A Semi-joint Neural Model for Sentence Level Discourse Parsing and Sentiment Analysis

by

Bita Nejat

B. Science, The University of British Columbia, 2014

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Computer Science)

The University of British Columbia
(Vancouver)

August 2017

Abstract

Discourse Parsing and Sentiment Analysis are two fundamental tasks in Natural Language Processing that have been shown to be mutually beneficial. In this work, we design and compare two Neural Based models for jointly learning both tasks. In the proposed approach, we first create a vector representation for all the segments in the input sentence. Next, we apply three different Recursive Neural Net models: one for discourse structure prediction, one for discourse relation prediction and one for sentiment analysis. Finally, we combine these Neural Nets in two different joint models: Multi-tasking and Pre-training. Our results on two standard corpora indicate that both methods result in improvements in each task but Multi-tasking has a bigger impact than Pre-training.

Lay Summary

In Natural Language Processing, gathering and processing human-labeled data requires considerable amount of time, money and resources. As a result of not having an abundance of such data, learning complex NLP tasks is a challenge. Therefore, being able to transfer and apply the knowledge learned in one task to another relevant task can be very beneficial.

In this work, we study two fundamental and closely related NLP tasks, Discourse Parsing and Sentiment Analysis, and explore two ways in which knowledge learned in one task could be transferred to the other task. Our research confirms that the knowledge-sharing between these two tasks helps boost the performance of each one individually.

Preface

This dissertation is an original intellectual product of the author, Bita Nejat.
I conducted all the experiments and wrote the manuscript. Dr. Raymond Ng and Dr. Giuseppe Carenini were the supervisory authors of this project and were involved throughout the project in concept formation and manuscript editing. This thesis is a paper-based thesis.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I would like to offer my wholehearted gratitude to everyone who has inspired or supported my work during my master study.
Special Thanks to my supervisors Dr. Giuseppe Carenini, and Dr. Raymond Ng. The past 3 years have been extremely challenging for me and you have been the most supportive, helpful and considerate supervisors that I could wish for. I would like to thank Julieta Martinez and Issam Laradji for their insightful comments on Neural Net libraries and Machine Learning techniques. Finally, a big thank you to my family for their unconditional love and support.

Chapter 1

Introduction

With the rapid growth of the amount of text data on the Internet, the need for methods to analyze these texts has grown as well. This thesis focuses on studying two fundamental NLP tasks, Discourse Parsing and Sentiment Analysis. The importance of these tasks and their wide applications (e.g., [10], [31]) has initiated much interest in studying both but no method yet exists that can come close to human performance in solving them.

It has been known that they are mutually beneficial, meaning that Discourse information could be used to improve Sentiment Analysis and likewise, knowing the sentiment of text spans within a sentence or document can help with improving Discourse Parsing. Our project relies on previous findings and borrowing from Transfer learning ideas to create a joint model that improves both Discourse Parsing and Sentiment Analysis.

1.1 Discourse Parsing

"*Clauses and sentences rarely stand on their own in an actual discourse; rather, the relationship between them carries important information that allows the discourse to express a meaning as a whole beyond the sum of its individual parts. Discourse analysis seeks to uncover this coherent structure.*" [13]

Discourse Parsing is the task of building a hierarchical tree structure over a

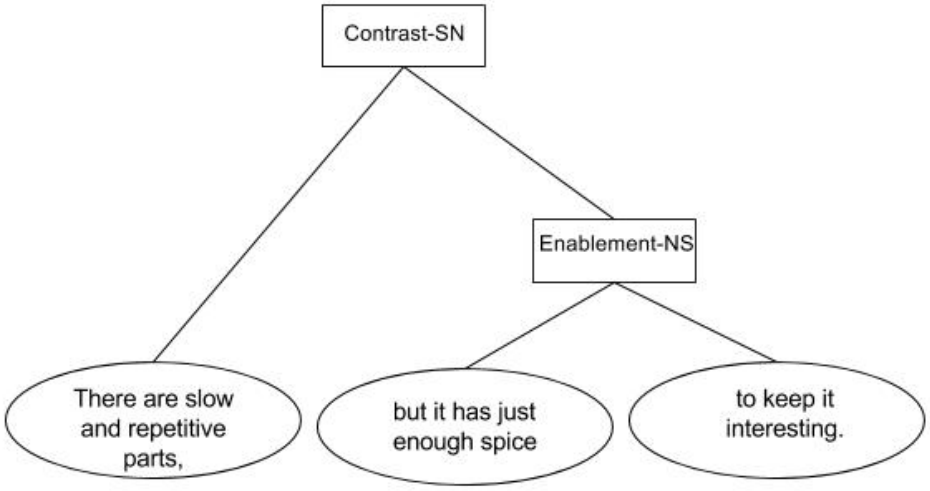


Figure 1.1: The Discourse Tree of a sentence from Sentiment Treebank dataset

sentence or a document, the leaves of which are clauses (also called Elementary Discourse Units) and nodes correspond to the juxtaposition of their children's text spans. A Discourse Tree is represented as a set of constituents $R[i,m,j]$ where $i \leq m < j$. R in this representation refers to the rhetorical relation that holds between the Discourse Unit containing EDUs i to m and the one containing EDUs $m+1$ to j.

The relation R also specifies nucliearity. Nuclei are the core parts of the relation and Satellites are the supportive ones.
R can take one of the following forms:

- Satellite-Nucleus (SN) : First Discourse Unit is Satellite and second Discourse Unit is Nucleus.
- Nucleus-Satellite (NS) : First Discourse Unit is Nucleus and second Discourse Unit is Satellite.
- Nucleus-Nucleus (NN) : Both Discourse Units are Nuclei.

In this approach relation identification and nuclearity assignment is done si-

multaneously. Figure 1.1 shows the Discourse Tree of a sample sentence. In this sentence, the Discourse Unit "There are slow and repetitive parts," holds a "*Contrast*" relationship with "but it has just enough spice to keep it interesting.". Furthermore, we can see that the former Discourse Unit is the satellite of the relation and the later part is the Nucleus.

Discourse Parsing is such a critical task in NLP because previous work has shown that information contained in the resulting Discourse Tree can benefit many other NLP tasks including but not restricted to automatic summarization (e.g., [10], [23], [20]), machine translation (e.g., [25],[11]) and question answering (e.g., [34]).

In contrast to traditional syntactic and semantic parsing, Discourse Parsing can generate structures that cover not only a single sentence but also multi-sentential text. However, the focus of this thesis is on sentence level Discourse Parsing, leaving the study of extensions to multi-sentential text as future work.

1.2 Sentiment Analysis

Given a piece of text, the task of Sentiment Analysis studies how a label, representing the contextual polarity of the text, can be assigned to it. Consider the following two examples:

- *"The whole cast looks to be having so much fun"*
- *"It's Robert Duvall."*

The sentiment for the first phrase is probably "Very Positive", while the sentiment label for the second phrase is probably "Neutral". Analyzing the overall polarity of a sentence is a challenging task due to the ambiguities that can be introduced by combinations of words and phrases. For example in the movie review excerpt shown in Figure 1.2, the phrase "There are slow and repetitive parts" has a negative sentiment. However when it is combined with the positive phrase "but it has just enough spice to keep it interesting", it results in an overall positive sentence.

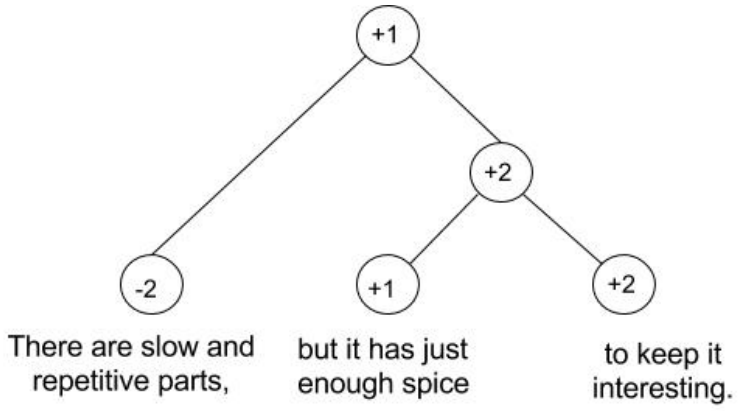


Figure 1.2: The Sentiment annotation (over Discourse Tree structure) of a sentence from Sentiment Treebank dataset

With a wide range of applications, including Review Analysis, Poll predictions and Recommender Systems [8], Sentiment Analysis has been one of the most studied and hottest areas of NLP, yet highly accurate Sentiment Analysis of a domain independent piece of text remains an ongoing research problem.

