on closer inspection of the code, they actually chose not to include it and therefore it is also omitted it from this work. The three tasks share every layer in the model until classification but each have their own fully connected layer that reduces the output down to the number of classes present for each task. Each also has it's own Softmax classifier. Cross entropy loss 3.2 is then calculated for each task. The relation classifier remains weighted 3.1, however the two component classifiers have equal weights for each class. The loss from each of the tasks is added together but because the relation task is the main task and wants more influence over the model's weights, it's loss is multiplied by a factor of 10. The classifiers in Figure 3.5 replace the single Softmax layer in Figure 3.4 to enable joint learning.



(a) ResNet architecture (b) ResNet Layer

Figure 3.4: Residual network replication

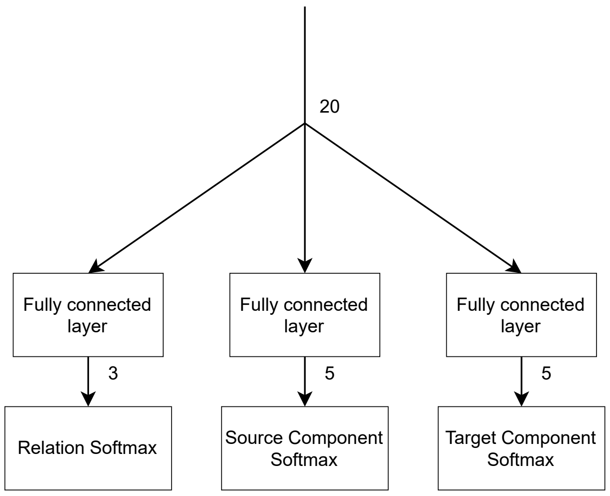


Figure 3.5: Joint learning classification

3.2.3 Inverse labels

Galassi also introduces inverse labels, meaning for each combination of components with a relation other than NONE, the reverse relation is labelled as it's inverse instead of NONE (e.g $C1 \rightarrow C2 =$ REASON, $C2 \rightarrow C1 = inverse$REASON). They state that the motivation for this is to mitigate the effects of the class imbalance between NONE and non-NONE labels by reducing the number of NONE labels. They also speculate that the additional labels may lead to better optimization. The additional labels are present during the training process but are replaced with NONE during evaluation.

Results

To understand which parts of the ResAttArg model were causing it to outperform the models outlined in the last section, 3 models are trained and evaluated with a range of learning rates, max sequence length of 166, and batch size of 20 for 15 epochs.

1. BERT + ResNet described in Section 3.2.1 and illistrated in Figure 3.4
2. BERT + ResNet with joint learning
3. BERT + ResNet with joint learning and inverse labels.

The best results on the dev set for each model are displayed in Table 3.4. They show that the base residual network approach performs worse on unseen data than all three of the architectures outlined in Section 3.1. The addition of both joint learning and inverse labels result in even worse performance across each of the metrics.

Analysis of the training metrics through Tensorboard shows unstable and slow learning of the training data; over the 15 epochs the models only achieve around 50 F1 score on the REASON class. This is well below BERT-FSC methods that entirely learn the training data and achieve ˜100 by ˜13 epochs. The recorded values can also be seen to have large ranges, oscillating between highs and lows. This gets more extreme as joint learning and inverse labels are incorporated. It is possible that after many more epochs the model will be able learn the training data but it is unlikely the results on unseen data would compete with previous methods. The slow and volatile learning contradicts claims that inverse labels lead to better optimization. Inconsistencies in implementation between the models and the true ResAttArg could be responsible for the drop in performance. However, it could also be possible that these methods do not compliment BERT, are just obsolete, or that suitable hyperparameters that stabilize the learning process had not been discovered. More time could be spent to try and improve these models, however they seem to be so much worse, and train so slowly that instead a different approach to the ResNet based methods are explored in the hope of designing a model that is much more successful at the task.

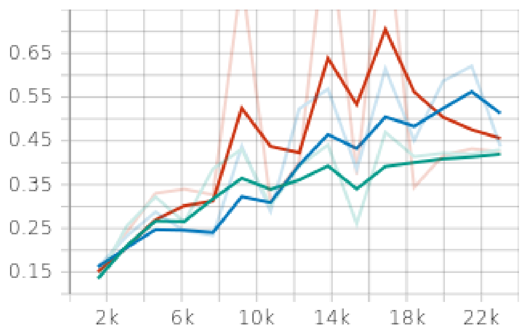


Figure 3.6: F1 score of REASON on the training set for ResNetJlInv, ResNetJl and ResNet.

	Relation F1			REASON		Component F1
Approach	Macro	NONE	REASON	Precision	Recall	Macro
BERTResNet	39.72	95.76	23.42	16	42	
BERTResNetJl	37.21	94.28	17.34	11	39	38.73
BERTResNetJlInv	35.10	93.35	11.97	8	29	16.80
ResAttArg (TEST SET)	42.95	98.32	30.56	-	-	78.71

Table 3.4: ResNet results

3.3 Adding document context

The methods tried thus far all rely on a pair of sequences in order to classify the relation between them. In this section exploration of the effects of using a whole document as input is carried out. Using this method, the model receives much more contextual information by additionally being aware of tokens around the pairs of argument components. In this section R-BERT [40] is implemented. It is a model that builds upon BERT specifically for the task of relation classification. This work modifies RBERT to jointly learn the source and target components, as well as the relation between them.

3.3.1 R-BERT

A key difference between BERT-FSC and R-BERT is the addition of tags ($E1$, #E2#) around the entities (argument components in this case) present in the sequence. Similarly to how BERT-FSC uses the hidden state of the [CLS] token as vector representation of the pair of sequences, R-BERT additionally uses averages of the two tokens' hidden states that surround each entity as vector representations of the entities within the sequence. Each of the vectors are then operated on by a Tanh activation function. The two entity vectors are passed through a common FC layer while the sequence vector is passed through its own FC layer. All three vectors are then concatenated and passed through a FC layer, before they are finally inputted to a Softmax layer that outputs class probabilities. Drop out layers proceed each FC layer in the network, which is keep constant at 10%.

