DISCOURSE PARSING AND ITS APPLICATION TO QUESTION GENERATION

by

Takshak Desai

APPROVED BY SUPERVISORY COMMITTEE:

Dr. Dan I. Moldovan, Chair

Dr. Latifur Khan

Dr. Sriraam Natarajan

Dr. Jessica Ouyang

To my mother, Purvi and my brother, Sakshar

DISCOURSE PARSING AND ITS APPLICATION TO QUESTION GENERATION

by

TAKSHAK DESAI, BS, MS

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN

COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

May 2021

ACKNOWLEDGMENTS

A sincere thank you to my advisor, Dr. Dan Moldovan, for his keen insight and guidance. I owe him the successful and timely completion of this dissertation. I extend my gratitude to committee members Dr. Latifur Khan, Dr. Sriraam Natarajan and Dr. Jessica Ouyang for their invaluable suggestions. I appreciate the crucial and relevant feedback provided by reviewers and area chairs of conferences and workshops where I submitted papers.

My deepest thanks to alumni and current members of my research group: Dr. Marta Tatu, Dr. Mithun Balakrishna, Dr. Tatiana Erekhinskaya, Dr. Linrui Zhang, and Parag Dakle for assisting me with my research. I also thank UT Dallas professors who taught courses that helped me comprehend fundamental concepts required to conduct research.

I am beyond grateful to my friends for being a constant source of motivation. I extend my thanks to my family, especially my mother, Purvi, and my brother, Sakshar. Words cannot describe how indispensable and inspiring your encouragement has been. I dedicate this dissertation to you and your support.

April 2021

DISCOURSE PARSING AND ITS APPLICATION TO QUESTION GENERATION

Takshak Desai, PhD
The University of Texas at Dallas, 2021

Supervising Professor: Dr. Dan I. Moldovan, Chair

Reading comprehension can be analyzed from three points of view: Semantics, Assessment, and Cognition. Here, Semantics refers to the task of identifying discourse relations in text. Assessment involves utilizing these relations to obtain meaningful question-answer pairs. Cognition means categorizing questions according to their difficulty or complexity levels. This dissertation addresses how to leverage or design natural language processing tools to perform underlying tasks and ultimately craft a reading comprehension quiz for use in a classroom environment.

Previous research has focused on mining shallow, sentence-level semantic relations and using them to craft intra-sentential, factoid questions. These are not very consequential in the context of large documents as they do not address how sentences coherently come together to comprise the full text. Discourse relations are capable of providing a comprehensive view of the text as they look beyond sentences. These relations focus on how sentences are logically and structurally linked to each other and provide a summarized, high-level overview of the document's semantics. Likewise, one can expect inter-sentential questions generated using discourse relations to be deep and inferential that can test comprehension abilities like analysis of the document's structure, identification of author's intent, and evaluation of stated arguments, among others. Testing these abilities allows one to assess student interpretation of a text effectively.

A deep multi-task learning framework is suggested that accurately deduces high-level discourse relations between text spans. The framework uses structure, syntax, and context-aware text representations that are robust enough to capture the document's meaning and intent. A set of syntactic transformations and well-formed transformation templates convert relations into question-answer pairs: the proposed model generates questions that are grammatically valid and intricate enough to gauge text comprehension. Then, a rich, feature-driven classifier categorizes these questions according to their difficulty levels. Results obtain empirically show that inter-sentential questions that test the ability to deduce high-level semantic relations in the text are more complex and meaningful than intra-sentential ones.

These modules are linked into a pipeline. The pipeline's performance is evaluated on benchmark corpora and it is shown that this pipeline can generate high-quality question-answer pairs that are more purposeful than human-designed ones and ones obtained from previously designed systems. By enhancing reading comprehension datasets with such questions, one can hope to advance research in question answering and reading comprehension.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

This chapter introduces readers to the problem statement, provides motivations for carrying out the research described in this document, summarizes original work proposed, and describes related theory pre-requisite to understanding the dissertation.

1.1 Problem Statement

Perform the following operations on a given a document D:

1. Induce a discourse tree $G = (V, E)$ from the text where V is the set of nodes representing textual arguments, and E refers to the set of edges that label relations between arguments. In the context of discourse analysis, G depicts high-level semantic relations, called discourse or coherence relations, that occur in the text and summarizes the meaning of the document. Consider this example that associates a small paragraph with its equivalent discourse tree:

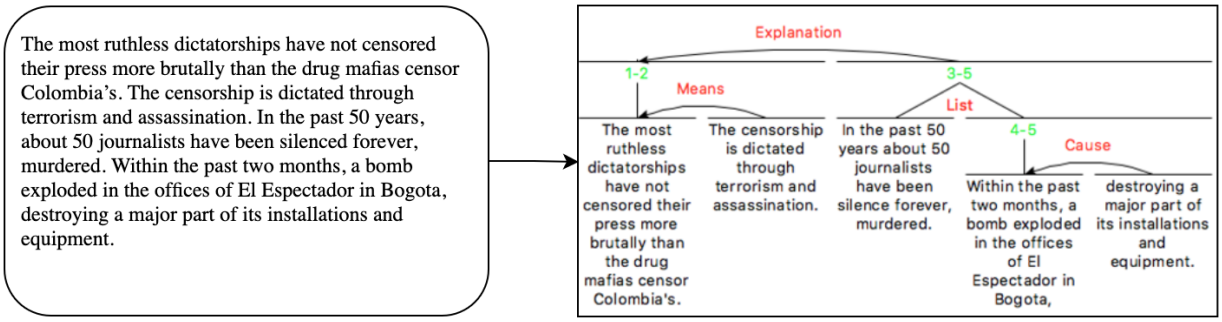

Figure 1.1. Associating text with discourse trees

Here, MEANS relation indicates the manner in which an event is carried out (in this example, press censorship). LIST relation is used to list out facts (here, a list of evidences supporting how the mafia has censored the press).

2. Translate the discourse tree G into question-answer pairs Q by applying syntactic transformations on the relations described by G. Consider this example that takes the discourse tree obtained from the previous step and transforms it into questions:

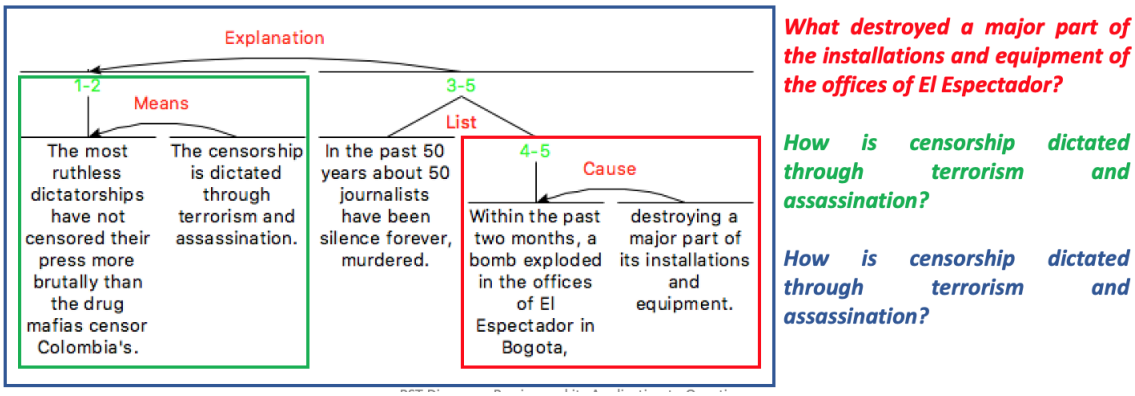

Figure 1.2. Transforming a discourse tree into questions

In this example, relations (as labelled by subtrees) are translated into questions by applying syntactic transformations on its arguments. Correspondence between relations and questions is shown via coloring.

3. Given a set of question-answer pairs Q associated with D, assign a difficulty label $c_k \in C$ where C is a set of difficulty levels to each pair $q_i \in Q$. Consider the example given below that associates obtained questions with a difficulty level between 1 and 3. The higher the rating, the more difficult the question is.

Figure 1.3. Associating questions with difficulty levels

Here, the first question, being trivial, is given a rating of 1 (the answer is indicated by the color green). A higher rating is given to the third question as it is inter-sentential and needs an understanding of the complete paragraph. One must provide a reasonably detailed response to answer such a question (answer indicated by the color orange).

An ultimate objective of performing these operations is to construct a reading comprehension assessment containing the text and associated questions ordered according to their difficulty. One can perform all processes successively, i.e., put together in a pipeline to create the quiz. Figure provides a pictorial view of this pipeline and its components. An article is inputted to this pipeline as shown. Modules comprising this pipeline are successively applied to the document to obtain a reading comprehension quiz.