1.3 Motivation

It has been suggested that the information extracted from Discourse Trees can help with Sentiment Analysis [2] and likewise, knowing the sentiment of two pieces of text might help with the identification of discourse relationships between them [16]. For instance, taking the sentence in Figure 1.1 as an example, knowing that the two text spans "There are slow and repetitive parts" and "but it has just enough spice to keep it interesting" are in a *Contrast* relationship to each other, also signals that the sentiment of the two text spans is less likely to be of the same type. Likewise, knowing that the sentiment of the former text span is "very negative", while the sentiment of the later text span is "very positive", helps to narrow down the choice of discourse relation between these two text spans to the *Contrastive* group which contains relations *Contrast*, *Comparison*, *Antithesis*, *Antithesis-e*,*Consequence-s*,*Concession* and *Problem-Solution*.

To the best of our knowledge there is no previous work that learns both of these tasks in joint model, using deep learning architectures.

1.4 Approach and Contributions

The main contribution of this thesis is to address this gap by investigating how the two tasks can benefit from each other at the sentence level within a deep learning joint model. More specific contributions include:

(i) The generation of embeddings for each task which is obtained by compressing generic skip-thought vectors. [15]

(ii) The development of three independent recursive neural nets: two for the key sub-tasks of discourse parsing, namely structure prediction and relation prediction; the third net for sentiment prediction.

(iii) The design and experimental comparison of two alternative neural joint models, Multi-tasking and Pre-training, that have been shown to be effective in previous work for combining other tasks in NLP ([6],[7],[18]).

Our results indicate that a joint model performs better than individual models in either of the tasks with Multi-tasking outperforming Pre-training.

1.5 Outline

Chapter 2 describes the background and previous work done on Discourse Parsing, Sentiment Analysis and ways of jointly training multiple tasks. In Chapter 3, we talk about the two datasets that we have used to train and test our models. We also discuss their properties and any preprocessing work needed to prepare them for our specific tasks. Chapter 4 discusses our framework and all of its subparts in detail. We describe how we learn the embeddings for text spans, our individual Neural Net models and finally our joint models. We then present the training and evaluation process in Chapter 5. Chapter 7 talks about some possible future avenues to improve the results or extend the work to other, more complicated tasks. Finally, we wrap up the thesis in Chapter 8.

Chapter 2

Related Work

In this section we discuss related projects and their advantages and disadvantages. The related work is divided into three subsections that address the three main areas of our work. The first subsection explores previous work done on Discourse Parsing and Sentiment Analysis which are the two tasks we want to solve. The second subsection describes previous work on distributed vector representations for sentences and text spans. Since we do not hand pick features, having meaningful vector representations for text spans is key to obtaining more accurate results. The third subsection explores previous work on joint models, focusing on two techniques: Pre-training and Multi-tasking. In particular, we describe how Multi-tasking and Pre-training affects the performances of the tasks that are jointly trained.

2.1 Discourse Parsing and Sentiment Analysis

Traditionally, **Discourse Parsing and Sentiment Analysis** have been approached by applying machine learning methods with predetermined, engineered features that were carefully chosen by studying the properties of the text.

Examples of effective sentence level and document level Discourse Parsers include CODRA [13] and the parser of [9] . These parsers use organizational, structural, contextual, lexical and N-gram features to represent Discourse Units and apply graphical models for learning and inference (i.e. Conditional Random Fields).

The performance of these parsers critically depends on a careful selection of informative and relevant features, something that is instead performed automatically in the neural models we propose in this thesis.

[27], [28] and [30], approach Sentiment Analysis using carefully engineered features as well as polarity rules. The choice of features also plays a key role in the high performance of these models.

Yet, with the rapid advancements of Neural Nets in complex areas such as vision and speech understanding, there has been increased interest in applying them to different NLP tasks. Socher et al. [33] approached the problem of Sentiment Analysis by recursively assigning sentiment labels to the nodes of a binarized syntactic parse tree over a sentence. At each non-leaf node, the Sentiment Neural Net first creates a distributed embedding for the node using the embedding of its two children and then assigns a sentiment label to that node. Their approach achieves state of the art results. In our work, we borrow from the same idea of Recursive Neural Nets to learn the Sentiment labels. However, the structure over which we learn the Sentiment labels is the Discourse Tree of the sentence as opposed to the syntactic parse tree, with the goal of testing if Sentiment Analysis can benefit directly from discourse information within a neural joint model.

Motivated by Socher's success on Sentiment Analysis, Li et al. [17] approached the problem of Discourse Parsing by recursively building the Discourse Tree using two Neural Nets. A Structure Neural Net decides whether two nodes should be connected in the Discourse Tree or not. If two nodes are determined to be connected by the Structure Neural Net, a Relation Neural Net then decides what rhetorical relation should hold between the two nodes. Their approach also yields promising results. In terms of representation, the recursive structure of a Discourse Tree is used to learn the embedding of each non-leaf node from its children. For leaf nodes (EDUs), the representation is learned recursively using the syntactic parse tree of the node. One problem with their work is that it is unclear how they combine the labeled Discourse Structure Tree with the unlabeled syntactic parse trees to learn the vector representations for the text spans.

Bhatia et al. [2] trained a Recursive Neural Network for Sentiment Analysis over a Discourse Tree and showed that the information extracted from the Discourse Tree can be helpful for determining the Sentiment at document level. In

their work however, they did not attempt to learn a distributed representation for the sub-document units. To represent EDUs, they used the bag-of-words features. For our work, we not only apply a Recurrent Neural Net approach to learn embeddings for the EDUs, but we also jointly learn models for the two tasks, instead of simply feeding a pre-computed discourse structure in a neural model for sentiment.

2.2 Learning Text Embeddings

Learning text embeddings is a fundamental step in using Neural Nets for NLP tasks. An embedding is a fixed dimensional representation of the data (text) without the use of handpicked features. As words are the building blocks of text, previous studies have created fixed dimensional vector representations for words [26] that capture the semantic and syntactic meaning of the words. However, creating meaningful fixed dimensional vector representations for text spans is an ongoing challenge.

Both Socher et al. [33] and Li et al. [17] learn the embedding of a text span in a recursive manner, given a binary tree over the text span with leaves being the words. The words are initialized with random vector representations and the embedding of a parent is computed from the embedding of its two children using a non-linear projection. The embedding is then used for training the task under study (Sentiment Analysis and Discourse Parsing respectively) and updated according to how useful it was for the task.

Recently Recurrent Neural Nets (RNNs) and their variant, Long Short-Term Memories (LSTMs) have become a more popular alternative for learning the embedding of a sentence ([15] and [29]).

In [15], an encoder RNN encodes a sentence into a fixed vector representation that is then used by a decoder RNN to predict the following and preceding sentences and based on how good the predictions were, updates both the decoder and encoder RNNs. Once training is done, the encoder RNN can be used on its own to create an embedding for any text span. In their work, Kiros et al. [15] applied skip thought vectors to Sentiment Treebank sentences to see if the representations learned could directly be used for determining the sentiment of a sentence. Their

results showed that representing a sentence with the skip-thought vectors without taking its structure into consideration would not improve the performance beyond the results Socher et al. [33] had achieved. In this project, we have used the encoder RNN to represent our EDUs but we further compress the resulting embeddings with a neural based compressor to limit the number of parameters.

2.3 Joint models

When training a neural model, the weights are usually initialized with random numbers taken from a uniform distribution. However, in their work, [7] argue that **Pre-training** a neural model helps initialize a neural network with better weights that prevent the network from getting stuck in local minima and results in better generalization and can enhance the performance of the model. And this general idea has been successfully applied in several scenarios (e.g., [5], [32]). For example, Chung et al. [5] used auto-encoders as a Pre-training mechanism and showed that Pre-training can lead to better performance compared to the same model with no Pre-training. In our work, we use the trained weights of one neural model (e.g. sentiment) as an initialization form for another task (e.g. discourse structure) to see if the features learned for one can be helpful for the other.