The motivation to use R-BERT comes from its proven ability on a similar task. Wu and He design R-BERT from a relation extraction viewpoint and evaluate the model on the SemEval-2010 Task 8 dataset (which will now be referred to as just SemEval). A clear difference between SemEval and CDCP is the change in magnitude of the sequence and entities. SemEval's sequences are sentences and the entities are words within them. This is in contrast to CDCP, made up of documents that span multiple sentences and often contain many argument components. Wu *et al.* chose to use a maximum sequence length of 128, providing evidence that the SemEval sequences are less than 128 tokens in length. For use with the CDCP dataset, maximum token length is set to BERT's maximum possible sequence length of 512 because the largest document in CDCP is 552 tokens long. Due to time constraints the simple solution of discarding documents that are too large for BERT is employed, resulting in a 10% reduction in the size of the dataset. This could reduce the performance of the model due to having less training examples as well as cause a loss of direct comparability with other models such as ResAttArg by having a non-identical test set. With more time it would have been useful to try out other solutions such as implementing a

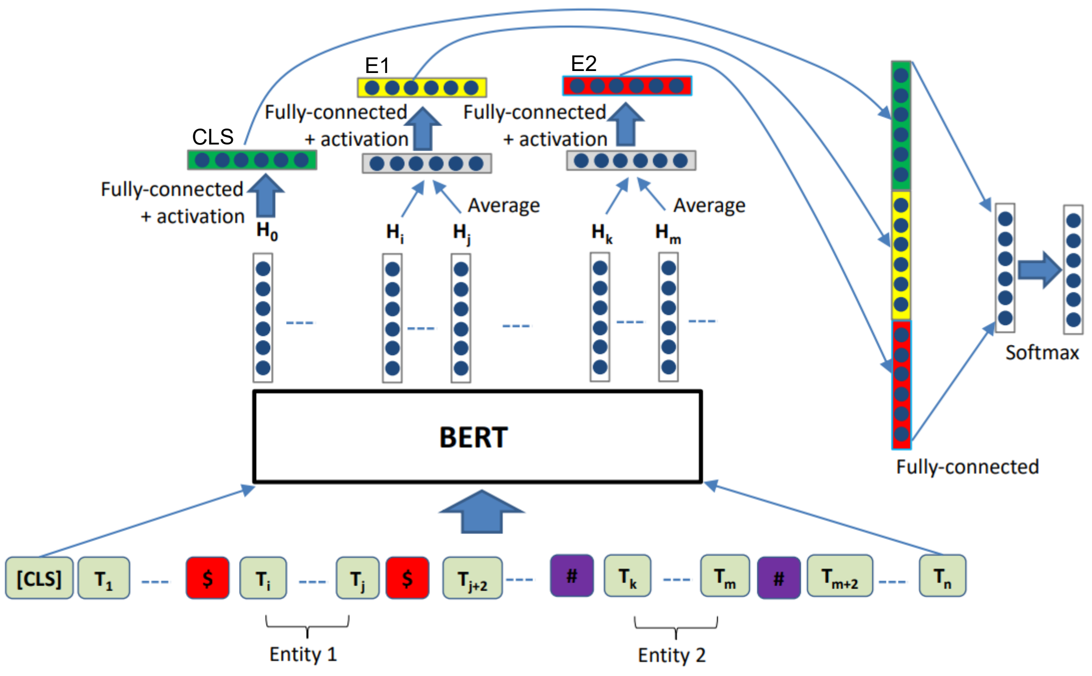


Figure 3.7: RBERT architecture diagram modified from the original RBERT paper [40]

	Relation F1			REASON		Component F1
Approach	Macro	NONE	REASON	Precision	Recall	Macro
RBERT	45.54	98.48	38.16	48	32	-
ResAttArg (TEST SET)	42.95	98.32	30.56	-	-	78.71

Table 3.5: RBERT dev results

new or existing method to reduce the size of larger documents while still maintaining their semantics and structure. Another option would have been to use a language model instead of BERT that allows for longer sequences, such as the Longformer [1]. The difference in scale between sequences and entities, as well as the distance between entities, are key differences to the problem explored in the original paper [40] and can be thought of as an extension of their research.

Another inherent difference is the undirected relationships present in the SemEval dataset and the directed REASON/EVIDENCE relations in CDCP and other AM datasets. To account for this, the original implementation that assigns the first and second entity \$ and #, respectively, is changed to instead assign the \$ tags to the source component and # tags to the target component.

The loss function class weights were also modified the same way as 3.1 to help deal with the large class imbalance.

R-BERT was expected to perform well due to it's proven ability at a similar task and extra contextual information it has access to. The main risks that were expected with this architecture was that the model could overfit due to the larger input sequence, or that the existing post-BERT network structure could be too small to learn the more complex representation of larger, more distant entities and directed relations.

Results

A number of RBERT models were initally trained and evaluated with varying learning rates because it is the hyperparameter that often has the most impact on performance [2]. The number of epochs were then varied. The best model was found when training RBERT for 10 epochs at 2e-5 learning rate. The results show a large improvement on any previous method, outperforming BERTDistComp 3.1.2 by 10.65 for the REASON F1 score. The results support the notion that BERT can make use of the extra context, even when the document can range up to its maximum input length. The precision score is considerably higher than BERTDistComp, highlighting how the added information particularly reduces false positive errors.

While the dev set contains different examples and is smaller in size than the test set, it still gives an indication of the results on unseen data. The test set will be reserved until final evaluation of the best performing model on the dev set in order to get the closest simulation of 'real-world' unseen data. Despite this difference in dataset used for evaluation, the results in Table 3.5 show that RBERT betters the state-of-the-art ResAttArg model across both the NONE and REASON class although it still does not make a single EVIDENCE prediction.

In order to compare the complete AM pipeline of relation and component classification using BERT, the BERTForSequenceTagging method developed by Mayer *et al.* [19] can be used for the component classification part of the pipeline. This model achieved a component macro F1 of 85.94 on the dev set, which is also comparable to ResAttArg.

3.3.2 Joint learning

While joint learning had a negative effect on the ResNet architecture, the method still seemed likely to prove worthwhile in another setting due to much of the literature boasting its success and in some cases necessity.

Two joint learning configurations of RBERT were tested. The first (V1) is as shown in Section 3.2.2. All three classifiers share the concatenation of the [CLS] and entity vectors just as in the original RBERT model.

In the second configuration (V2), the E1 (source) and E2 (target) vectors from the entity FC layer are used for the source and target component classifiers, respectively. It seemed intuitive to use the entity vector representations alone for the entity classifications, as their representations are more specific to the entities themselves, but to combine all three vectors as in the original architecture for the relation classification.