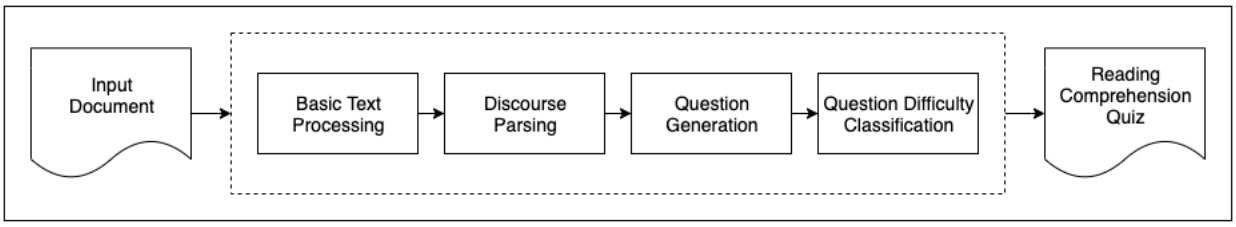

Figure 1.4. A pipeline and its modules for creating a reading comprehension quiz

Modules making up this pipeline are:

1. Basic Text Processing: This is the first module in the pipeline and collectively refers to tokenization and sentence boundary detection tasks. As text processing is relatively trivial, we do not address it in this dissertation. We include this module in the pipeline for the sake of completeness. We make use of the Stanford CoreNLP package (Manning et al., 2014) for carrying out text pre-processing

2. Discourse Parsing: The second module in the pipeline performs discourse parsing. It refers to the task of extracting high-level semantic relations between text spans in the document. We use the Rhetorical Structure Theory or RST (Mann and Thompson,

1988) as a framework for identifying such text-level relations. Specifically, this module has two parts:

   (a) Discourse Segmentation: Here, we identify arguments or text spans between which relations potentially hold. These are also called segments or Elementary Discourse Units (EDUs) and form nodes in the discourse tree.

   (b) Relation Classification: This refers to the broad task of inducing a discourse tree and mainly involves classifying relations between text spans. These relations are also called discourse, coherence, or rhetorical relations and label edges in the discourse tree.

3. Question Generation: The pipeline's third module performs question generation. We fashion a set of robust templates and well-defined syntactic transformations that translate relations present in the discourse tree obtained from the previous module into question-answer pairs.

4. Question Difficulty Classification: The pipeline's final module carries out question difficulty categorization. Question-answer pairs extracted from the previous step are associated with difficulty or complexity levels using a rich feature-driven classifier.

This pipeline's output is a reading comprehension quiz that comprises the document and associated questions ordered according to their difficulty levels.

1.2 Motivation

Reading comprehension refers to the ability to understand and process text, gauge its purpose, and identify the author's point of view. The fundamental skills required for reading comprehension include the facilities to:

1. Know and understand the meaning of a word from its context

2. Identify the main idea(s) presented by the author(s) by understanding their intentions

3. Follow the passage's organization and identify antecedents and references in it

4. Draw inferences from the text about its content, and so on.

A learning outcome of reading comprehension instruction is to help students develop knowledge, skills, and experiences to become competent and enthusiastic readers.

Simply put, a good reader must correctly deduce semantic relations from text and answer questions asked around them (Storey, 1993). A beginner would identify simple, intra-sentential relations; an advanced reader should infer more complex, inter-sentential ones. In other words, one can view reading comprehension from three related perspectives or lenses:

1. Semantics: This refers to identifying semantic relations that hold in text. By identifying these relations, one can get a deep understanding of the passage and the author's intent behind writing it.

2. Assessment: This refers to the task of answering questions asked around the document's semantics. One can use low-level semantic relations to craft simple questions and high-level ones to construct more profound and insightful questions.

3. Cognition: This means ranking the questions according to their difficulty or complexity. A beginner can quickly answer trivial questions, while advanced readers should respond to more difficult ones.

To motivate this idea, consider the reading comprehension assessment provided in Figure 1.5. It consists of the text and associated questions annotated with difficulty levels (between 1 and 3). The higher the difficulty level, the more difficult it is to answer.

Briefly, the passage begins by stating that Mobil Corporation cut down the size of its workforce. Then it provides additional details and background information about staff reductions. Later, it outlines the reasons behind job cuts; and possible outcomes of said cuts.

Passage:

Mobil Corp. is preparing to slash the size of its work force in the U.S., possibly as soon as next month, say individuals familiar with the company's strategy. The size of the cuts isn't known, but they'll be centered in the exploration and production division, which is responsible for locating oil reserves, drilling wells and pumping crude oil and natural gas. Employees haven't yet been notified. Sources said that [meetings to discuss the staff reductions have been scheduled for Friday at Mobil offices in New Orleans and Denver]$_1$.
Mobil's latest move could signal the beginning of further reductions by other oil companies in their domestic oil-producing operations. [In yesterday's third-quarter earnings report, the company alluded to a $40 million provision for restructuring costs involving U.S. exploration and production operations. The report says that the restructuring will take place over a two-year period and will principally involve the transfer and termination of employees in our U.S. operations. A company spokesman, reached at his home last night, would only say that there will be a public announcement of the reduction program by the end of the week.]$_4$ [Most oil companies, including Mobil, have been reporting lower third-quarter earnings, largely as a result of lower earnings from chemicals as well as refining and marketing businesses.]$_5$
Individuals familiar with Mobil's strategy say that [Mobil is reducing its U.S. work force because of declining U.S. output.]$_2$
[Yesterday, Mobil said domestic exploration and production operations had a $16 million loss in the third quarter, while comparable foreign operations earned $234 million.]$_5$ [Industry wide, oil production in this country fell by 500,000 barrels a day to 7.7 million barrels in the first eight months of this year. Daily output is expected to decline by at least another 500,000 barrels next year.]$_3$ Some Mobil executives were dismayed that a reference to the cutbacks was included in the earnings report before workers were notified.

Sample Questions:

No.	Question	Difficulty
1	Where have meetings to discuss staff reduction been scheduled?	1
2	Why is Mobil Corp. reducing its U.S. work force?	1
3	Evaluate the situation of oil production in the Unites States.	2
4	What structural changes is Mobil Corp. undergoing?	2
5	Why is Mobil Corp. reporting lower third-quarter earnings?	3

Figure 1.5. A representative example from the corpus: the passage, generated questions and their difficulty levels are shown.

As stated previously, this document can be viewed from different three points of view: Semantics, Assessment and Cognition. It can be noted that the three are closely related to each other.

A reader at the beginners' level would be able to detect sentence-level semantics. For example, in sentence 1 in Figure 1.5, beginners can immediately identify the LOCATION relation between "meetings to discuss the staff reductions have been scheduled" and "Mobile offices in New Orleans and Denver". Likewise, they can quickly infer a CAUSE relation between "Mobil is reducing its U.S. workforce" and "because of declining U.S. output" in sentence 2 owing to the presence of the keyword 'because'. This makes Questions 1 and 2 trivial.

A more advanced reader would identify inter-sentential relations like the ones shown in examples 3, 4 and 5. Such questions require a deduction of high-level coherence relations such as implicit CAUSE, EVALUATION, CIRCUMSTANCE, and the construction of a detailed response. Such questions are more challenging to answer than examples 1 and 2.

Previous research has primarily focused on sentence-level semantics (Moldovan and Blanco, 2012) and transforming simple assertions into questions (Heilman and Smith, 2010; Du et al., 2017). These are too shallow in the context of large texts as shown in Figure 1.5. It would be interesting to identify the more consequential discourse relations in a document and leverage them to craft questions that cater to different reading levels. Motivated by this idea, the rest of this dissertation focuses on how NLP tools can either be designed or leveraged to analyze reading comprehension from the three perspectives mentioned previously, i.e., Semantics, Assessment, and Cognition.

1.3 Summary of work done

Research work described in this document can be summarized thus:

1. The construction of text-level discourse parsers is considered a challenging problem for several reasons, such as the sparsity of training data, the inherent nature of discourse, complexity of relation classification, etc. This dissertation attempts to mitigate these

challenges by using a deep discourse segmenter and parser that accurately fashions discourse trees from the text.

(a) We propose a deep model that leverages BERT's (Devlin et al., 2019) complex structure for performing discourse segmentation. By jointly learning syntactic features and casting the problem as token classification (as opposed to the traditional way of sequence tagging), this dissertation advances the state-of-the-art and inches closer to human performance (the proposed segmenter achieves an F-score of 96.7 versus human performance of 98.3)

(b) We design a deep model leveraging the representational powers of BERT and Hierarchical Attention Networks or HANs (Yang et al., 2016) to obtain syntax, structure, and context-aware text representations. These representations are then fed to a shift-reduce parser to carry out accurate text-level discourse parsing. Evaluation on benchmark datasets shows that the model rivals the performance of state-of-the-art feature-driven and neural parsers. Additionally, the use of HANs allows one to get insights into the model's behaviour and understand errors made during classification.