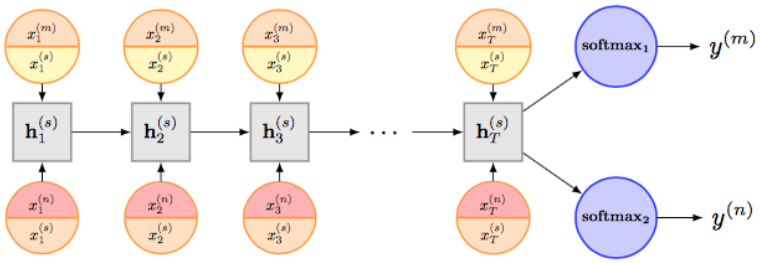
Neural **Multi-tasking** was originally proposed by [6], who experimented with the technique using deep convolutional neural networks. In essence, the basic idea is that a network is alternatively trained with instances for different tasks, so that the network is learning to perform all these tasks jointly. In [6] a model is trained to perform a variety of predictions on a given sentence, including part-of-speech tags, chunks, named entity tags, semantic roles, semantically similar words and the likelihood that the sentence makes sense using a language model. They showed that multitasking using a neural net structure can improve the generalization of the shared tasks and result in better performance. Following up on this initial success, many researchers have applied the neural multi-tasking strategy to several tasks, including very recent work in vision [14] and NLP (e.g., text classification [18] and the classification of implicit discourse relations [19]).

Liu et al [18] showed that a multi-tasking system can improve the performance

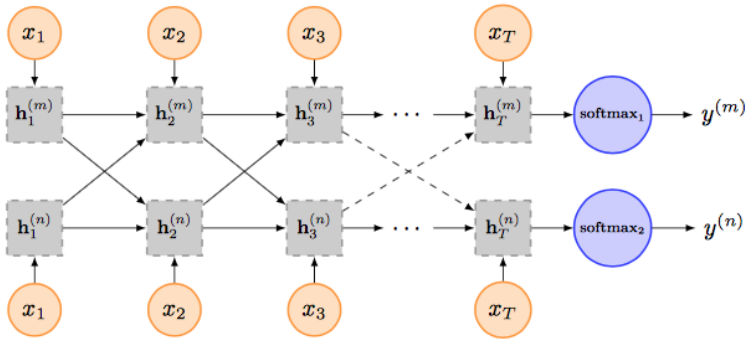
of a task with the help of other related tasks. The goal is to learn representations for phrases, text spans and sentences using Recurrent Neural Nets (RNNs) through supervised training of four related tasks. They proposed three multi-task models as shown in Figure 2.1. In the first model (Model-I in the Figure), there is only one, shared RNN for all tasks (but producing different outputs). In the second model (Model-II in the Figure), each task has its own RNN but each RNNs shares the hidden representations at each time step with the other RNNs at the same time step. In the last model (Model-III in the Figure), each task has its own RNN but each RNN is connected to a mutually shared RNN to share hidden representations at each time step.

Experimental results on each of the three Multi-task models showed significant improvements compared to the individual model. The amount of improvement on varied among tasks and datasets. The first model resulted in an average of 2% improvement, while the second model resulted in an average of 2.3% improvement. The average improvement for the third model with added Pre-training and fine tuning was 2.8%.

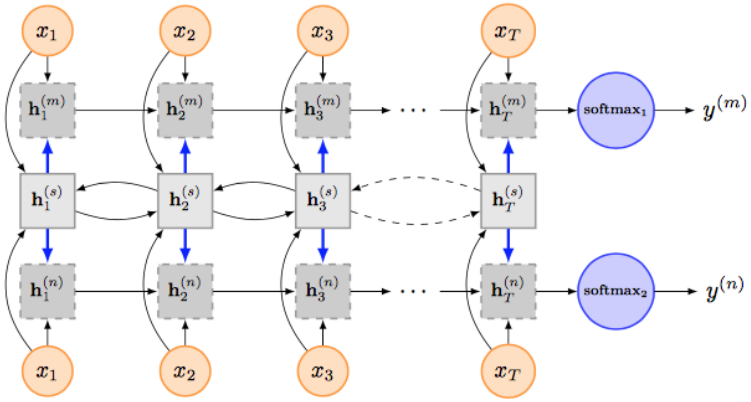
In their work [19], Liu et al studied implicit discourse relation classification using Mutli-task learning of four related tasks. As can be seen from Figure 2.2, their Neural based model consists of a Convolutional layer that compresses the augument pairs of different tasks into low-dimensional vector representations. Each task owns a unique representation and a shared representation connecting all tasks. The two are then concatenated and mapped into a task specific representation. They then attach additional surface-level features and the resulting vector representation is fed to each task's Neural Net.



(a) Model-I: Uniform-Layer Architecture



(b) Model-II: Coupled-Layer Architecture



(c) Model-III: Shared-Layer Architecture

Figure 2.1: Three architectures for modelling text with multi-task learning. (Figure adapted from [18])

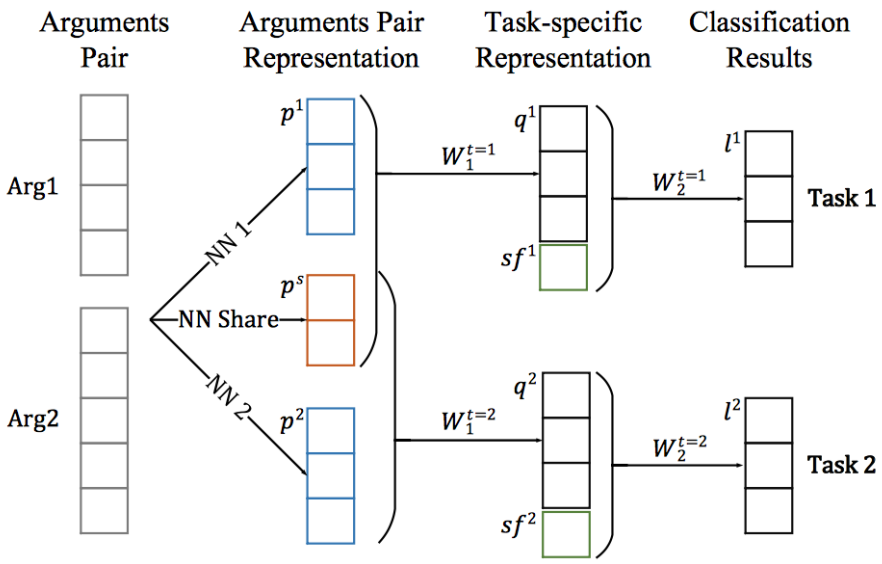


Figure 2.2: Architecture of Multi-task Neural Networks for Discourse Relation Classification (Figure adapted from [19])

Their Multi-tasking model is trained over four different tasks:

- Implicit PDTB Discourse Relation Classification using Penn Discourse Treebank
- Explicit PDTB Discourse Relation Classification using Penn Discourse Treebank
- RST-DT Discourse Relation Classification using RST-DT
- Connective Word Classification using New York Times Corpus

Their experimental results show that a multi-task model achieves significant improvements over individual models. The amount of improvement is different among different relations ranging from a minimum of %2 to a maximum of 16% improvement in classifying each implicit relation .

[29] have also used multitasking and Deep Neural architectures for Semantic Dependency Parsing. In its basic form as shown in Figure 2.3, their model included

a layer of bi-directional Long Short Term Memories (BiLSTM) for representing the sentences, followed by two layers of Deep Neural Networks to predict dependency relations in a parse tree. In this work, they explored two multitask learning approaches. In the first approach, parameters of the BiLSTM part of the model were shared among the tasks. In the second approach, higher-order structures were used to predict the graphs jointly. Their work on a jointly trained multitask system showed statistically significant improvements over an individual model. However, the improvement was rather small, from 87.4% to 88%.