	Relation F1			REASON		Component F1
Approach	Macro	NONE	REASON	Precision	Recall	Macro
RBERT	45.54	98.48	38.16	48	32	-
RBERT-JL						
V1	46.33	98.49	40.51	48	35	65.35
V2	**47.24**	**98.52**	**43.21**	**50**	**38**	**67.19**
ResAttArg *test set*	42.95	98.32	30.56	-	-	78.71

Table 3.6: RBERT-JL dev results

Results

The results in Table 3.6 show that jointly learning both the argument components and the relations between them benefits the learning process. V1 sees major improvements in REASON precision score due to the model correctly classifying more examples, hence the rise in recall but while still producing the same ratio of type 1 errors as without joint learning. This suggests that learning to classify the argument components helps the model to make more confident positive predictions and in turn predict the negative NONE class less frequently.

The second approach to joint learning improves the performance metrics for REASON even further, where both a jump in precision, showing that the model produces less type 2 errors, and recall can be observed. There is also a slight increase in component F1 score which shows that classifying the argument components from vectors that are more specific to their original token representation helps to more accurately learn the relations and components within the document.

It would have been useful, given more time, to try even more adaptations to RBERT-JL. One further change could be to add more FC layers for the concatenated vector to pass through, allowing the model to learn a more complex and accurate representation of the underlying representation. Another is to experiment with using other hidden state vectors from the output of BERT, as RBERT provides evidence that using more hidden states from BERT can lead to better classifications. It would also be interesting to explore changing the loss function class weights for the argument components like what was implemented for the relation classes. One final thing to investigate is the loss function weightings. All tests have been carried out with a scaling of 10:1:1 for relation, source, target, respectively. However, this is unlikely to be optimal and it is likely that increasing the ratio of loss gained from the source and target component predictions would improve their classification performance, although perhaps at the cost of relation performance.

Chapter 4

Evaluation

The Approach 3 chapter covered the iterative development of the BERT based AM model and concluded by analysing the performance metrics of the best performing model, RBERT-JL(V2). This chapter documents quantitative analysis of RBERT-JL, in which comparison of the example attributes from subsets of the test set is carried out. These subsets are created by grouping examples that were correctly/incorrectly predicted by RBERT-JL and BERTDist, which is used as a baseline due to its similarities with the state-of-the-art. This quantitative analysis helps to guide qualitative analysis of specific examples within the dataset.

4.1 Test set results

Evaluation of the models using the dev set has been carried out throughout their development and documented within Chapter 3 alongside the model's implementation. Due to this process of continuous evaluation using the dev set, there is a risk of adding bias to the design choices by potentially tailoring the design to the limited dev set that is have available. For this reason, an unused test set has been reserved until this point in the project. This set acts as completely unseen data. By using the test set to do the final evaluation, it allows closer simulation of the models performance in the 'real world', as if new examples had been collected.

The test set is identical to many previous models that are developed and evaluated using the CDCP dataset. This is useful because it allows direct comparison with previous methods. The slight exception in the case of BERT is its limitation of 512 input tokens and, as discussed in section 3.3.1, it was decided that the oversized examples would be omitted, thus reducing the test set by 1020 NONE, 8 REASON and 0 EVIDENCE examples.

The results in Table 4.1 show that RBERT-JL outperforms the state-of-the-art across all metrics. This suggests that the extra context that RBERT uses over ResAttArg is useful for classifying both the relations and possibly the components. RBERT-JL's test set scores are slightly below the dev set scores in all by component F1. Not only does the test set have its unseen properties but it's also larger than the dev set which reduces sample variance and subsequently increases representation accuracy of the population, providing more reliable results.

One objective of the thesis was to identify why ResAttArg performs so well but when roughly recreated with a similar but BERT based architecture it was never able to reproduce results as strong as the original architecture . One possibility is that ResAttArg's encoding and embedding layers are more specialised for the AM task than BERT. They train their model using static GloVe embeddings for 100 epochs in comparison to BERT model fine-tuning and other network layer training for 10-20 epochs. It is only when longer input sequences are made use of in the RBERT implementations that a BERT-based architecture manages to out perform ResAttArg.

	Relation F1			REASON		Component F1
Approach	Macro	NONE	REASON	Precision	Recall	Macro
RBERT-JL (V2)	46.46	98.39	40.98	48	36	81.43
ResAttArg	42.95	98.32	30.56	-	-	78.71

Table 4.1: F1 scores on the test set

4.2 Example aggregation

This section focuses on making sense of the test results from RBERT at macro level. The overall motivation behind this is to make the model's results more interpretable as this can lead to a better understanding of how the model is making decisions. This should provide a good understanding of where future work could lead to improvements.

Given more time, it would have been preferential to use the test results of ResAttArg to compare against RBERT-JL, however, various problems were encountered when trying to get the model to run on the Bristol HPC machines. Time pressures ultimately prevented this from being completed. Instead that next best model to use for comparison is BERT-Dist (DIST) will be used. This decision is justified because of the fact it is the best performing model that takes the same input as ResAttArg (2 components and the distance encoding). Despite DIST's worse performance, perhaps the types of examples it struggled with are similar to ResAttArg. This is unfortunately only educated speculation but it still provides a basis for comparison with RBERT-JL.

The confusion matrices in Figure 4.1 presents the distribution of true and predicted labels. The NONE class is by far the dominant class and the difference in correct NONE predictions between the models is a very small proportion of the total number of examples. RBERT predicts twice as many REASON examples correctly than DIST. During the rest of this analysis the results of examples labelled with REASON relations are used because it most distinctly differentiates between the models. This is because the NONE relation is a negative case that makes up the vast majority of examples of which all models perform relatively similarly at. Additionally, the EVIDENCE class has so few examples and is subsequently very rarely predicted, and never correctly. Both models still struggle with the lack of EVIDENCE examples, although RBERT attempts to classify 4 examples as EVIDENCE, showing that it has built up a rudimentary representation of that class.

Sets of test examples are grouped by different conditions and then various statistics from these groups are calculated. Table 4.2 contains the mean and standard deviation of a few metrics that help to discriminate between groups of examples.