2. Modern automated question generation systems either generate Wh-type questions (Heilman and Smith, 2010), use shallow semantics and ontologies for developing questions (Graesser et al., 2003; Araki et al., 2016; Stasaski and Hearst, 2017) or use deep learning for generating questions from one to few sentences (Du et al., 2017). This dissertation suggests a method for generating inferential, inter-sentential questions that test the ability to deduce high-level relations between text spans.

3. This dissertation defines what it means for a question to be 'meaningful' by describing a set of metrics that measure its quality. In sharp contrast to measures like grammatical

and semantic correctness, these metrics help gauge the quality and complexity of a question.

4. A rich semantic feature-driven classifier is described that categorizes questions according to their difficulty or complexity. Using these features allows one to model information that cannot be captured by simple baselines, thereby observing a relative improvement of 26% in model performance.

5. These modules are arranged into a pipeline and applied in succession to a benchmark reading comprehension corpus: SQuAD (Rajpurkar et al., 2016). We show that this pipeline can generate better quality questions that are more complex than those already present in the corpus. One can hope to advance research in question answering by analyzing such deep, inferential questions.

1.4 Related Work

The dissertation is motivated by two sources of information: The Rhetorical Structure Theory or RST (Mann and Thompson, 1988), a framework for analyzing document-level relations, and Bloom's taxonomy (Bloom, 1964), a cognitive framework for relating student understanding to educational objectives and learning outcomes. This section highlights the pre-requisite theory behind RST and Bloom's taxonomy required to comprehend the rest of this dissertation. Subsequent chapters in this dissertation describe computational models and frameworks for performing discourse analysis and question generation.

1.4.1 Rhetorical Structure Theory

Documents are not a random collection of text spans but rather a structured list of textual components forming a discourse. Discourse structures describe their organization in terms of coherence or rhetorical relations. Mann and Thompson (1988) presented Rhetorical

Structure Theory (RST), a framework that defines the functional organization of textual components in the text. RST describes relations between text parts called segments, identifying the relation's transition point and the extent of the items related. It identifies a hierarchical structure in the text, allowing one to get a high-level view of semantics at the document-level.

While several discourse representation theories have been proposed over the years (Asher et al., 2003; Prasad et al., 2008); reasons for choosing RST over these include:

1. From a linguistic point-of-view, RST provides a robust, hierarchical view of text organization.

2. It points to a tight relation between relations and coherence in text

3. From a computational point of view, it provides a characterization of text relations that has been implemented in different systems and for applications as text generation and summarization.

A typical RST-style discourse parser operates in two stages:

1. Discourse Segmentation: This refers to the task of fragmenting a document into multiple disjoint chunks of text called elementary discourse units or EDUs. In the context of RST, these form the leaves of the discourse tree between which relations hold.

2. Relation Classification: Once the EDUs have been identified, a tree structure is induced where directed arcs between EDUs and/or subtrees represent relations.

An example of a discourse tree is provided in Figure 1.6. It can be seen that nodes represent EDUs or arguments between which relations hold and edges indicate relations.

One can also note that this tree is hierarchical, and the granularity of a relation reveals the type of information it conveys. For example, the CAUSE relation between segments 4

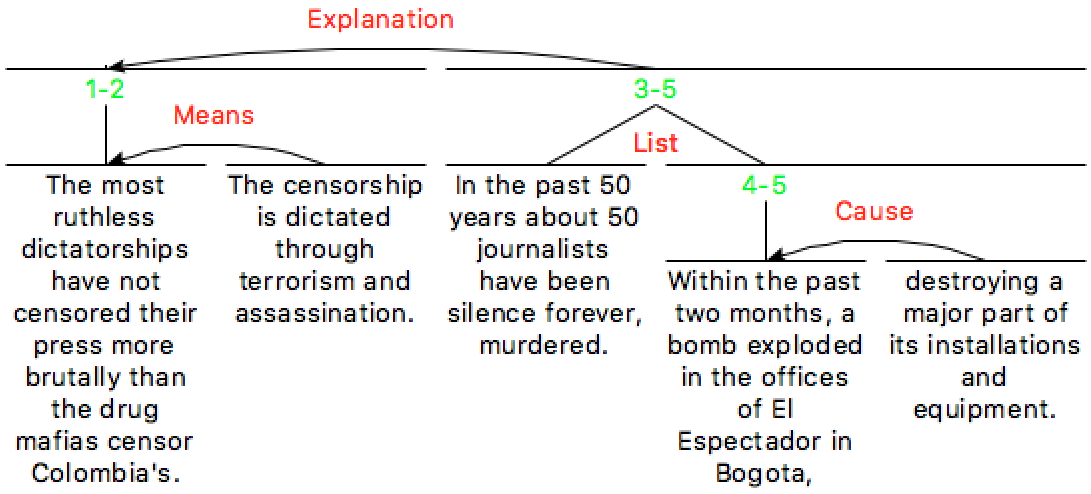

Figure 1.6. A representative example showing how a discourse tree is defined by the RST

and 5 in this tree is a trivial intra-sentential relation. Such a relation is not very meaningful in the context of the entire document. However, it does summarize information conveyed by the sentence. Likewise, the MEANS relation between segments 1 and 2 is an inter-sentential relation. This relation is more meaningful than the CAUSE relation described previously. The EXPLANATION relation labels the root of the tree and the text's main idea. This relation is more high-level than all other relations described by the tree.

Typically, RST defines 4 objects to describe the construction of the discourse tree. Subsequent subsections provide definitions and examples of each object:

Relations

Relations are defined between two disjoint contiguous chunks of text. Formally, a relation has three attributes: R, N and S:

- R refers to the name of the relation that holds between text spans. For example, some relations present in Figure 1.6 are CAUSE, MEANS, LIST etc.

- N is the nucleus. The nucleus is a piece of text that is more central to the author's point of view and can be interpreted independently.

- S means satellite. The satellite is a chunk of text less central to the author's intentions and can only be interpreted with respect to the nucleus.

For example, in Figure 1.5, the CAUSE relation has 'Within the past two months, a bomb exploded in the offices of El Espectador in Bogota,' as its Nucleus and 'destroying a major part of its installations and equipment' as its Satellite.

Note that relations may be of two types: asymmetric relations that hold between a nucleus and a satellite (consider examples EXPLANATION, MEANS and CAUSE in Figure 1.6); and symmetric relations that hold between multiple nuclei (for instance, in the LIST relation in Figure 1.6 a series of nuclei is given without contrast or explicit comparison).

RST defines more than 70 relations in its inventory. However, only few (research focuses on 16 in particular) are most commonly studied or referred. A complete list of these relations along with their definitions and examples can be found in Carlson and Marcu (2001). The most commonly studied relations are described in Table 1.4.1.

Table 1.1. Description of most common relations found in RST. Relations annotated with an asterisk (*) are always multi-nuclear. Definitions have been taken from Carlson and Marcu (2001)

Relation	**Definition**	**Example**
ATTRIBUTION	Used to mark instances of reported speech, both direct and indirect. The satellite is the source of the attribution, for example, a clause containing a reporting verb, or a phrase beginning with 'according to', and the nucleus is the content of the reported message which must be in a separate clause.	Nucleus: The legendary GM chairman **declared** Satellite: that his company would make "a car for every purse and purpose."
BACKGROUND	In this relation, the satellite establishes the context or the grounds with respect to which the nucleus is to be interpreted. Understanding the satellite helps the reader understand the nucleus.	Nucleus: Banco Exterior was created in 1929 to provide subsidized credits for Spanish exports. Satellite: The market for export financing was liberalized in the mid-1980s, however, forcing the bank to face competition.
CAUSE	The situation presented in the nucleus is the cause of the situation presented in the satellite. The cause, which is the nucleus, is the most important part. The satellite represents the result of the action.	Nucleus: This year, a commission appointed by the mayor to revise New York's system of government completed a new charter, Satellite: expanding the City Council to 51 from 35 members.
COMPARISON*	Here, two textual spans are compared along some dimension. Some examples are similar, different, greater-than, less-than, etc.	Nucleus: It said it expects full-year net of 16 billion yen, Nucleus: compared with 15 billion yen in the latest year.
CONDITION	Here, the truth of the proposition associated with the nucleus is a consequence of the fulfillment of the condition in the satellite. The satellite presents a situation that is not realized.	Nucleus: S.A. brewing would make a takeover offer for all of Bell Resources Satellite: if it exercises the option, according to the commission

Relation	Definition	Example
CONTRAST*	Here, two or more nuclei come in contrast with each other along some dimension. The contrast may happen in only one or few respects, while everything else can remain the same in other respects. Typically, a CONTRAST relation includes a contrastive discourse cue, such as 'but', 'however', 'while', whereas a COMPARISON does not.	Nucleus: The proposal reiterates the U.S. desire to scrap or reduce a host of trade-distorting subsidies on farm products. Nucleus: But it would allow considerable flexibility in determining how and when these goals would be achieved.
ELABORATION	In this relation, the satellite gives additional information or detail about the situation presented in the nucleus. This relation is extremely common at all levels of the discourse structure, and is especially popular to show relations across large spans.	Nucleus: Under a proposal by Democrats to expand Individual Retirement Accounts, a $2,000 contribution by a taxpayer in the 33% bracket would save $330 on his taxes. Nucleus: The savings was given incorrectly in Friday's edition.
ENABLEMENT	In this relation, the situation presented in the nucleus is unrealized. The action presented in the satellite increases the chances of the situation in the nucleus being realized.	Nucleus: The administration of federal credit should closely parallel private lending practices, including the development of a loan loss reserve. Nucleus: Establishing these practices would permit earlier identification of emerging financial crises and provide better information for loan sales and budgeting decisions.
EVALUATION	In an EVALUATION relation, one span assesses the situation presented in the other span of the relationship on a scale of good to bad. An evaluation can be an appraisal, estimation, rating, interpretation, or assessment of a situation.	Nucleus: What defeated General Aoun was not only the weight of the Syrian army. The weight of Lebanon's history was also against him; Nucleus: and it is a history Israel is in danger of repeating.