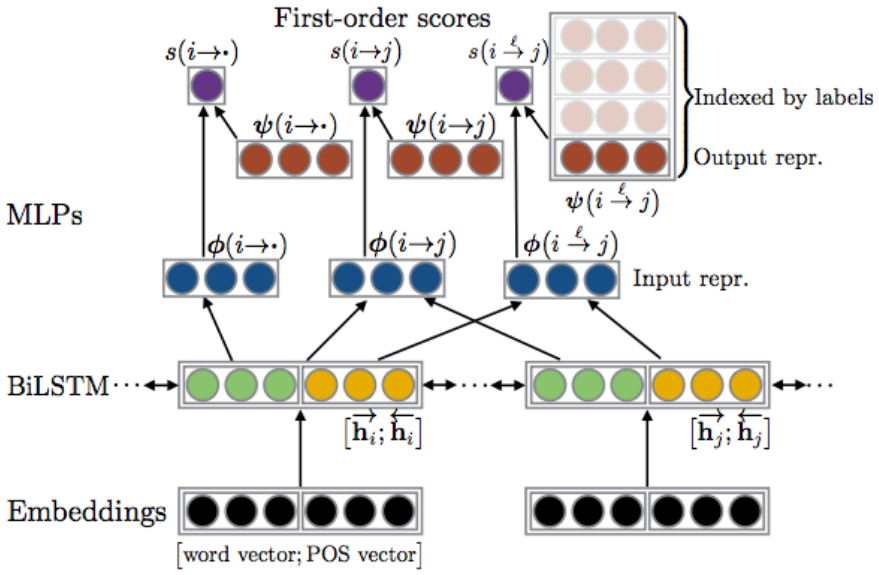


Figure 2.3: Illustration of the architecture of the basic model. (Figure adapted from [29])

In all these projects we notice that Multi-tasking benefits all the tasks in which it was applied but we also observe that the benefit varies from task to task and model to model with some tasks getting more benefit out of multi-tasking than others.

Chapter 3

Corpora

For the task of Discourse Parsing, we use RST-DT ([3], [4]). This dataset contains 385 documents along with their fully labeled Discourse Trees. The annotation is based on the Rhetorical Structure Theory (RST), a popular theory of discourse originally proposed in [21]. All the documents in RST-DT were chosen from Wall Street Journal news articles taken from the Penn Treebank corpus [24]. Since we are focusing only on sentence-level discourse parsing, the documents as well as their Discourse Trees were first preprocessed to extract the sentences and sentence-level Discourse Trees. The sentence-level Discourse Trees were extracted from the document-level Discourse Tree by finding the sub-tree that exactly spans over the sentence. This resulted in a dataset of 6846 sentences with well-formed Discourse Trees, out of which 2239 sentences had only one EDU. Since sentences with only one EDU have trivial Discourse Trees, these sentences were excluded from our dataset, leaving a total of 4607 sentences.

For the task of Sentiment Analysis, we use the Sentiment Treebank [33]. This dataset consists of 11855 sentences along with their syntactic parse trees labeled with sentiment labels at each node. For this work, since our models label sentiment over a Discourse Tree, we had to preprocess the datasets in the following way. For each sentence in the Sentiment Treebank dataset, a Discourse Tree was created using [13]. Next, for each node of the discourse tree, a sentiment label was extracted from the corresponding labeled syntactic tree by finding a subtree that exactly (or almost exactly) matches the text span represented by the node in the discourse tree.

Relation	percentages	Relation	percentages
elaboration	33.29	temporal	2.38
attribution	23.00	condition	1.91
same-unit	10.53	comparison	1.52
joint	5.62	manner-means	1.43
enablement	4.21	evaluation	1.10
background	4.13	summary	0.78
contrast	3.97	topic-comment	0.12
cause	3.48	topic-change	0.03
explanation	2.44		

Table 3.1: Distribution of RST-DT relations

Exact match was not possible when the syntactic and the discourse structures were not fully aligned, which happened in 31.9% of the instances. In this case, an approximation of the sentiment was computed by considering the sentiment of the two closest subsuming and subsumed syntactic sub-trees.

Both datasets were highly unbalanced across different classes. In the case of RST-DT, the discourse relations outlined in [21], were further grouped under 16 classes (also outlined in [21]). Table 3.1 shows the distribution of each of these 16 classes of relations across RST-DT at each node of the sentence level discourse trees for sentences with more than one EDU. Notice that after adding the appropriate nuclearity labels (explained in Section 1.1) to these sets, we get a total of 41 different sets of relations since some of the relations can only take one of the three forms of *"-NS"*, *"-SN"* or *"-NN"*. From this table we can see that some of the relations are very infrequent and some hardly ever appear at the sentence level.

Figure 3.1 shows the distribution of sentiment labels of Sentiment Treebank sentences at all levels of the Discourse Tree created over them as described above. As we can see from the figure, the majority of text spans in Sentiment Treebank are "neutral", followed by "positive" and "negative" labels. "very negative" and "very positive" labels are much more infrequent than others.

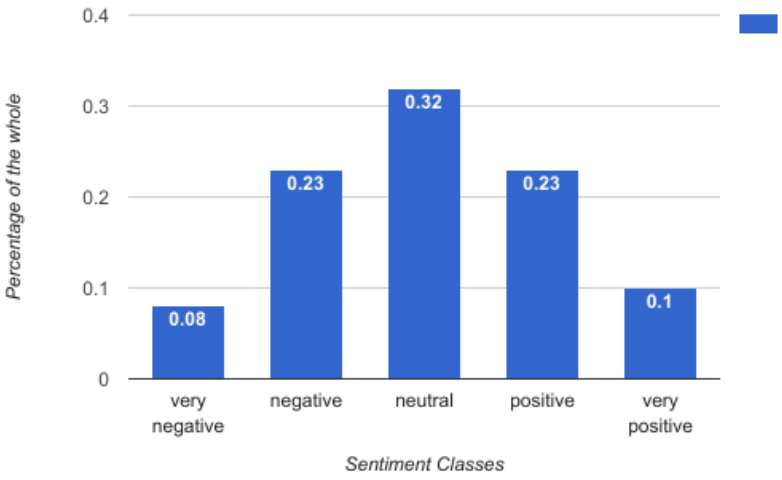


Figure 3.1: Distribution of sentiment labels over Sentiment Treebank sentences at all nodes of created Discourse Trees

Chapter 4

Joint Models

Our framework consists of three main sub parts. Given a segmented sentence, the first step is to create meaningful vector representations for all the EDUs. This is discussed in the first section. Next, we devise three different Recursive Neural Net models, each designed for one of discourse structure prediction, discourse relation prediction and sentiment analysis. In section 4.2, we discuss the structure of these Neural Nets in detail. Finally, we join these Neural Nets in two different ways: Multitasking and pre-training. The final section of this Chapter talks about these two ways of joining the Neural Nets.

4.1 Learning Text Embeddings

One of the most challenging aspects of designing effective Neural Nets is to have meaningful representations for the inputs. Since we refrain from hand picking features, and choose to feed text spans consisting of multiple words to the Neural Nets are our inputs, it is very important to come up with vector representations that are generalizable but also meaningful for the two tasks that we approach.

Initially, we considered directly applying the Skip-thought framework [15] to each text span to get generic vector representations for them, since the original Skip-thought vectors were shown in [15] to be useful for many NLP tasks. However, given the size of our datasets (only in the thousands of instances), it was clear that using 4800-dimensional Skip-thought vectors would have created an

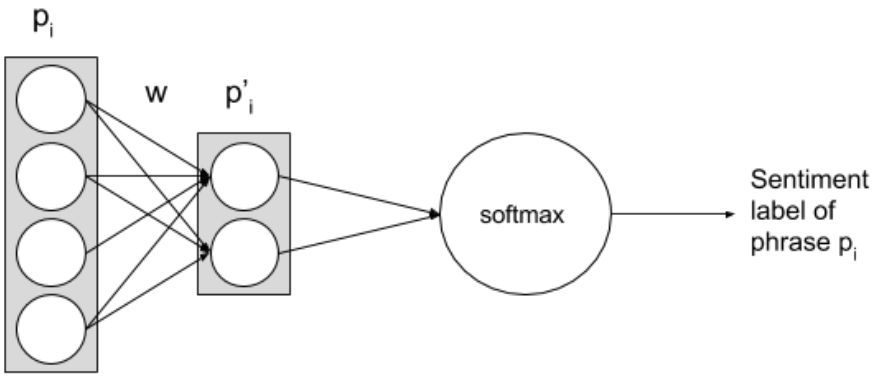


Figure 4.1: The Sentiment Neural Compressor

over-parametrized network prone to over-fitting. Based on this observation, in order to simultaneously reduce the dimensionality and to produce vectors that are meaningful for our tasks, we devised a compression mechanism that takes in the Skip-thought produced vectors and compresses them using a Neural Net. Figures 4.2 and 4.1 show the structure of these compressors for our two different tasks.