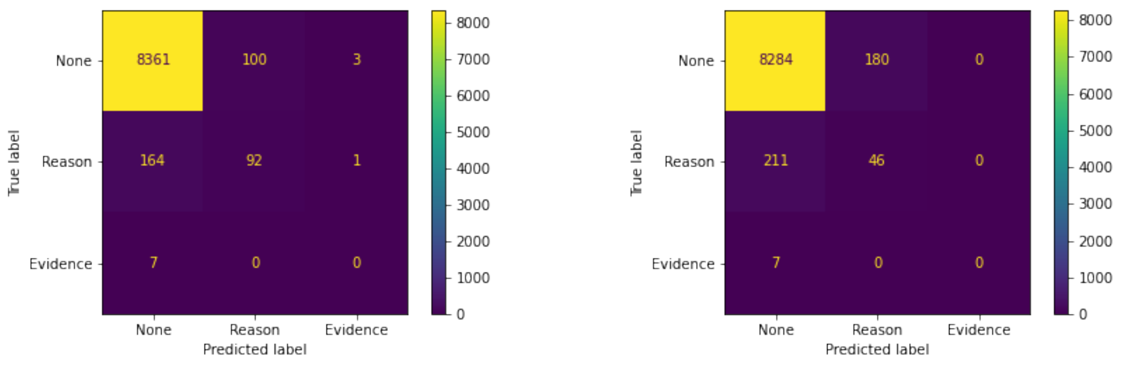
The following metrics to summarise groups of examples:

- Source is first: if the source component proceeds the target component
- Source/Target length: the number of tokens in the Source/Target component
- Document Length: the total number of tokens in a document
- Bridge Length: the number of tokens between the two argument components
- Non-component Length: the number of tokens that are not part of either of the argument components

		$\mathbb{U}$	R ∩ D	¬ R ∩ ¬ D	R ∩ ¬D	¬R ∩ D
Source is first	%	0.30	0.29	0.28	0.30	0.47
Source Length	Mean	22.82	22.39	23.66	21.64	20.20
Target Length	Mean	19.09	19.90	19.46	17.62	19.60
Document Length	Mean	191.05	142.32	201.31	178.74	239.33
Document Length Std	Std	112.31	106.39	115.50	100.28	108.57
Bridge Length	Mean	11.37	0.00	17.11	0.57	21.40
Bridge Length Std	Std	26.66	0.00	32.31	3.36	24.10
Non-comp Length	Mean	149.14	100.03	158.19	139.48	199.53
Non-comp Length Std	Std	114.75	106.10	117.95	104.02	113.33
No. of examples		257.00	31.00	150.00	61.00	15.00

Table 4.2: Quantitative example analysis of REASON relation examples

R and ¬R are the sets of examples that RBERT predicts correctly and incorrectly, respectively. D refers to DIST.
See notation



(a) RBERT (b) DIST

Figure 4.1: Confusion Matrix

4.2.1 Component Distance

The most striking observation between the subsets of examples in Table 4.2 is that almost all of the examples that RBERT classified correctly have a bridge length of 0. At first this is surprising, however, when closer inspection is given to the complete distribution of bridge lengths over the REASON examples 4.4 it is evident that the vast majority of examples have little to no tokens between components. This has already been eluded to in section 3.1.2, when Galassi's analysis showed that 70% of related components are adjacent to each other. Figure 4.5 supports Galassi's analysis and also shows that REASON relations are less frequent as distance increases. RBERT correctly predicts REASON relations between adjacent components with a much higher frequency than DIST. Whereas some aspects of DIST enable it to exclusively classify a few of the more distant REASON relations. This is evidence that RBERT does not find the context from tokens between components useful in predicting longer distance relations.

Assuming that the training set follows a similar distribution as the test set and therefore contains vastly more adjacent REASON examples than any other distance, it makes sense that RBERT makes the most improvements on that type of example. Although, an increase in accuracy across distant components in proportion with the number of that examples of that distance in the training set would also be expected. However, this is clearly not the case with RBERT as shown in Figure 4.2. The figure shows the change in the number of correct predictions between RBERT and DIST for examples split up by distance and in proportion to the total number of examples of that particular distance. RBERT correctly predicts an additional 25% of total adjacent relation examples on top of DIST but performs much worse at relations of all other distances. One explanation for this is that because the vast majority of REASON examples in the training set contain adjacent components, the model overfits to this type of example. It seems that the due to the imbalanced data, whenever there are tokens between the components, the model views the example as very unlikely to contain a REASON relation. It's likely that the attention mechanism of BERT strongly alters the representation of the components (and the [CLS] token) to reflect this. In contrast, DIST's encoded distance information is added to BERT's output vector representation and therefore is never processed by BERT, removing the possibility of BERT to overfit in the same fashion and subsequently allows it to still make some longer distance REASON relation predictions.

Below is an example of where both RBERT and DIST correctly predict the adjacent REASON relation (relation A) but only DIST is able to spot the the two distant REASON relations (B & C).

> 'My company calls, auto-dials, emails, and snail mails customers many times before they are sent to a collection agency. Many debtors simply refuse to respond to the original debtor. They only make payment after the account is assigned to a collection agency – akin to seeing just how far they can go before thereś a consequence to not making payment. [TRG] **Requiring a validation notice from the owner of the debt should be optional on the creditorś part.**[/TRG] [SRC1] *The FDCPA requires the collection agency to send their first notice after the debt is assigned.* [/SRC1] [SRC2] *More notices from the creditor are useless and expensive.* [/SRC2] And the single most frequently used excuse of any debtor is "I didnt́ receive the letter/invoice/statement". [SRC3] *A first class letter costs $0.55 minimum (inclusive of postage, paper, and envelope). That cost does*

not include overhead and employee cost. Multiply that cost by the number of notices and statements already sent (and ignored) and you begin to see the true cost of collection. [/SRC3]'

- REASON relation A:
  TRG: Requiring a validation notice from the owner of the debt should be optional on the creditorś part.

  ↑ REASON ↑

  SRC1: The FDCPA requires the collection agency to send their first notice after the debt is assigned.
- REASON relation B $TRG \longleftarrow SRC2$:
  TRG: Requiring a validation notice from the owner of the debt should be optional on the creditorś part.

  ↑ REASON ↑

  SRC2: More notices from the creditor are useless and expensive.
- REASON relation C $TRG \longleftarrow SRC3$:
  TRG: Requiring a validation notice from the owner of the debt should be optional on the creditorś part.

  ↑ REASON ↑

  SRC3: A first class letter costs \$0.55 minimum (inclusive of postage, paper, and envelope). That cost does not include overhead and employee cost. Multiply that cost by the number of notices and statements already sent (and ignored) and you begin to see the true cost of collection.

4.2.2 Non-component Length of examples with adjacent relations

Now that it has been established that the entirety of RBERT's gains are on examples with adjacent components, further investigation what particular types of these examples RBERT classifies with better accuracy than DIST can be carried out. It seems sensible to assume that the model would find contextual information on either side of the adjacent components useful. Figure 4.3 is useful for this analysis, showing the change in the number of correct predictions between RBERT and DIST for examples binned by non-component token length and in proportion to the total number of examples in each bin. However, all examples with distance greater than 1 (not adjacent) have been omitted. RBERT makes similar improvements across the whole distribution with two exceptions. The first is that the improvements are lower on the final two bars of the graph which could signal that context past that point is not only too much for RBERT to find useful, but is actually detrimental. The other exception is the bin around 200 tokens. There does not seem to be any intuitive reason for this especially as there are plenty of training examples in this bin, as shown in Figure ??.