Relation	Definition	Example
EXPLANATION	An EXPLANATION relation usually pertain to actions and situations that are independent of the will of an animate agent.The satellite provides a factual explanation or empirical evidence for the situation presented in the nucleus.	Nucleus: That system has worked. Satellite: The standard of living has increased steadily over the past 40 years; more than 90% of the people consider themselves middle class.
MANNER-MEANS	A means satellite specifies a method, mechanism, instrument, channel or conduit for accomplishing some goal. It should tell one how something was or is to be accomplished.	Nucleus: Some underwriters have been pressing for years to tap the low-margin business Satellite: by selling some policies directly to consumers.
SUMMARY	In a SUMMARY relation, either the nucleus or satellite summarizes the information presented in the satellite or nucleus respectively. The size of the summary is shorter than the size of the text it is summarizing.	Nucleus(summary): The Singapore and Kuala Lumpur stock exchanges are bracing for a turbulent separation, following Malaysian Finance Minister Daim-Zainuddins long-awaited announcement that the exchanges will sever ties. Satellite: On Friday, Datuk Daim added spice to an otherwise unremarkable address on Malaysias proposed budget for 1990 by ordering the Kuala Lumpur Stock Exchange to take appropriate action immediately to cut its links with the Stock Exchange of Singapore. The delisting of Malaysian-based companies from the Singapore exchange may not be a smooth process, analysts say. Though the split has long been expected, the exchanges are not fully prepared to go their separate ways.

Relation	Definition	Example
SAME-UNIT*	A pseudo-relation used as a device for linking two discontinuous text fragments that are really a single EDU, but which are broken up by an embedded unit.	Nucleus: The yens softness Nucleus: apparently stems from Japanese investors interest Nucleus: in buying dollars against the yen to purchase U.S. bond issues.
TEMPORAL	In this relation, the situation present in the nucleus occurs either before, during the same time or after that presented in the satellite.	Nucleus (occurs temporally after satellite): RJR Nabisco Inc. is disbanding its division responsible for buying network advertising time, Satellite: just a month after moving 11 of the groups 14 employees to New York from Atlanta.
TEXTUAL-ORGANIZATION*	This is a multinuclear relation used to link elements of the structure of the text, for example, to link a title with the body of the text, a section title with the text of a section, etc.	Nucleus (date): Friday, October 13, 1989 Nucleus (text): The key U.S. and foreign annual interest rates ...
TOPIC-COMMENT*	A general statement or topic of discussion is introduced, after which a specific remark is made on the statement or topic. This relation is always multinuclear, as both spans are necessary to understand the context.	Nucleus: As far as the pound goes, Nucleus: some traders say a slide toward support at $1.5500 may be a favorable development for the dollar this week. ...
TOPIC-SHIFT	The relation TOPIC-SHIFT is used to link large textual spans when there is a sharp change in focus going from one segment to the other. The same elements are not in focus in the two spans.	Nucleus: South Africa freed the ANCs Sisulu and seven other political prisoners. Satellite: The Soviet Union reported that thousands of goods needed to ease widespread shortages across the nation were piled up at ports and rail depots. The delay in sending these goods to the citizens has been attributed to harsh winters.

Relation	Definition	Example
EVIDENCE	Here, the situation presented in the satellite provides evidence or justification for the situation presented in the nucleus. Evidence is data on which judgment of a conclusion may be based, and is presented by the writer or an agent in the article to convince the reader of a point. An evidence satellite increases the chance of the reader accepting the information presented in the nucleus.	Nucleus: That system has worked. Satellite: The standard of living has increased steadily over the past 40 years; more than 90% of the people consider themselves middle class.
LIST*	This is a multinuclear relation whose elements can be listed, but which are not in a comparison, contrast or other, stronger type of multinuclear relation. A LIST relation usually exhibits some sort of parallel structure between the units involved in the relation. At lower levels of the discourse structure, such as between clauses or sentences, a LIST relation is often selected when there is some sort of parallel syntactic or semantic structure between the units, such as in the examples below. At higher levels of the discourse structure, the relation may be found when there are paragraphs of items enumerated in a similar fashion.	Nucleus: A union, sooner or later, has to have an adversary, Nucleus: and it has to have a victory.

Schemas

Schemas define how text spans and relations structure the discourse tree. As an abstract data type, a schema has three fields: (1) text spans that make up the leaves and subtrees of the

discourse tree, (2) definition and nuclearity of the relation(s) that hold between spans; and (3) a specification of how the nuclei are related to the whole collection, i.e., the represented document. The RST recognizes four schema types as depicted in Figure 1.7 that show how text constituents make up a tree.

The first figure provides a simple example of the mono-nuclear CAUSE relation. The relation is directed from nucleus to satellite, i.e., from A to B. The second figure provides an example of two relations, i.e., EVALUATION and MOTIVATION creating a subtree rooted at the node labeled as 6. The third and fourth diagrams give an example of the multi-nuclear JOINT and SEQUENCE relations that are similar in structure to the LIST relation. Note that these relations can hold between more than two nuclei (as indicated in the 4th diagram). To simplify the view of such a tree, one can transform this into a right-branching binary tree. In other words, the relation SEQUENCE(H, I, J) can be interpreted as SEQUENCE(I,J) and SEQUENCE(H, I:J) where I:J represents the text span containing segments I and J.

It is important to note that unlike constituency parsing that defines clear production rules on how one can represent the grammar or the syntax of a language, RST is more abstract. Therefore, the schema types diagrammed in Figure 1.7 are only loosely analogous to constituency grammars and cannot be treated as production rules to carry out deterministic parsing (Li et al., 2014).

Schema Applications

Schema application refers to how spans come together to constitute the discourse tree. Three conventions determine the possible applications of schemas:

1. **Unordered spans**: Schema application place no constraints on the nucleus and satellite arrangement in the text to which the schema is applied. As an example, consider the LIST relation. Strictly from a structural perspective, the order in which nuclei of the LIST relation occur is not important. From a semantic perspective, one may want