The sentiment neural compressor (Figure 4.1) takes as input, the skip-thought produced vector representations for all phrases of the Sentiment Treebank in the training set. For example, consider a phrase i with skip-thought produced vector $P_i \in R^{4800}$. The Sentiment Neural Compressor learns compressed vector $P'_i \in R^d$ through

$$P'_i = f(W.P_i) \tag{4.1}$$

where f is a non-linear activation function such as *relu* and $W \in R^{d \times 4800}$ is the matrix of weights. This Neural Net uses the sentiment of phrase i for supervised learning of the weights.

Similarly, the Discourse Parsing neural compressor (Figure 4.2) takes the skip-thought produced vector representations for two EDUs e_i, e_j that are connected in their Discourse Tree and learns the compressed vectors e'_i and e'_j, each with d

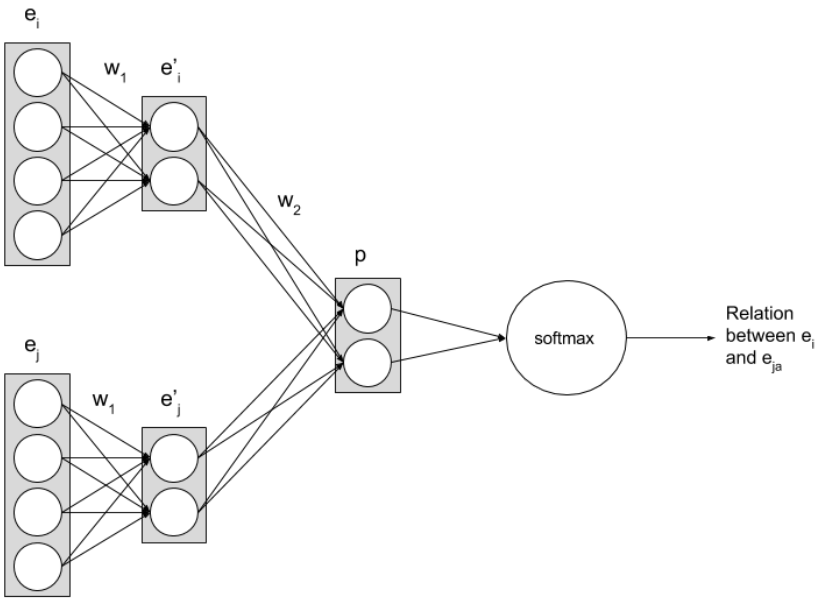


Figure 4.2: The Discourse Neural Compressor

dimensions where

$$\begin{aligned} e'_i &= f(W_1.e_i) \\ e'_j &= f(W_1.e_j) \end{aligned} \tag{4.2}$$

where f is again a non-linear activation function such as *relu* and $W_1 \in R^{d \times 4800}$ is the matrix of weights. Note that the same set of weights are used for both EDUs because we are looking for a unique set of weights to compress an EDU.

4.2 Neural Net Models

Following [33]'s idea of Sentiment Analysis using recursive Neural Nets, we designed three Recursive Neural Nets for each task of Discourse Structure prediction, Discourse Relation prediction and Sentiment Analysis. All these three Neural Nets

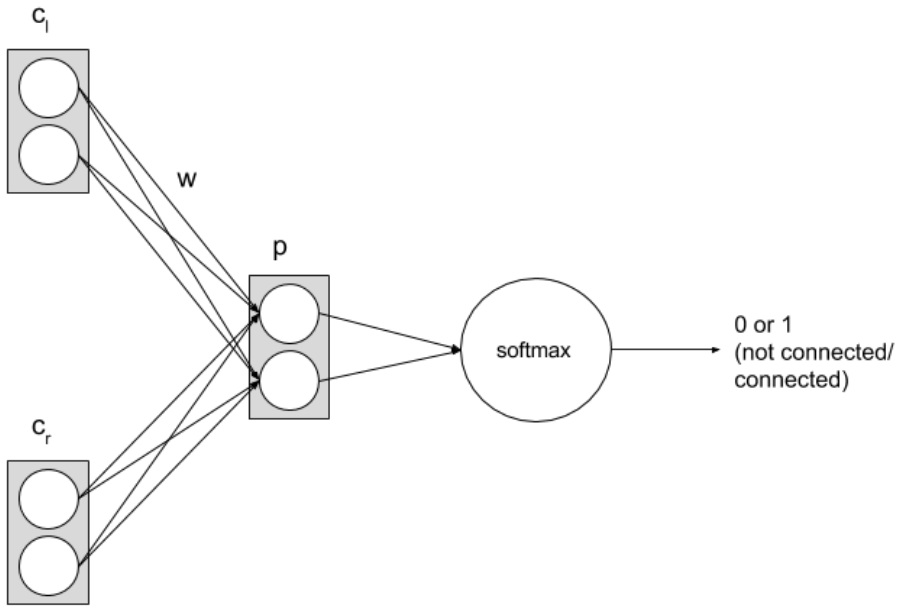


Figure 4.3: The Discourse Structure Neural Net

are classifiers.

The Structure Neural Net takes in the compressed vector representation ($\in R^d$) for two Discourse Units and learns whether they will be connected in the Discourse Tree (Figure 4.3). In this process, it also learns the vector representation for the parent of these two children. So for a parent p with children c_l and c_r, the vector representation for the parent is obtained by:

$$p = f(W_{str}[c_l, c_r] + b_{str}) \tag{4.3}$$

where $[c_l, c_r]$ denotes the concatenating vector for the children; f is a non-linearity function; $W_{str} \in R^{d \times 2d}$ and $b_{str} \in R^d$ is the bias vector.

The Relation Neural Net takes as input the compressed vector representation for two Discourse Units that are determined to be connected in the Discourse Tree and learns the relation label for the parent node. The Relation Neural Net is the same in structure as the Structure Neural Net in Figure 4.3.

The Sentiment Neural Net takes as input the compressed vector representation for two Discourse Unit that are determined to be connected in the Discourse Tree and learns the sentiment label for the parent node. This Neural net also shares the same structure as the one in Figure 4.3.

4.3 Individual Models

Before joining the models using pre-training or multi-tasking, each task is trained individually as a baseline. Algorithm 1 describes the training process for an individual model (before joining) which is a standard 10-fold cross validation with the addition of a Neural Compression step.

Algorithm 1 Training an individual Neural Net

for $i \leftarrow 0$ to 10 **do**
  $train\_set \leftarrow$ load the training set for fold i
  $test\_set \leftarrow$ load the test set for fold i

  $train\_set \leftarrow$ Train Neural Compressor on $train\_set$, and compress its vectors
  $test\_set \leftarrow$ Compress $test\_set$ vectors using the trained Neural Compressor

  Train the recursive Neural Net on $train\_set$
  Test the recursive Neural Net on $test\_set$
end for

At test time, for the task of Sentiment Analysis, given a sentence, with its Discourse Structure tree, each node of the discourse tree is labeled with a sentiment label representing the sentiment of the text span the node corresponds to.

However, for the task of Discourse Parsing, given a sentence a the discourse tree needs to be created and labeled with discourse relations. To build the most probable tree, a CKY-like bottom-up parsing algorithm that uses dynamic programming to compute the most likely parses is applied [13].

4.4 Joint Models

Our hypothesis in creating a joint model is that the accuracy of prediction obtained in a joint design would be higher than the accuracy of prediction coming from independent Neural Nets applied to each task. We explore two ways of creating a joint model. For both approaches, we train three neural nets (Discourse Structure, Discourse Relation and Sentiment Neural Nets) that interact with one another for improved training. The input to the Structure net are all possible pairs of text spans that can be connected in a Discourse Tree. The input to the Relation and Sentiment nets are the pairs of text spans that are determined to be connected by the Structure net.

Inspired by **Multitasking** [6], our goal is to find a representation for the input that will benefit all the tasks that need to be solved. Since the first layer in a Neural Net learns relevant features from the input embedding, in this approach, the first layer is shared between the three Neural Nets and training is achieved in a stochastic manner by looping over the three tasks. As shown in Figure 4.4, at each time step, one of the tasks is selected along with a random training example for that task. Afterwards, the neural net corresponding to this task is updated by taking a gradient step with respect to the chosen example. The end product of this design is a joint input representation that could benefit both Sentiment Analysis and Discourse Parsing.