Below is a small, simple document that is a common archetype of document present within CDCP. RBERT often gets these types of examples correct, as it does in this particular case.

> '[TRG] This is not an appropriate requirement [/TRG] [SRC] because not all states have licensing or registration for debt collectors. [/SRC]'

On the other end of the spectrum, here is a long document less typical in CDCP but that BERT also predicts the REASON relation correctly for.

> 'I appreciate Mr. Bartmannś comments but they appear to be grounded in a collection agency perspective (there is always business and market share tension between collection agencies and law firms). The reality is that (and this was voiced by other participants) many (not all but many) debtors simply wont́/cant́ pay until they have to. The FDCPA and TCPA have truly made communications with debtor both difficult and hazardous for the collector. For instance, the fact that you cant́ safely leave a voice message for a debtor should be very troubling. Over the years, our firm has stopped initial outbound calling (responding only to inbound calls) and also limited our pre-suit letters to just two. We send the required disclosure letter and (there being no response and the account reviewed for suit) then send a discounted payment offer that says we are going to file suit but would rather settle. We offer reasonable terms to the debtors to avoid a costly suit but the response rate is absolutely miserable, probably in the single digits percentage wise. Believe me, suits are time consuming and expensive, but they are often simply the only option. Even when the debtors are served, few contact us, and even fewer when the court sends them the judgment. Worse, a judgment doesnt́ guarantee payment on the account. Clarification via Regulations as to what we can say and do in communications would at least promote more communication between the parties.

> With that, perhaps more points of contact could resolve debts before suit. [TRG] I strongly disagree that litigation should be ”discouraged” [/TRG] [SRC] as it is not only a simple reality in much of debt collection but a fundamental right of a creditor.[/SRC] However, I think Regulations making it clear what can and cannot be done in communications that can shield the industry from the wild west of FDCPA and TCPA lawsuits would greatly encourage more communication. I also think a reasonable, good faith attempt to resolve the case (offering a payment plan and/or lump sum) before suit is filed is still a good idea as well even though our own response rate has not been great. Anything more will infringe on a creditorś rights and interfere with state law. Perhaps the CFPB Portal could have a debt resolution aspect to it where a debtor could try to resolve an account in a ”safer” more disarming manner.’

Figure 4.6 shows that RBERT out performs DIST on smaller documents at a higher frequency than longer documents. However, there are some improvements across the whole distribution, up to a length of around 400 tokens. This also seems to support the hypothesis that RBERT's longer input sequence is not the cause of its better performance. However, one reason could be that RBERT can utilise amounts of context up to a certain length, but as the document gets longer, it isn't able to exploit the extra information.

4.2.3 Still lacking context

While BERT's pre-training process, the large size of CDCP, and the use of document size input by RBERT all contribute to building BERT's broader contextual knowledge representation, some examples contain information too rare or abstract for it to understand the text and make correct decisions. An example where both RBERT and DIST are unable to spot the REASON relation is provided below. The ambiguity in the example makes it unclear for even a human to determine if the target is referring to another online user called 'Massachusetts' or the place's laws around robo-calls. It is likely that the model doesn't make the abstract link between whatever the user is referring to by 'Massachusetts' and even if it did it is extremely unlikely to know the context behind it. In some cases even though RBERT has the context from the whole user comment, this is not be enough to make sense of some of the relations between components that either relate to information that is not represented within BERT's parameters. The example also does not contain any obvious syntactic indicators of a REASON relation, such as words like 'because', separating the two components which could have made up for the lack of semantic clarity.

> <trg> I'm with Massachusetts on this one. </trg> <src> Repetitive and robo-calls are annoying and not productive. </src> Another fact about robo-calls is that their messages often start in the middle, or maybe this is done on purpose. When it has happened to me, I just hang up. Policies regulating the number of contacts made within a specific time period should include all modes of technology.

4.2.4 Summary

In summary, the analysis shows RBERT is able to learn from the majority of the dataset better than DIST but lacks the ability to extend this knowledge to some more of the fringe examples with longer distance between components. This is surprising as it seems intuitive that the information between components would be a good indicator as to the relation between them. However, RBERT is not able to exploit this information, which asks the question, why is it making better classifications? While the net number of correctly predicted examples with more tokens on the outside of adjacent examples goes up (including more context), this is because there are more training examples of this type, and when we adjust to look at the increase in proportion to the amount of training data there seems little difference between large and small amount of extra tokens.

This points to the possibility that the [CLS] token hidden state is not providing much benefit other than perhaps representing the distance between components. Instead, the more information rich mechanism that is employed to capture the component information in RBERT - using the average of two sets of BERT hidden states that surround each entity - could be responsible for the jump in performance. This is in contrast to DIST's architecture that solely relies on the [CLS] tag hidden state as a joint representation of the two components.

Future work could explore the performance of modified version of DIST that utilises additional hidden states for the argument components, as in RBERT, whilst keeping the distance encoding the same. This would provide more insight into how RBERT manages to outperform DIST so drastically. A significant difference between RBERT and DIST is that distance information is incorporated into DIST after the BERT encoding has been produced, preventing BERT from overfitting to adjacent components. One of the question's that the original RBERT paper [40] puts forward for future work is, how well does the model perform when used with data of much longer length with larger distances between entities. CDCP is a great example of this kind of data and current analysis suggests that it does not perform well. This seems to be because the model views the presence of any tokens between components as an indicator of there being no relation - which is true for the majority of the dataset.

Another consideration should be to gain better insight by training RBERT using a dataset with a balanced number of examples over the different distances between components. One option could be to artificially balance the CDCP dataset, by removing a collection of examples with adjacent components in order for the dataset to have a more uniform distribution of component distances. Although, consideration should be given to the fact that this will reduce the size of the dataset drastically, and therefore likely reduce the effectiveness of the model significantly, potentially even entirely. For this reason, using a different dataset could be more suitable. However, even if the model's overall performance drops, it may still show that the model is able to learn further distances, albeit with less overall accuracy, when the data is balanced.

An alternate line of enquiry for future work could omit the [CLS] token hidden state from the calculation of class probabilities. This would give a better understanding of how useful the whole document's representation is when predicting relations between components as current analysis seems to suggests it has little bearing on the results.