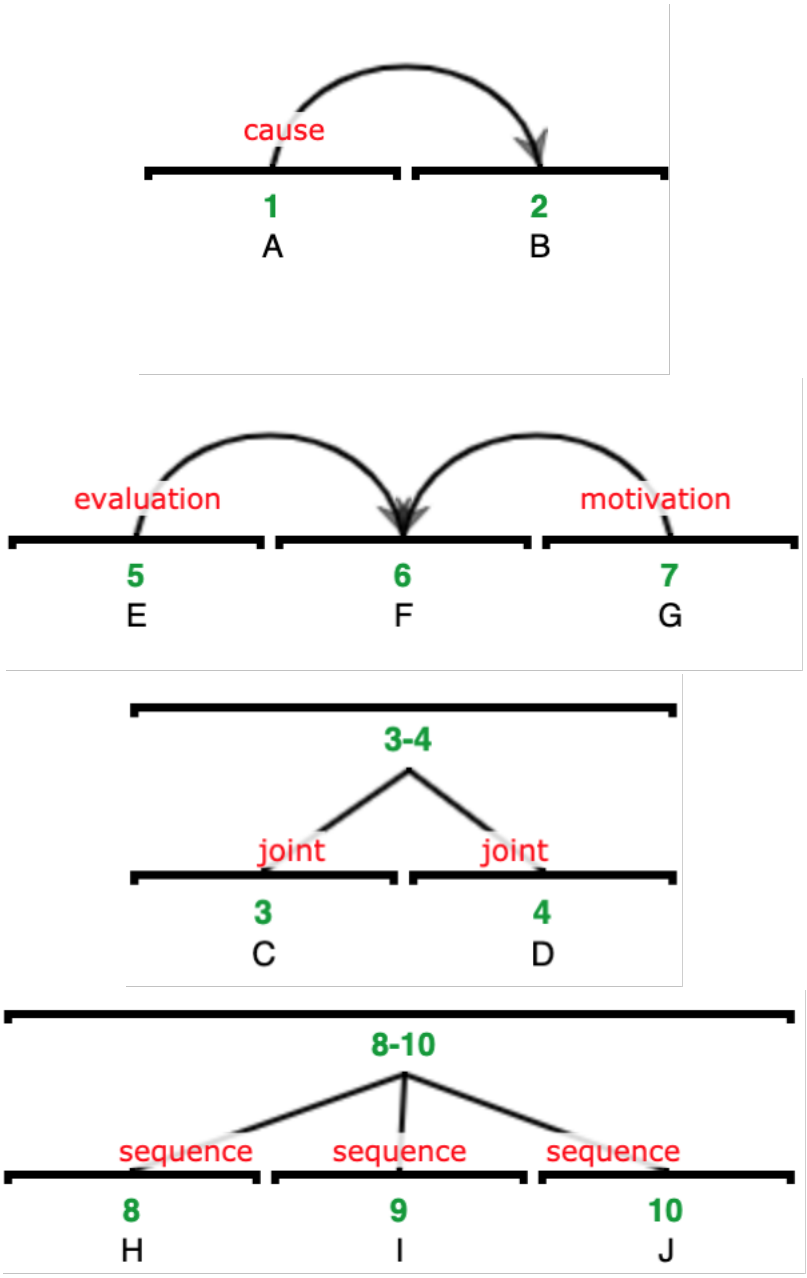

Figure 1.7. Diagrams showing schema types and how they can be used to construct a RST tree. Letters A-J are placeholders for actual text

the most important list item to be at the beginning. However, schema applications only care about the structure of the tree and not the underlying semantics.

2. **Optional relations**: For schema involving many relations, all individual relations are optional, but at least one must hold. For instance, in Figure 1.6, 4 out of the many possible relations defined by RST hold. Other relation instances cannot be found in this tree.

3. **Repeated relations**: A relation can be applied any number of times in a schema. For instance, the ELABORATION relation is fairly common and is often found many times in the same discourse tree used to represent fairly large texts. Likewise, a relation may also occur at multiple levels of tree granularity within the same discourse tree.

Structure

Similar to schema applications, following constraints hold with respect to the structural composition of the discourse tree:

1. **Completeness**: All text spans present in the document must be covered by the schema. As shown in Figure 1.5, every text span is related to at least one other text span via a relation.

2. **Connectedness**: Except for the root node of the tree i.e. the node representing the text in its entirety, every text span must be connected to at least one other text span.

3. **Uniqueness**: Every document must be represented by exactly 1 discourse tree. In other words, more than 2 relations cannot hold between text spans. In the likely event that is true, the more informative relation is selected. For example, consider this text:

   Nucleus: The project under construction will raise Las Vegas' supply of rooms by 20%.

   Satellite: Clark county will have 18000 new jobs.

Here, a CAUSE relation holds between the nucleus and satellite. Likewise, a TEMPORAL relation also holds between the two as Satellite will occur after Nucleus takes place. However, the CAUSE relation takes precedence as it is more informative than TEMPORAL.

4. **Adjacency**: A left-to-right traversal of the tree yields the entire text as it appears in the document. In other words, there is a direct edge between two adjacent text spans while non-adjacent text spans are indirectly connected via a series of edges.

1.4.2 Bloom's Taxonomy

The argument for a strong correlation between question difficulty and student perception comes from Bloom's taxonomy (Bloom, 1964). It is a framework that attempts to categorize question difficulty by educational goals. The framework has undergone several revisions over time and currently has six levels of perception in the cognitive domain: Remembering, Understanding, Applying, Analyzing, Evaluating, and Creating (Anderson et al., 2001).

To get a deeper understanding of the goals laid out by the taxonomy, consider the description and examples provided in Table 1.2. While the lower levels of this taxonomy cater to beginners, the higher levels cater to advanced readers. One may assume beginners to be able to infer simple sentence-level semantic relations like AGENT, explicit CAUSE and LOCATION and thus answer questions that cater to Levels 1 or 2 of the taxonomy. On the other hand, intermediate to advanced readers should infer more complex relations such as EVALUATION, SOLUTIONHOOD and so on; and thus answer questions that cater to Levels 3 or up in the taxonomy.

1.5 Dissertation Organization

This section outlines how the rest of this dissertation is organized and provides a motivation for including each chapter.

Table 1.2. A brief description and examples of questions as per Bloom's taxonomy. Definitions are taken from Anderson et al. (2001)

Level	Description	Example
Knowledge	Involves recognizing or remembering facts, terms, basic concepts, or answers without necessarily understanding what they mean	Name different types of access specifiers in C++.
Comprehension	Involves demonstrating an understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions, and stating the main ideas	Differentiate between constructors and destructors.
Application	Involves using acquired knowledge-solving problems in new situations by applying acquired knowledge, facts, techniques and rules	Predict the output of the given C++ code.
Analysis	Involves examining and breaking information into component parts, determining how the parts relate to one another, identifying motives or causes, making inferences, and finding evidence to support generalizations	Write a correctness proof for the given algorithm.
Synthesis	Involves building a structure or pattern from diverse elements; it also refers to the act of putting parts together to form a whole.	Complete the given incomplete C++ code snippet.
Evaluation	Involves presenting and defending opinions by making judgments about information, the validity of ideas, or quality of work based on a set of criteria.	Which of these algorithms is the most efficient and why?

First, we provide a brief and directed review of deep learning models for natural language processing in Chapter 2. Deep learning has been widely used in many language tasks (Collobert et al., 2011) and has given state-of-the-art results on almost all of them. The chapter provides explicit details on models and paradigms used for building the discourse parser.

Reading about these models will give one pre-requisite information required to understand the parser's functionality in depth.

Next, this dissertation dives into a study of the discourse parser. We divide the discussion on the parser's design into two chapters: the discourse segmenter (argument detection) in Chapter 3; and the relation classifier (discourse tree induction) in Chapter 4. We provide specific details on challenges associated with designing each module and how deep learning can mitigate these issues. We also provide information on how the modules benefited from a particular feature or model enhancement.

Post discourse parsing, Chapter 5 describes the vital syntactic transformations and templates designed to convert relation triples into question-answer pairs. These templates are robust enough to generate valid and meaningful questions. Additionally, this chapter also defines what it means for a question to be 'meaningful' and describes metrics for assessing its quality.

Then, Chapter 6 describes how to categorize questions according to their difficulty level. We outline typical challenges associated with question difficulty classification. When we enrich the baseline model with distinct semantic features, it can accurately ascribe difficulty levels to questions.

Chapter 7 provides a thorough assessment of the pipeline on the SQuAD dataset (Rajpurkar et al., 2016) and enriches the questions already present in the corpus with inter-sentential, inferential ones. We provide statistics on what fraction of relations extracted are implicit (hard to identify) and what fraction of associated questions are easy or difficult in complexity.

Finally, Chapter 8 wraps up this dissertation by providing conclusions and suggesting potential avenues for further research.

CHAPTER 2

REVIEW OF DEEP LEARNING

The field of Natural Language Processing has evolved from creating rule-based systems to developing shallow models trained on high-dimensional, sparse data to using complex deep learning architectures for tackling language problems. Deep learning, in particular, has made impressive advances in this field by achieving state-of-the-art results on many disparate tasks such as Sentiment Analysis, Question Answering, Dialogue Systems, Text Summarization, among others. One can attribute the success of deep learning to its ability to automatically identifying multi-level features from text. These features are more expressive than hand-crafted ones. While an extraordinarily detailed and comprehensive summary of recent trends in deep learning for Natural Language Processing can be found in Young et al. (2018), this section summarizes only those models and approaches used in this Dissertation. Readers familiar with deep learning can skip this chapter and proceed to the next one.

Specifically, this chapter provides details about two neural network architectures: BERT (Devlin et al., 2019), a powerful language model that supplies deep contextualized representations, and HAN (Yang et al., 2016), a network used to model context and structure in long inputs such as documents. It also familiarizes readers with multi-task learning (Caruana, 1997): a popular mechanism for learning multiple objective functions (in this case, language tasks) simultaneously.

2.1 Need for Deep Learning

The inherent nature of natural language makes it difficult to represent it for computational approaches. In particular, natural language possesses these properties that make it challenging to broach language tasks (Goldberg, 2016):

1. Language is symbolic and discrete. The most fundamental elements of language are characters that, in turn, form words. These elements may represent concepts, events,

or ideas. Both characters and words are discrete symbols. But, words such as "leaf" and "tree" are treated as a strings (sequences of characters) and not as pockets of information. Subsequently, an inherent relation between these concepts cannot be inferred by looking at these strings alone. One must consult a dictionary to identify some form of association between the meanings of these words.