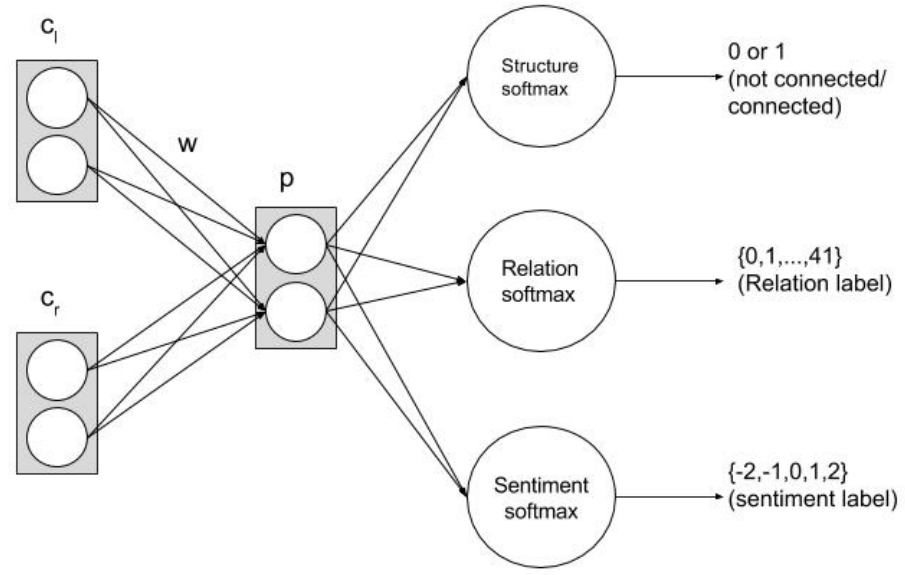


Figure 4.5: Multi-tasking Network

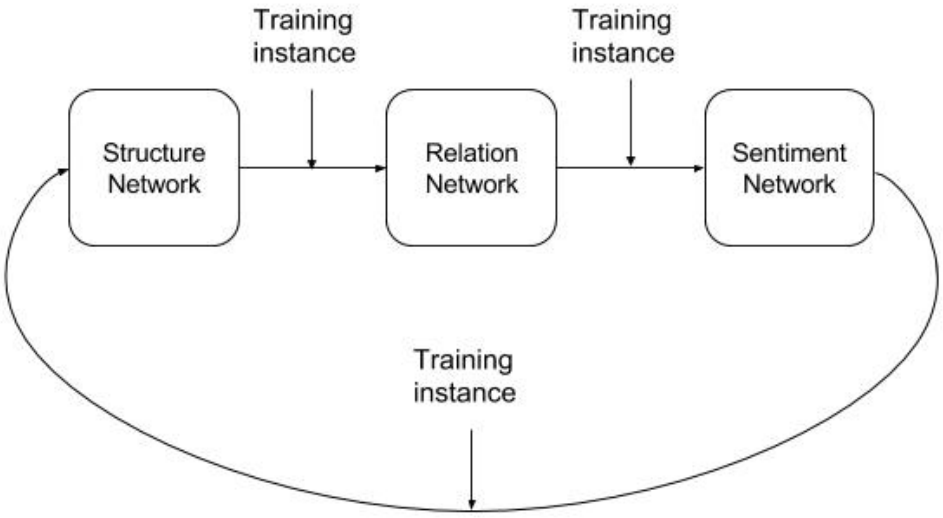


Figure 4.4: Multi-tasking

Inspired by **Pre-training Neural Nets** [7], in this approach we study how the

parameters of one Neural Net after training can be used as a form of initialization for the network applied to the other task. As shown in Figure 4.6, in this setting, we first fully train the Discourse Structure Neural Net, then the weights from this trained net are used to initialize the Discourse Relation Neural Net and once this net is fully trained, its weights are used to initialize the weights of the Discourse Structure Neural Net again. After another round of training the Discourse Structure Neural Net, its weights are used to initialize the Sentiment Neural Net. After training the Sentiment Neural Net, its weights are again used to initialize the Structure Neural Net. We experimented with 2,3 and 10 iterations using 10-fold cross validation on the datasets and achieved best results with 3 iterations, which appears to be a good compromise between accuracy and training time. Algorithm 2 describes the training process for the Pre-training setting. Notice that in this setting, we need both Sentiment and Discourse Neural Compressors, where each one would be trained once on their relevant set of data before entering the training loop.

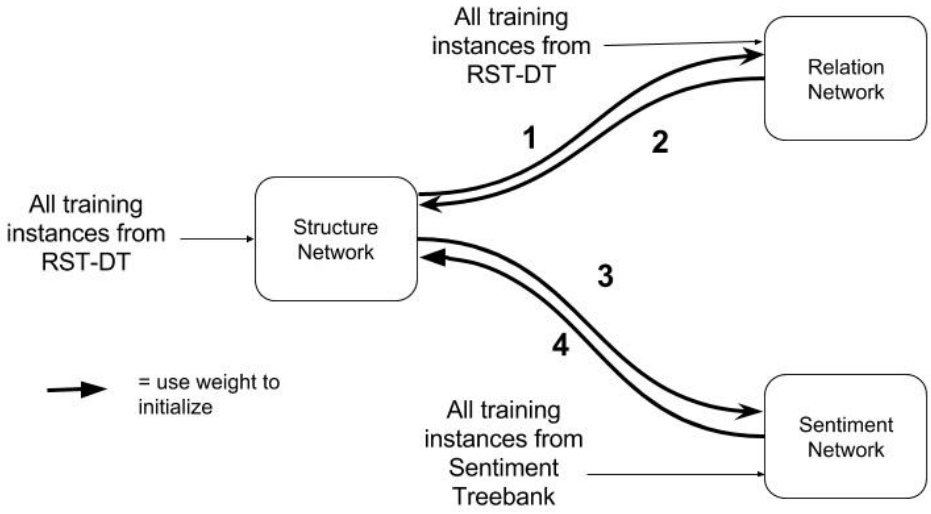


Figure 4.6: Using the weights of one network as a form of pre-training for another network

Algorithm 2 Training Model in Pre-training setting

1: **for** $i \leftarrow 0$ to 10 **do**
2: $sentiment\_train\_set \leftarrow$ load the sentiment training set for fold i
3: $sentiment\_test\_set \leftarrow$ load the sentiment test set for fold i
4: $discourse\_train\_set \leftarrow$ load the discourse training set for fold i
5: $discourse\_test\_set \leftarrow$ load the discourse test set for fold i

6: $sentiment\_train\_set \leftarrow Sentiment\_Neural\_Compressor.train(sentiment\_train\_set)$
(compresses its vectors as well)
7: $sentiment\_test\_set \leftarrow Sentiment\_Neural\_Compressor.compress(sentiment\_test\_set)$

8: $discourse\_train\_set \leftarrow Discourse\_Neural\_Compressor.train(discourse\_train\_set)$
(compresses its vectors as well)
9: $discourse\_test\_set \leftarrow Discourse\_Neural\_Compressor.compress(discourse\_test\_set)$

10: **for** $j \leftarrow 0$ to pre_train_itr **do**
11: $Structure\_Neural\_Net.train(discourse\_train\_set)$
12: $Relation\_Neural\_Net.train(discourse\_train\_set)$
13: $Structure\_Neural\_Net.train(discourse\_train\_set)$
14: $Sentiment\_Neural\_Net.train(sentiment\_train\_set)$
15: **end for**

16: $Discourse\_Neural\_Net.test(discourse\_test\_set)$
17: $Sentiment\_Neural\_Net.test(sentiment\_test\_set)$
18: **end for**

Chapter 5

Training and Evaluating The Models

All the neural models presented in this project were implemented using the TensorFlow python package [1]. We minimize the cross-entropy error using the Adam optimizer and L2-regularization on the set of weights. For the individual models (before joining), we use 200 training epochs and a batch size of 100.

We evaluate our models using 10-fold cross validation on the sentiment treebank and on RST-DT. All the experiments were based on manual Discourse Segmentation. In Table 5.1 and Table 5.3, a star indicates that there is statistical significance with a p-value less than 0.05. The significance is with respect to the joint model vs the model before joining.