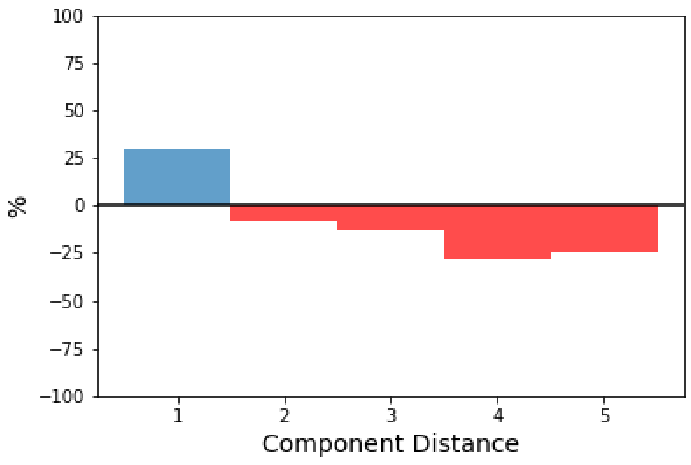


Figure 4.2: Component Distance proportional change between RBERT & DIST

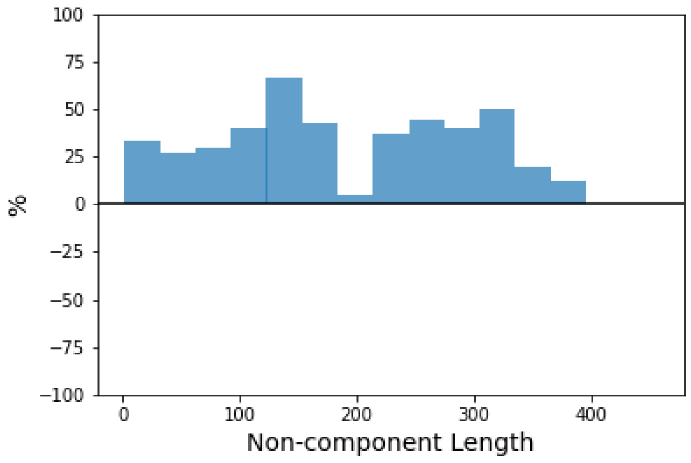
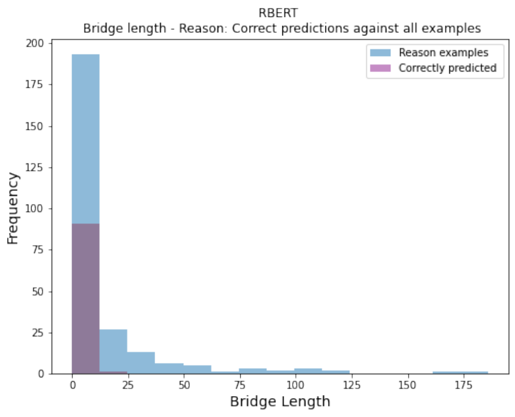
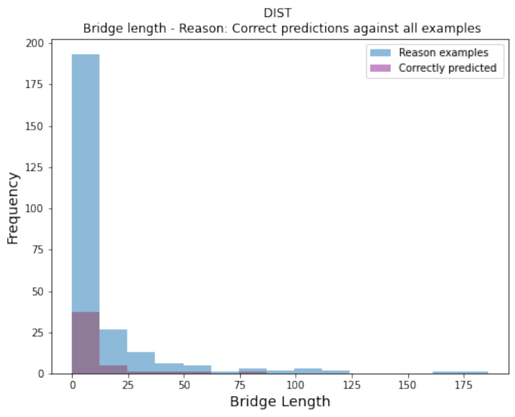


Figure 4.3: Non-component proportional change between RBERT & DIST of examples with adjacent components