2. Language is **compositional**. Letters form words; words form phrases and sentences, which in turn form paragraphs and documents. Interpretation of text requires one to look beyond the meaning of its constituent words. The meaning of a document can be larger than the meaning of the individual sentences that comprise it. Interpreting these documents requires one to follow a set of intricate rules that govern the logical connection between language elements.

3. There are practically **infinite** ways in which sentences can be formed from words. Therefore, the number of grammatically and semantically correct sentences is also practically infinite. One cannot possibly **enumerate** all such sentences. Additionally, there is no clear way of identifying a relationship between two sentences, for instance, defining how similar in meaning two sentences are to each other. This poses a challenge when one uses supervised machine learning for language tasks: even with a vast amount of training data, it is likely that one may encounter an example in the test data that was not present in the training data.

Shallow machine learning models like Logistic Regression and Support Vector Machines could yield decent results on some language tasks. However, even simple feed-forward neural networks were able to outperform these models (Collobert et al., 2011). This difference in performance is due to their ability to correctly represent inputs and effectively learn the desired input-output mapping. This makes them more effective than traditional machine

learning methods that rely on manually crafted features and sparse, high-dimensional input representations.

As with any machine learning algorithm, a deep learning model also takes in some feature vector as input, performs a series of successive transformations on this input, and predicts the output. Concerning natural language, the feature vector encodes linguistic information such as words, part-of-speech tags, parse-tree features, and so on. However, unlike traditional machine learning models, deep learning models do not one-hot encode the features. Instead, they represent these features as dense vectors, i.e., each feature is embedded into a d-dimensional space and represented as a vector in that feature space. The specific advantage offered by dense representations over one-hot representations include:

1. Most neural networks do not work well with very high-dimensional, sparse representations. Using low-dimensional, dense features offers computational benefits.

2. Dense features can generalize better than sparse ones. One expects similar words to have similar representations (McDonald and Ramscar, 2001) in a dense feature space. So, for instance, one may expect the learned vector for the concept 'leaf' to be similar to the learned vector for the concept 'tree' instead of the learned vector for 'hamburger'.

Bengio et al. (2003) proposed a simple neural language model that learned vector representations or embeddings for words: construction of word embeddings was inspired by the distributional hypothesis. In simple words, this hypothesis states that semantically similar words tend to occur in similar linguistic contexts. Learned embeddings were then concatenated to form an embedding for the entire sentence: this would help generalize better for unseen sentences. Likewise, Collobert and Weston (2008) designed a neural network that used pre-trained word embeddings for learning representations; and popularized their use for carrying out NLP tasks.

The use of word embeddings was revolutionized by Mikolov et al. (2013); Pennington et al. (2014) who came up with word2vec and GloVe, respectively: two approaches that became immensely popular for obtaining good-quality embeddings. Additionally, using pre-trained embeddings obtained from word2vec or GloVe in a transfer learning setup became ubiquitous with learning almost all language tasks.

2.2 BERT

2.2.1 Need for Contextualized embeddings

The quality of word embeddings is determined by their ability to represent syntactic and semantic information effectively. With word2vec and GloVe, every word is associated with one embedding representation, regardless of the context in which it occurs. This particularly poses a problem for polysemous words where one word may be associated with multiple word senses. The responsibility of learning context lies on the deep model used. For instance, consider the following sentences that contain the word 'tree':

(1) An AVL ***tree*** *is a height-balanced binary search tree.*

(2) The apple does not fall from the ***tree****.*

(3) Mr. ***Tree*** *owns a farm in Hershey, PA.*

In each of these sentences, the word 'tree' has a different meaning or word sense. It would not be wise to represent 'tree' with the same embedding vector, as given by traditional word embedding approaches like word2vec or GloVe. This calls for the need to leverage context while obtaining word embeddings as these would lead to improved semantic representations. Among the several approaches and architectures that provide such deep contextualized representations, popular examples include ELMo (Peters et al., 2018), OpenAI-GPT (Radford et al., 2018), BERT (Devlin et al., 2019), etc.

Bidirectional Encoder Representations from Transformers (BERT), in particular, became quite popular, due to its ability to provide powerful sentence representations that captured almost all syntactic, semantic and positional dependencies between its constituent words. Fine-tuned BERT models were able to outperform existing models by large margins, on key language tasks such as Question-Answering, Named Entity Recognition, Natural Language Inference, etc.

BERT was designed to train deep bidirectional representations using the Transformer model (Vaswani et al., 2017) as its basic unit of repetition in a 12-layer architecture. The Transformer itself consists of a simple feed-forward layer network and a unit of multi-head self-attention.

2.2.2 Pre-training BERT

BERT is pre-trained using two unsupervised tasks: masked language model (LM) and next sentence prediction (NSP).

To obtain a bidirectional representation of the sentence, the masked language model begins by randomly masking some percentage of the input tokens at random; and then predicting these masked tokens. The idea is similar to the cloze task in literature where one must "fill in the blanks" with the correct word or token.

Language modeling does not capture the relation between two sentences: to train a model that understands relations between sentences, BERT pre-trains for a simple binarized sentence-prediction task which can be trivially generated from any corpus. In practice, two sentences A and B are chosen from the corpus. 50% of the time, B follows A and 50% of the time, B is chosen at random from the corpus.

BERT is pre-trained on a very large corpus containing more than 3,000 million words. This makes it suitable for fine-tuning to any language task which may have very little training data to work with.

2.2.3 Fine-tuning BERT

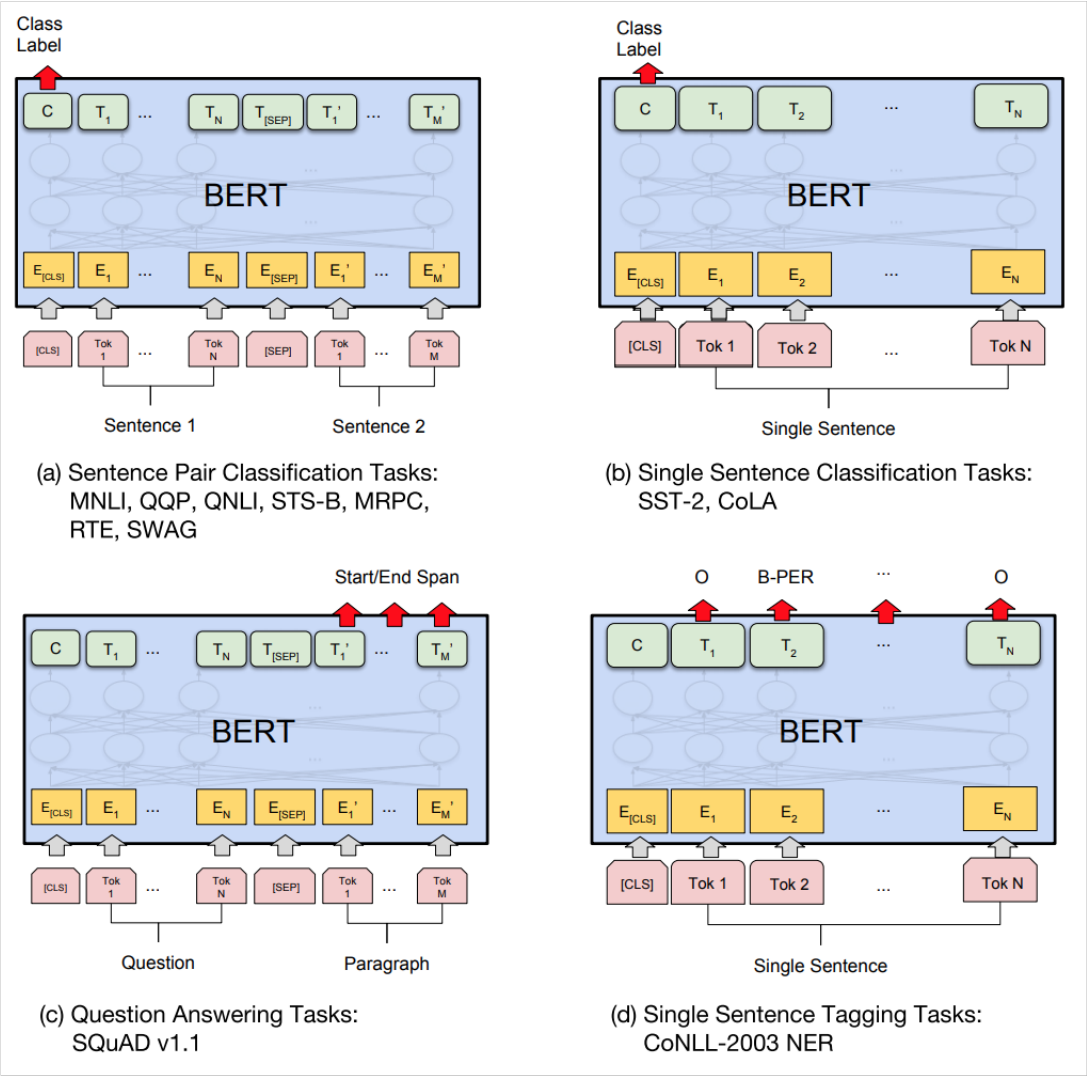

Figure 2.1. Typical use-cases for BERT

Fine-tuning is relatively straightforward since the self-attention mechanism in the Transformer model allows BERT to model many downstream tasks, whether they involve single text or text pairs, by swapping out the appropriate inputs and outputs. For each task, one must simply plug in the task-specific inputs and outputs into BERT and fine-tune all