For the task of Discourse parsing, the three predictions are: whether two discourse units should be connected (span), what relation holds between them (relation) and which one is the nucleus (Nuclearity). For these three sub_tasks, the metrics used to evaluate the model are Precision, Recall and F_score proposed by Marcu [22]. Since we are using manual discourse segmentation, Precision, Recall and F_score are the same and so we only show the F_score.

The results for Discourse Parsing are shown in Table 5.1. We have used the 41 relations outlined in [21] for training and evaluation of the Relation prediction. From the results, we see some improvement on Discourse Structure prediction when we are using a joint model but the improvement is statistically significant

Approach	Span	Nuclearity	Relation
Discourse Parser (Before Joining)	93.37	73.38	57.05
Joined Model Pre-training	**94.35**	74.92	58.82
Joined Model Multi-tasking	94.31	**75.91***	**60.91***

Table 5.1: Discourse Parsing results based on manual discourse segmentation

Relation \ Setting	Individual	Pre-training	Multi-tasking
Comparison	18.97	20.87	27.08
Contrast	15.19	17.74	20.83
Cause	7.6	8.11	8.61
Average	13.92	15.57	18.84

Table 5.2: Contrastive Relation Prediction results under different training settings

only for the Nuclearity and Relation predictions. The improvements on the Relation predictions were mainly on the Contrastive set ([2]), specifically the class of *Contrast*, *Comparison* and *Cause* relations as defined in [21]. The result for each of these relations under different training settings are shown in Table 5.2. Notice that the accuracies may seem low but because we train over 41 classes of relations, a random prediction results in 2.43%. Among the contrastive relations, the *Problem-Solution* did not improve due to the fact that this relation is hardly seen at the sentence level. This confirms our hypothesis that knowing the sentiment of the two Discourse Units that are connected in a discourse tree can help with the identification of the discourse relation that holds between them.

For the task of Sentiment Analysis, the results are shown in Table 5.3. To train

the model, we use the five classes of sentiment ({*very negative, negative, neutral, positive, very positive*}) used in [33]. We measure the accuracy of prediction in two different settings. In the fine grained setting we compute the accuracy of exact match across five classes. In the Positive/Negative setting, if the prediction and the target had the same sign, they were considered equal[1]. The huge difference in accuracy between these two settings signals that distinguishing between *very positive* and *positive* and distinguishing between *very negative* and *negative* is very hard. The results of sentiment shown in Table 5.3 are also consistent with our hypothesis. When jointly trained with Discourse Parsing, we can get a better performance on labeling nodes of the Discourse Tree with sentiment labels compared to an individual sentiment analyzer applied to a Discourse Tree.

Interestingly, if we compare the two joint models across both tasks it appears that Multi-tasking does better that Pre-training in all cases except for discourse structure. A possible explanation is that by transferring weights from one network to another (as done in Pre-training), the neural net starts learning with a possibly better initialization of the weights. However Multi-tasking performs a joint learning at the finer granularity of single training instances and so an improvement in learning one task immediately affects the next.

All results in Table 5.1 and 5.3 were obtained by setting the dimension d of the compressed vectors to 100. Experimentally, we found that the performance of the model was rather stable for $d \in \{1200, 600, 300, 100\}$ and was substantially lower for $d \in \{50, 25\}$.

In terms of actual runtime, Pre-training and the individual models are an order of magnitude faster than the Multi-tasking model. This is because even though they require a larger number of epochs to converge (200 for individual, vs 6 for Multi-tasking), they can be trained in parallel.

Notice that training and testing of the networks is done on Sentiment Treebank for sentiment analysis and on RST-DT for discourse parsing. [13]'s Discourse parser was run on Sentiment Treebank to get the sentiment annotation at the granularity required for the joint model with discourse. However, having a gold dataset of sentiment labels corresponding to discourse units could further improve the results.

[1]Notice that this is different from training a classifier for binary classification, which is a much easier task (see [2])

Approach	Fine grained		Positive/Negative	
	All	Root	All	Root
Sentiment Analyzer (Before Joining)	43.37	40.6	52.86	51.27
Joined Model Pre-training	42.46	40.36	53.82	53.15
Joined Model Multi-tasking	**45.49***	**44.82***	**55.52***	**54.72***

Table 5.3: Sentiment Analysis over Discourse Tree

Chapter 6

Discussion

Several differences between this work and previous approaches make direct comparisons challenging and possibly not very informative. In this section, we highlight and explain the differences between our work and the two most recent Sentiment Analyzers and Discourse Parsers.

6.1 Comparison to previous Sentiment Analyzers

Socher et al. ([33]) use syntactic trees, as opposed to discourse trees, as the recursive structure for training. Due to this underlying structural difference, we cannot compare our "All"-level results with those of his. For "Root"-level, which represents the sentiment prediction for the whole sentence, Socher et al. reports 45.7% fine-grained sentiment accuracy compared to 44.82% of our Multi-tasking. This difference is unlikely to be significant while the sentiment annotation of syntactic structure is definitely more costly than one at the EDU level because a syntactic parse tree of a sentence has considerably more nodes than the sentence's discourse tree.

Bhatia et al. ([2]) focuses on document level Sentiment Analysis, using bag-of-word features for EDUs. While in future, our work can be extended to document level sentiment analysis, the model we use learns the distributed representations of the EDUs, which will remain a key difference between our work and that of Bhatia

et al. Furthermore, Bhatia et al. train a binary model while assuming the discourse tree as given. In our approach, we train a 5-class model while joinly learning the discourse tree.

6.2 Comparison to previous Discourse Parsers

Since our work focuses on sentence-level Discourse Parsing, we cannot compare with Li et al. ([17]) because they only report overall results without differentiating sentence vs document level.

Our model is outperformed by CODRA ([13]) which achieves better performance on sentence level Discourse Parsing. While we believe that with more training data, as it has been shown with other NLP tasks, we would eventually outperform CODRA, the primary goal of our work is not to beat the state of the art on each single task, but to show how the two tasks of Discourse Parsing and Sentiment Analysis can be jointly performed in a neural model.

Chapter 7

Future Work

From incorporating more data, to more complicated models and even benefiting from other relevant tasks, there is so much that can be done to improve the results and to increase generalization of these models. Below we will discuss some possible avenues for future work.

7.1 Improving Corpora

In this work, we used a pre-existing Discourse Parser to parse the sentences of Sentiment Treebank into Discourse Trees and then tried to label the nodes of the produced discourse trees with sentiment labels through a matching process. Pre-existing discourse parsers are not perfect and introduce some error. Furthermore, to that, the matching process described in Chapter 3 is also error prone. The best way to go about this is to either use crowd-sourcing for producing correct discourse trees and sentiment labels or to use multiple pre-existing discourse parsers to produce high quality discourse trees (see [12]) and follow it up with crowd-sourcing to label each node with more accurate sentiment labels.

7.2 More Data

The amount of data greatly affects the performance of a Neural Nets. With more data, networks can better learn the patterns, and the results are more reliable. In this project, the number of training instances for the Structure Neural Net was 8639, for

Approach	Sentence Level		Document Level
	Manual Segmentation	Automatic Segmentation	Automatic Segmentation
Span	95.4	80.1	83.84
Nuclearity	88.6	75.2	68.90
Relation	78.9	66.8	55.87

Table 7.1: CODRA's Discourse Parsing results at sentence-level and document-level, based on manual and automatic segmentation

Relation Neural net was 7892 and for Sentiment Neural net was 5381, while the number of parameters that need to be trained for each network are around 20,000. Using methods that can augment the datasets [12] could help with the issue of small dataset size.

7.3 Recursive Neural Nets coupled with Recurrent Neural Nets

Granted more data, one could learn the representation for the EDUs using a variant of Recurrent Neural Nets that are jointly trained with the Recursive model. This solution can eliminate the need for a Neural Compressor applied to Skip-thought vectors because text embeddings for EDUs can be learned in any desired dimension directly. A similar model is described by Peng et al. [29] and shown to be helpful in learning task-specific meaningful representations.