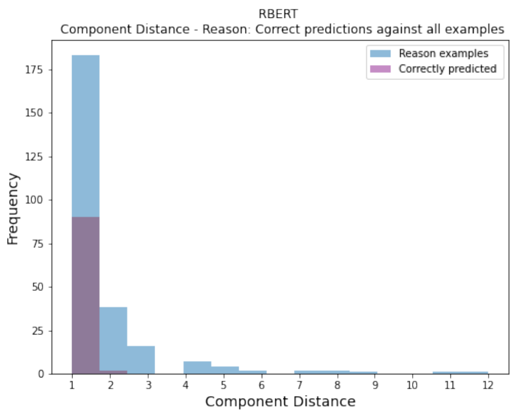


(a) RBERT Bridge Length

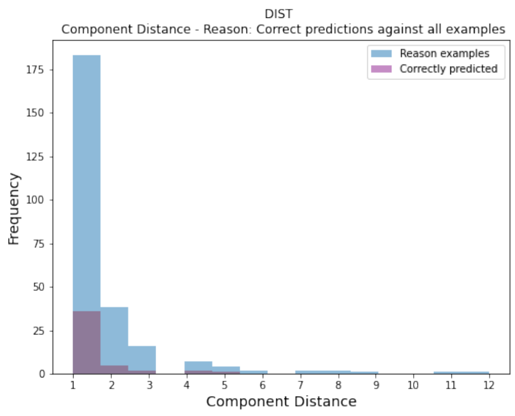


(b) DIST Bridge Length

Figure 4.4: Bridge Length

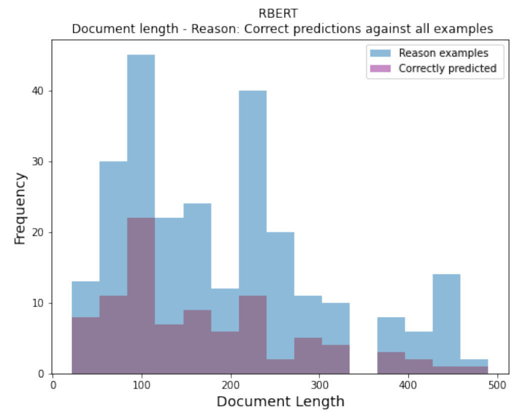


(a) RBERT Component Distance

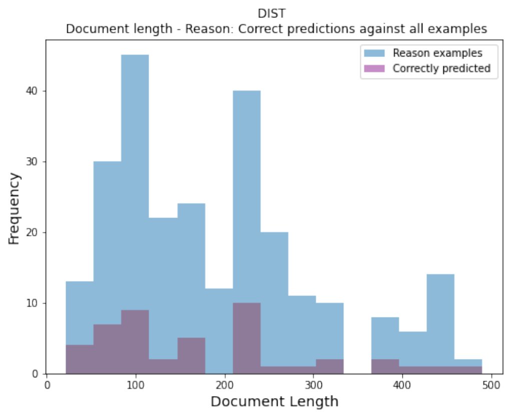


(b) DIST Component Distance

Figure 4.5: Component Distance

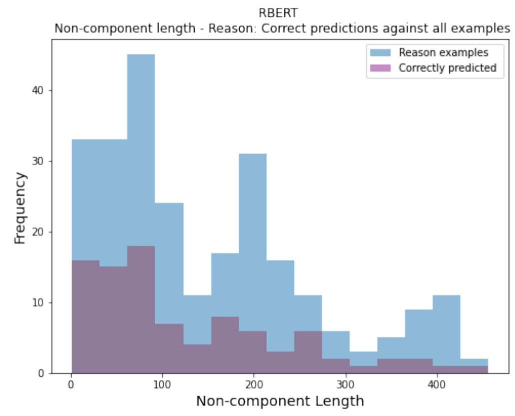


(a) RBERT Document Length

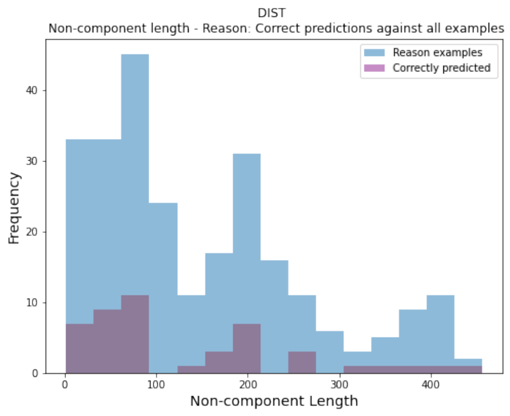


(b) DIST Document Length

Figure 4.6: Document Length



(a) RBERT Non-Component Length



(b) DIST Non-Component Length

Figure 4.7: Non-Component Distance

Chapter 5

Conclusion

5.1 Summary

This section will recap the previous sections and outline the project's main contributions.

Chapter 1 introduced the topic of AM and some of it's central motivations and applications. This is also where argument structure is explained, as well as the four subtasks of AM. A review of previous AM work is then provided and supported by a summary of the current gaps within the research and the main challenges that current work in the field is trying to overcome. The CDCP dataset used throughout the project is then described and accompanied by an example. Finally, the three main steps of the approach to developing a more advanced AM model are outlined, with particular focus on the main research hypothesis: that additional contextual information outside of the argument components would allow the model the model to make more informed argument relation classifications.

Chapter 2 summarizes supporting research papers that provide the reader with the background knowledge necessary to comprehend the rest of the thesis. In particular, this covers previous AM research as well as other relevant advances in machine learning and computational linguistics.

Chapter 3 provides the main bulk of the thesis, outlining the iterations of exploration that culminates in producing a new state-of-the-art Transformer-based model for AM that jointly learns both the argument component types and relations between them within a document of up to 512 tokens. The chapter begins with an overview of BERTForSequenceClassification, an existing and basic use of BERT embeddings paired with a classification head. The existing state-of-the-art ResAttArg model is then broken down in an attempt to find out what made it work so well. The chapter finished with an introduction to R-BERT's architecture and the implemented modifications that allow joint learning. All three of the aforementioned chapters contain evaluation of the performance metrics and provide a hypothesis for how and why each method works as well as it does.

Chapter 4 evaluated the new state-of-the-art RBERT-JL model. It featured quantitative performance analysis of RBERT-JL and BERT-DIST that aimed to provide evidence of how well different types of argument examples are processed. It showed that while RBERT does perform much better on average over the whole dataset than DIST, it loses the ability to generalise it's relation representation to distant relations. It concludes by hypothesising that due to the imbalanced nature of the CDCP dataset, the language model overfits to the adjacent relations and views any example with tokens between components as highly unlikely to be related, and changes the sequence embedding to reflect this. It is also hypothesised that the reason DIST does not overfit in the same way is because the distance information is only used by the classification head, rather than being part of BERT's input, and therefore does not allow BERT's attention mechanism to focus too heavily on the distance between components.

5.2 Project Status

The aims of the project are as follows:

- To first understand the current state of AM to determine potential gaps in the research.

- To extend the work of Mayer et al. [19] by improving the Transformer-based architecture for AM and in particular for the more challenging and widely used CDCP dataset.

- To experiment with different inputs to the language model as well explore the effects of different types of classification head.
- To better understand the existing state-of-the-art and what makes it work so well.
- To make more context available to the language model by using the whole document as input while highlighting the argument components via tags.
  - By using the existing RBERT architecture to do this, another aim that surfaced was to produce some insights into the future work suggested by Wu and He [40] around exploring how the model copes with longer distance relations.
- After much of the previous literature stated that joint learning was essential for competitive AM performance, one aim was to compare the effects of systems with and without joint learning.
- To reformat the problem to reinforcement learning setting as a way to optimise the joint combination of links for a document, as opposed to one-by-one.
- Ultimately, the aim was to produce a model that performed better than all previous models at AM (at least on the CDCP dataset) and to carry out in-depth analysis on it in order to pinpoint why it performs better.

From the points just mentioned, the three that were not completed will now be discussed.

Firstly, whilst the existing state-of-the-art was inspected and better understood, none of the methods to reproduce parts of it's architecture produced positive results. One possibility for this could be due to the incompatibility of BERT and parts of their architecture. However, the more probable reason is the fact that the replicas produced are not similar enough to ResAttArg to be directly comparable. The better approach would have been to start with their code base, and therefore exact architecture, and modify, isolate and test it from there. Although, this would not have been possible in the time available for this project and so the actual approach taken is still appropriate, albeit slightly ineffective.

Secondly, time did not allow for the reinforcement learning approach to extra model fine tuning. This addition would have been a truly novel and interesting approach to AM had it had been feasible.

Lastly, in Section 4, it would have been useful to evaluate RBERT-JL alongside the existing state-of-the-art, rather than DIST. Similarly to the first point about replication, complete use of their code base would have made this possible, however numerous issues were encountered trying to get it to run on the HPC and time constraints meant that it was not possible.

5.3 Further Work

Further investigation into existing models

As mentioned in Chapter 4, more insight can be gained into why RBERT performs better than DIST. An obvious line of enquiry could be to modify DIST so it utilises additional BERT hidden states similarly to RBERT. Currently, DIST uses the BERTForSequenceClassification method of using the [CLS] tag final hidden state to make classifications. However, while this may work well for sequence classification tasks, it does not seem to capture enough information from the sequence for RC (and component classification if jointly learned).