the parameters end-to-end. At the input, sentence A and sentence B from pre-training are analogous to (1) sentence pairs in paraphrasing, (2) hypothesis-premise pairs in entailment, (3) question-passage pairs in question answering, and (4) a degenerate text-ϕ pair in text classification or sequence tagging. At the output, the token representations are fed into an output layer for token-level tasks, such as sequence tagging or question answering, and the [CLS] representation is fed into an output layer for classification, such as entailment or sentiment analysis.

Figure 2.1 provides an illustration of how BERT can be used for different language tasks[1].

2.3 Hierarchical Attention Networks

To address the issue of compositionality in language, Yang et al. (2016) introduced Hierarchical Attention Networks or HANs for classifying documents (not sentences). HANs offer two specific advantages over other neural network architectures:

1. HANs model the structure or layout of a document by attending to multiple levels of the text hierarchy. In other words, they first focus on words, which make up sentences and then on sentences that comprise the full text.

2. HANs can identify which words and sentences are more consequential in contributing to the meaning of the text by weighing them differentially (according to their importance). This allows them to specifically attend to those parts of the document that contribute more significantly to the underlying language tasks while disregarding the rest.

To understand why HANs are useful in the context of document understanding, consider the example given below. This example is taken from the IMDb Sentiment Review dataset

[1]Figure taken from Devlin et al. (2019)

(Maas et al., 2011) where multi-sentence movie reviews are associated with a positive or negative sentiment.

If you like adult comedy cartoons, like South Park, then this is nearly a similar format about the small adventures of three teenage girls at Bromwell High. Keisha, Natella and Latrina have given exploding sweets and behaved like bitches. I think Keisha is a good leader. There are small stories going on with the teachers of the school. There's the idiotic principal, Mr. Bip, the nervous Maths teacher and many others. The cast is also ***fantastic****. I didn't know this came from Canada, but it is very* ***good****. Very* ***good****!*

In this fairly long review which has a positive sentiment associated with the text, not all sentences and/or words are central to the classification problem. Words that are bolded are clearly more important as compared to others. Likewise, sentences that are colored in red are more meaningful in identifying the sentiment of this review, compared to other sentences. HANs are able to identify such words and/or sentences and give them more importance than other components making up the text. Figure 2.2 depicts the architecture of a HAN network[2].

Specifically the HAN model contains 4 important components that have been described below. For understanding the functionality of these components, consider the following set of notations:

Define a document T as a large chunk of text that consists of multiple sentences. Assume T has m sentences and each sentence s_j contains n words. Word w_{ij} represents the ith word in the jth sentence where $i \in [1, n]$ and $j \in [1, m]$. The HAN encoder constructs representations

[2]Figure taken from Yang et al. (2016)

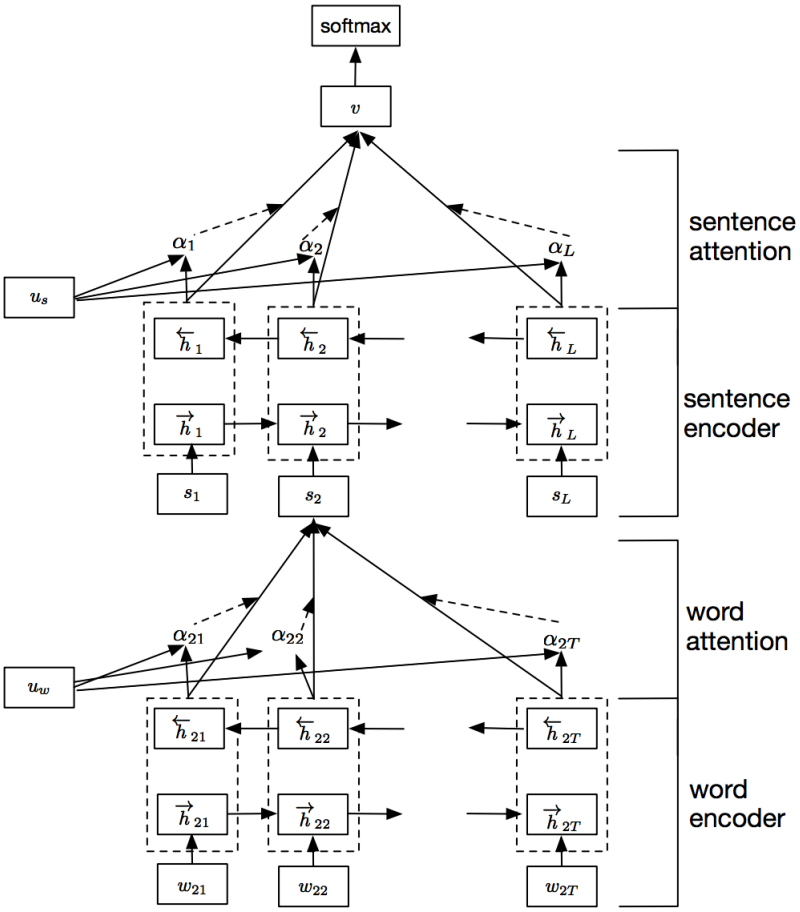

Figure 2.2. Architectural diagram showing the components of a HAN

for words and segments in a bottom-up fashion and then uses them to progressively obtain a document representation for the full text.

1. **Word Encoder**: Given a sentence s_i containing n words $w_{ij}, j \in [1, n]$, the model first embeds the words into a d-dimensional space (embeddings may be initialized using pre-trained embeddings like GloVe or word2vec). These embeddings are then fed to a bidirectional GRU that captures sequential information across the segment. The final representation for a word h_{ij} is obtained by concatenating the forward and backward hidden states associated with the bidirectional GRU; as shown below:

$$
\begin{aligned}
x_{ij} &= \text{BERT}(w_{ij}) \\
\overrightarrow{h_{ij}} &= \overrightarrow{\text{GRU}}(w_{ij}) \\
\overleftarrow{h_{ij}} &= \overleftarrow{\text{GRU}}(w_{ij}) \\
h_{ij} &= [\overrightarrow{h_{ij}}; \overleftarrow{h_{ij}}]
\end{aligned}
$$

2. **Word Attention**: Next, the model uses the attention mechanism to figure out which words are more important within the sentence. To do this, the representation h_{ij} is first passed through a linear layer to obtain a representation u_{ij}. To obtain the importance weight of a word α_{ij}, multiply the vector u_{ij} with a context vector u_w and pass it through a softmax function. The final representation of a sentence is obtained by multiplying the hidden state representation h_{ij} and the weight vector α_{ij}.

$$
\begin{aligned}
u_{ij} &= \tanh(W_w h_{ij} + b_w) \\
\alpha_{ij} &= \frac{\exp(u_{ij}^T u_w)}{\sum_j \exp(u_{ij}^T u_w)} \\
s_i &= \sum_j \alpha_{ij} h_{ij}
\end{aligned}
$$

3. **Sentence Encoder**: Given a document T containing m sentences $s_i, i \in [1, m]$, a bidirectional GRU encodes the sentences to obtain a representation for T. The underlying math is depicted in the equations shown below:

$$
\begin{aligned}
\overrightarrow{h_i} &= \overrightarrow{\text{GRU}}(s_i) \\
\overleftarrow{h_i} &= \overleftarrow{\text{GRU}}(s_i) \\
h_i &= [\overrightarrow{h_i}; \overleftarrow{h_i}]
\end{aligned}
$$

4. **Sentence Attention**: The representation u_i is obtained by passing the hidden representations from the GRU through a linear layer. Then, the context vector u_s is multiplied with u_i to obtain the attention vector α_i. The final representation of the document T will be the product of α_i and h_i. The equations shown below describe the math behind how attention is calculated at the segment-level:

$$u_i = \tanh(W_s h_i + b_s)$$
$$\alpha_i = \frac{\exp(u_i^T u_s)}{\sum_j \exp(u_i^T u_s)}$$
$$T = \sum_i \alpha_i h_i$$

The final representation T will be a representation of the entire document or text, that is obtained by composing the word and sentence representations. This can be fed to a decoder such as a linear layer to perform classification. Since this dissertation is also concerned with document understanding and parsing, HANs will be useful here and can help identify important words and segments that can be used to induce a discourse tree accurately.