7.4 Document Level Discourse Parsing and Sentiment Analysis

It is relatively easier to perform sentence-level discourse parsing and sentiment analysis than the same tasks performed on multiple sentences or at document level. As an example, the results achieved by CODRA [13] (also shown in the Table 7.1) indicates that document-level discourse parsing is much harder than sentence level discourse parsing. Using neural nets, a similar behaviour may be present but more data, better vector representations of text spans as well as experimenting with more complicated models can help minimize the drop.

It would also be interesting to observe how the performance of Sentiment

Analysis would be affected at the Document level using Multi Tasking Neural Networks. Just as determining the sentiment of sentences is harder than determining the sentiment of words, determining the sentiment of documents is harder than determining the sentiment of sentences. Bhatia et al. [2] reported an 84.1% accuracy on document level binary Sentiment Analysis. However, in their work, they used a vector constructed from bag-of-words features to represent the EDUs. As a possible future work, one could look at the fine-grained (5 class) document level Sentiment Analysis and the effects of learning the text span embeddings when scaled to documents.

7.5 Simultaneous Pre-training and Multi-tasking

As another future avenue, one could combine a form of pre-training with Multi-tasking. Under that setting, (possibly unsupervised) pre-training can be used to initialize the weights in a better way which can then be followed by a loop of multi-tasking for each task to benefit from the other tasks' features.

Chapter 8

Conclusions

Discourse Parsing and Sentiment Analysis are two fundamental NLP tasks that have been shown to be mutually beneficial. Evidence from previous work indicates that information extracted from Discourse Trees can help with Sentiment Analysis and likewise, knowing the sentiment of two pieces of text can help with identification of discourse relationships between them. In this thesis, we show how synergies between these two tasks can be exploited in a joint neural model. The first challenge entailed learning meaningful vector representations for text spans that are the inputs for the two tasks. Since the dimension of vanilla skip-thought vectors is too high compared to the size of our corpora, in order to simultaneously reduce the dimensionality and to produce vectors that are meaningful for our tasks, we devised task specific neural compressors, that take in Skip-thought vectors and produce much lower dimensional vectors.

Next, we designed three independent Recursive Neural Nets classifiers; one for Discourse Structure prediction, one for Discourse Relation prediction and one for Sentiment Analysis. After that, we explored two ways of creating joint models from these three networks: Pre-training and Multitasking. Our experimental results show that such models do capture synergies among the three tasks with the Multi-tasking approach being the most successful, confirming that latent Discourse features can help boost the performance of a neural sentiment analyzer and that latent Sentiment features can help with identifying contrastive relations between text spans.

In the short term, we plan to verify how syntactic information could be explicitly leveraged in the three task-specific networks as well as in the joint models. Then, our investigation will move from making predictions about a single sentence to the much more challenging task of dealing with multi-sentential text, which will likely require not only more complex models, but also models with scalable time performance in both learning and inference. Next, we intend to study how pre-training and multitasking could be both exploited simultaneously in the same model, something that to the best of our knowledge has not been tried before. Finally, as another venue for future research, we plan to explore how sentiment analysis and discourse parsing could be modeled jointly with text summarization, since these three tasks can arguably inform each other and therefore benefit from joint neural models similar to the ones described in this thesis.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org. → pages 26

[2] P. Bhatia, Y. Ji, and J. Eisenstein. Better document-level sentiment analysis from rst discourse parsing. In *Proceedings of Empirical Methods for Natural Language Processing (EMNLP)*, September 2015. URL http://www.aclweb.org/anthology/D/D15/D15-1263.pdf. → pages 4, 7, 27, 28, 30, 34

[3] L. Carlson and D. Marcu. Discourse tagging reference manual. *ISI Technical Report ISI-TR-545*, 54:56, 2001. → pages 14

[4] L. Carlson, M. E. Okurowski, and D. Marcu. *RST discourse treebank*. Linguistic Data Consortium, University of Pennsylvania, 2002. → pages 14

[5] Y.-A. Chung, H.-T. Lin, and S.-W. Yang. Cost-aware pre-training for multiclass cost-sensitive deep learning. *arXiv preprint arXiv:1511.09337*, 2015. → pages 9

[6] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008. → pages 5, 9, 22

[7] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010. → pages 5, 9, 23

[8] S. Faridani. Using canonical correlation analysis for generalized sentiment analysis, product recommendation and search. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 355–358. ACM, 2011. → pages 4

[9] V. W. Feng and G. Hirst. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 511–521, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-1048. → pages 6

[10] S. Gerani, Y. Mehdad, G. Carenini, R. T. Ng, and B. Nejat. Abstractive summarization of product reviews using discourse structure. In *EMNLP*, pages 1602–1613, 2014. → pages 1, 3

[11] F. Guzmán, S. R. Joty, L. Màrquez, and P. Nakov. Using discourse structure improves machine translation evaluation. In *ACL (1)*, pages 687–698, 2014. → pages 3

[12] K. Jiang, G. Carenini, and R. T. Ng. Training data enrichment for infrequent discourse relations. → pages 32, 33

[13] S. Joty, G. Carenini, and R. T. Ng. Codra: A novel discriminative framework for rhetorical analysis. *Computational Linguistics*, 2015. → pages 1, 6, 14, 21, 28, 31, 33

[14] T. Kaneko, K. Hiramatsu, and K. Kashino. Adaptive visual feedback generation for facial expression improvement with multi-task deep neural networks. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 327–331. ACM, 2016. → pages 9

[15] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015. → pages 5, 8, 17

[16] A. Lazaridou, I. Titov, and C. Sporleder. A bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations. In *ACL (1)*, pages 1630–1639, 2013. → pages 4

[17] J. Li, R. Li, and E. H. Hovy. Recursive deep models for discourse parsing. In *EMNLP*, pages 2061–2069, 2014. → pages 7, 8, 31

[18] P. Liu, X. Qiu, and X. Huang. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*, 2016. → pages viii, 5, 9, 11

[19] Y. Liu, S. Li, X. Zhang, and Z. Sui. Implicit discourse relation classification via multi-task neural networks. *arXiv preprint arXiv:1603.02776*, 2016. → pages viii, 9, 10, 12

[20] A. Louis, A. Joshi, and A. Nenkova. Discourse indicators for content selection in summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 147–156. Association for Computational Linguistics, 2010. → pages 3

[21] W. C. Mann and S. A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988. → pages 14, 15, 26, 27

[22] D. Marcu. *The theory and practice of discourse parsing and summarization*. MIT press, 2000. → pages 26

[23] D. Marcu and K. Knight. Discourse parsing and summarization, May 11 2001. US Patent App. 09/854,301. → pages 3

[24] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993. → pages 14

[25] T. Meyer and A. Popescu-Belis. Using sense-labeled discourse connectives for statistical machine translation. In *Proceedings of the Joint Workshop on Exploiting Synergies between Information Retrieval and Machine Translation (ESIRMT) and Hybrid Approaches to Machine Translation (HyTra)*, pages 129–138. Association for Computational Linguistics, 2012. → pages 3

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. → pages 8

[27] T. Nakagawa, K. Inui, and S. Kurohashi. Dependency tree-based sentiment classification using crfs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794. Association for Computational Linguistics, 2010. → pages 7

[28] B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008. → pages 7

[29] H. Peng, S. Thomson, and N. A. Smith. Deep multitask learning for semantic dependency parsing. *arXiv preprint arXiv:1704.06855*, 2017. → pages viii, 8, 12, 13, 33

[30] V. Rentoumi, S. Petrakis, M. Klenner, G. A. Vouros, and V. Karkaletsis. United we stand: Improving sentiment analysis by joining machine learning and rule based methods. In *LREC*, 2010. → pages 7

[31] S. Rosenthal, A. Ritter, P. Nakov, and V. Stoyanov. Semeval-2014 task 9: Sentiment analysis in twitter. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 73–80. Dublin, Ireland, 2014. → pages 1

[32] S. Z. Seyyedsalehi and S. A. Seyyedsalehi. A fast and efficient pre-training method based on layer-by-layer maximum discrimination for deep neural networks. *Neurocomputing*, 168:669–680, 2015. → pages 9

[33] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts, et al. Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, 1631:1642, 2013. → pages 7, 8, 9, 14, 19, 28, 30

[34] S. Verberne, L. Boves, N. Oostdijk, and P.-A. Coppen. Evaluating discourse-based answer extraction for why-question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 735–736. ACM, 2007. → pages 3