Another possible investigation could be focused on understanding how much the [CLS] token final hidden state helps to classify relations. This could be carried out by simply removing that part of the BERT embedding from use by the classification head and comparing the results back to the existing method.

The motivation behind these ideas are to make the models more explainable. This is useful not only to be able to explain to others why and how the model works but also to be able to better direct future development towards changes that will produce the best improvements.

Experimenting with other datasets

Similarly to the last section, another way of better understanding the current model would be to train it on an argumentative dataset with a more uniform distribution of relation distances. One problem with CDCP is the huge imbalance in this regard and it is suspected from the analysis that this contributes to RBERT overfitting to adjacent relations. Possible existing AM datasets to chose from include Doctor Inventor Argumentative Corpus [14] [6], AbstRCT [19] [18], Persuasive Essays Corpus [35].

Reinforcement Learning

As mentioned in Chapter 1, model fine tuning via RL was a project aim but did not manage to fit it into the time available for the project.

Codebase Improvements

It was an objective to keep the code modular and free of duplication by using object oriented principals, however this became less of a priority as time pressure increased nearing the end of the project. For this reason, much of the logic for similar model configurations is repeated and slightly bloats the repository. While future work should be primarily research focused, this could also be kept in mind and some of the files tidied up and refactored.

One concurrent focus alongside refactoring some of the code for tidiness could be to make the software more generalizable to extra datasets and related tasks. One focus of Galassi's work is the general nature of the algorithm and how it was designed with broader applications in mind. While this project has focused around CDCP and joint learning of AM tasks, it would not take much work to bring it more inline with ResAttArg's more general applicability. In particular, developing the support for other datasets into the system would be greatly beneficial.

A final improvement to the codebase would be to produce a more automated and efficient process for training and evaluating models. The ability to store predictions and performance metrics in a database would speed up future development significantly. The current system requires the developer to sift through text based log files or Tensorboard [1] event logs. The latter is already extremely useful, especially for viewing results as the models are training. However, one issue is that the number of training runs (and the numerous metrics that are collected) can be overwhelming and unorganised.

[1] https://www.tensorflow.org/tensorboard

Bibliography

[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[2] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[3] Robert Caba. Renewable energy persuasive essay. 2016.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding.

[5] Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. Neural end-to-end learning for computational argumentation mining, 2017.

[6] Beatriz Fisas, Francesco Ronzano, and Horacio Saggion. A multi-layered annotated corpus of scientific papers. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3081–3088, 2016.

[7] Andrea Galassi, Marco Lippi, and Paolo Torroni. Argumentative link prediction using residual networks and multi-objective learning. In *Proceedings of the 5th Workshop on Argument Mining*, pages 1–10. Association for Computational Linguistics.

[8] Andrea Galassi, Marco Lippi, and Paolo Torroni. Multi-task attentive residual networks for argument mining. version: 1.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition.

[10] Xinyu Hua and Lu Wang. Understanding and detecting supporting arguments of diverse types.

[11] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition.* Pearson Prentice Hall, Upper Saddle River, N.J., 2009.

[12] Alexandros Komninos and Suresh Manandhar. Dependency based embeddings for sentence classification tasks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1490–1500, San Diego, California, June 2016. Association for Computational Linguistics.

[13] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. 47:498–519.

[14] Anne Lauscher, Goran Glavaš, and Simone Paolo Ponzetto. An argument-annotated corpus of scientific publications. In *Proceedings of the 5th Workshop on Argument Mining*, pages 40–46, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[15] Gyoung Ho Lee and Kong Joo Lee. Automatic text summarization using reinforcement learning with embedding features. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 193–197. Asian Federation of Natural Language Processing.

[16] Joohong Lee, Sangwoo Seo, and Yong Suk Choi. Semantic relation classification via bidirectional lstm networks with entity-aware attention using latent entity typing. *Symmetry*, 11(6):785, 2019.

[17] Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. Context dependent claim detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1489–1500. Dublin City University and Association for Computational Linguistics.

[18] Tobias Mayer, Elena Cabrio, Marco Lippi, Paolo Torroni, and Serena Villata. Argument mining on clinical trials. In *COMMA*, pages 137–148, 2018.

[19] Tobias Mayer, Elena Cabrio, and Serena Villata. Transformer-based argument mining for healthcare applications. In *ECAI 2020 - 24th European Conference on Artificial Intelligence.*

[20] Mohsen Mesgar, Edwin Simpson, and Iryna Gurevych. Improving factual consistency between a response and persona facts. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 549–562, Online, April 2021. Association for Computational Linguistics.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. 518(7540):529–533.

[22] Raquel Mochales and Marie-Francine Moens. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22, 2011.

[23] Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. Automatic detection of arguments in legal texts. In *Proceedings of the 11th international conference on Artificial intelligence and law*, pages 225–230, 2007.

[24] Huy Nguyen and Diane Litman. Improving argument mining in student essays by learning and exploiting argument indicators versus essay topics. In *The Twenty-Ninth International Flairs Conference*, 2016.

[25] Vlad Niculae, Joonsuk Park, and Claire Cardie. Argument mining with structured SVMs and RNNs.

[26] 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.

[27] Joonsuk Park, Arzoo Katiyar, and Bishan Yang. Conditional random fields for identifying appropriate types of support for propositions in online user comments. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 39–44. Association for Computational Linguistics.

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

[29] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations.

[30] 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, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[31] David E Rumelhart, Geoffrey E Hintont, and Ronald J Williams. Learning representations by back-propagating errors. page 4.

[32] P. Saint-Dizier. Challenges of argument mining: Generating an argument synthesis based on the qualia structure. In *INLG*, 2016.

[33] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[34] Maria Skeppstedt, Andreas Peldszus, and Manfred Stede. More or less controlled elicitation of argumentative text: Enlarging a microtext corpus via crowdsourcing. In *Proceedings of the 5th Workshop on Argument Mining*, pages 155–163, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[35] Christian Stab and Iryna Gurevych. Parsing argumentation structures in persuasive essays.

[36] Christian Stab and Iryna Gurevych. Identifying argumentative discourse structures in persuasive essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56, Doha, Qatar, October 2014. Association for Computational Linguistics.

[37] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, Yasemin Altun, and Yoram Singer. Large margin methods for structured and interdependent output variables. 6(9).

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[39] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks, 2017.

[40] Shanchan Wu and Yifan He. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364, 2019.

[41] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.