2.4 Multi-task Learning

Typically, machine learning models are designed around improving the performance metric (e.g. accuracy) of one task. For instance, to perform document classification, one can build a HAN model as shown in the previous section. However, Caruana (1997) argues that by being laser-focused on one task, one may be ignoring information that could help yield better performance. Specifically, this information comes from related tasks. For example, Named Entity Recognition could benefit from part-of-speech tagging or text classification could benefit from semantic role labeling. By sharing representations between related tasks, one can enable the model to generalize better on the original task. This approach is called Multi-Task Learning or MTL (Ruder, 2017).

One can view multi-task learning as a form of inductive transfer that can help improve a model by introducing an inductive bias. This causes the model to prefer some hypotheses over others. In the case of MTL, inductive bias is provided by auxiliary tasks, which cause the model to prefer hypotheses that explain more than one task. Consequently, this helps the model perform better generalization.

MTL can be performed using one of these ways in deep learning: (1) hard parameter sharing, where MTL is applied by sharing hidden layers between all tasks, while keeping several task-specific output layers; and (2) soft parameter sharing where each task has its own model with its own parameters. Figure 2.3 gives examples of very simple neural network structures that can be used in MTL[3].

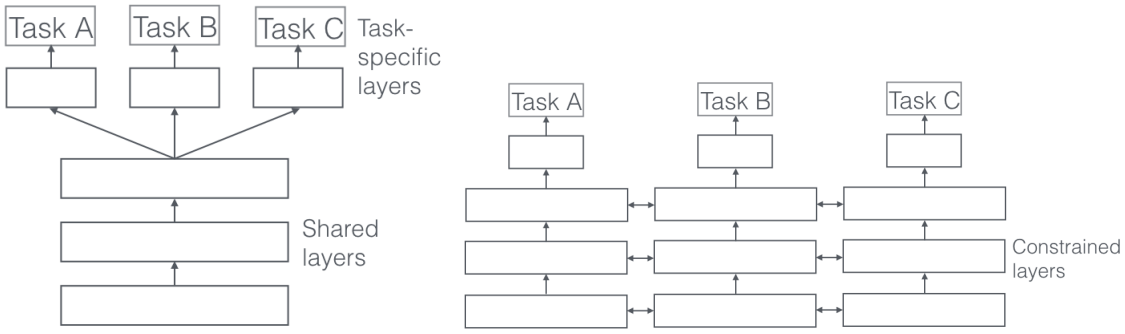

Figure 2.3. Hard and soft parameter sharing multi-task learning frameworks

There are several reasons why MTL leads to improved performances:

1. MTL effectively increases the sample size that is used for training the model. When training a model that performs a certain task A, the objective is to learn a good representation that ignores the data-dependent noise and generalizes well. As different tasks have different noise patterns, a model that learns two tasks simultaneously is expected to learn a better representation. Just learning task A bears the risk of overfitting,

[3]Figure taken from Ruder (2017)

while learning A and an auxiliary task B jointly enables the model to obtain a more general representation by averaging the noise patterns.

2. If a task is very noisy or data is limited and high-dimensional, it could be difficult for a model to differentiate between relevant and irrelevant features. MTL can help the model focus its attention on relevant features as other tasks will provide additional evidence for the importance of those features.

3. Some features G are easy to learn for some task B, while difficult to learn for another task A. This might either be because A interacts with the features in a more complex way or because other features are impeding the model's ability to learn G. Through MTL, one can allow the model to learn G through task B.

4. MTL acts as a regularizer by introducing an inductive bias. This reduces the risk of overfitting and the ability to capture random noise.

2.5 Conclusions

It is fairly evident that deep learning can help perform a myriad of language tasks with high accuracy. It is also evident that leveraging universal sentence representations from a complex language model like BERT can significantly improve the performance of any model. In subsequent chapters, it is empirically shown that fine-tuning BERT with other layers in the model architecture significantly helped improve model performance and achieve results that rival the state-of-the-art.

Likewise, since this dissertation concerns with the identification of document-level semantics, using HANs to model text hierarchy and compositionality can help locate specific parts of the document that are central to understanding its meaning or uncovering a relation.

Finally, joint learning can bolster both discourse segmentation and parsing by providing an inductive bias to prevent overfitting. More importantly, any potential effects of the sparsity of training data on model performance could also be averted.

Subsequent chapters will show each of these models or paradigms in action and provide empirical evidence that support claims made in this chapter.

CHAPTER 3

DISCOURSE SEGMENTATION

Chapter 1 briefly introduced readers to the task of discourse segmentation and its significance in the context of discourse parsing. This chapter describes a deep, accurate model for performing discourse segmentation.

3.1 Introduction

3.1.1 Problem Definition

Discourse segmentation refers to the task of fragmenting a document into minimal disjoint chunks of text called elementary discourse units or EDUs. In the context of RST-style discourse parsing, these EDUs form the leaves of the discourse tree; and edges label relations between leaves and/or subtrees. Subsequent examples show how a sentence can be segmented into EDUs (segments are shown via bracketing):

1. [The results underscore Sears's difficulties] [in implementing the "everyday low pricing" strategy] [that it adopted in March, as part of a broad attempt] [to revive its retailing business.]

2. ["We have tried our best to tell the people in Bataan] [that maybe this time it will not go to them,] [but certainly we will do our best to encourage other investors to go to their province,"] [Mrs. Aquino told Manila-based foreign correspondents.]

It should be noted that while the definition of segmentation is applicable to the full document, discourse segmentation is an intra-sentential phenomenon. In other words, one can begin by first identifying all sentences in the document and then running a segmenter model on top of each sentence. Following previous research (Lin et al., 2019; Muller et al.,

2019), it is assumed that gold tokenization and sentence boundary detection outputs are available and segmentation is performed on the sentence (not the full text).

3.1.2 Challenges to discourse segmentation

While at first discourse segmentation looks essentially like clause delimitation, it is considered a challenging problem for several reasons.

First, the boundary between syntax and discourse is blurry. While Mann and Thompson (1988) intuitively defined relations between clauses, Carlson et al. (2003) argue that applying this idea to the task of consistently annotating a large corpus is difficult as relations hold between arbitrary spans of text; regardless of whether or not they are grammatically or lexically linked. To motivate the idea, consider the following examples:

1. [Xerox Corp.'s third-quarter net income grew 6.2% on 7.3% higher revenue.] [This earned mixed reviews from Wall Street analysts.]

2. [Xerox Corp.'s third-quarter net income grew 6.2% on 7.3% higher revenue,] [which earned mixed reviews from Wall Street analysts.]

3. [Xerox Corp.'s third-quarter net income grew 6.2% on 7.3% higher revenue,] [earning mixed reviews from Wall Street analysts.]

4. [The 6.2% growth of Xerox Corp.'s third-quarter net income on 7.3% higher revenue earned mixed reviews from Wall Street analysts.]

These examples convey the same meaning i.e. the presence of a consequence relation, but range in structure from two distinct sentences to a single clause. Ideally, one would want to capture such kind of rhetorical information irrespective of the syntactic form in which it is represented. However, as example 4 indicates, separating syntax from semantic analysis is not always easy as the sentence here is not segmented into two EDUs.

Second, there is little to no agreement on what exactly constitutes a EDU. Some researchers (Grimes, 1972; Givon, 1984) define clauses as EDUs, while some regard sentences (Polanyi, 1988) as basic units of discourse. Likewise, some researchers also suggest taking prosodic units (Hirschberg and Litman, 1993), turns of talks (Sacks et al., 1978) or intentionally defined discourse segments (Grosz and Sidner, 1986) as EDUs. In order to find some balance between the level of granularity and carrying out annotations consistently, annotators came up with a set of well-defined principles to demarcate segment boundaries (Carlson et al., 2003). While the clause is considered a basic EDU, segment boundaries are often determined via lexical and syntactic cues. Descriptions and examples of these principles are provided in Appendix A. It is important to note that these principles are fairly complicated and it is difficult to construct an accurate, rule-based system around syntactic parse trees that can segment sentences. Furthermore, errors made by systems that generate parse trees can creep into the decision making process, hurting model performance.

Third, the amount of training data available for discourse segmentation is limited. This makes training of complex, data-hungry models like neural networks difficult. For example, the English RST-DT dataset (Carlson et al., 2003) contains 385 documents with 8,313 sentences and 17,646 EDUs. Likewise, the Dutch Discourse Treebank (Redeker et al., 2012) contains 80 documents having 1,707 sentences fragmented into 2,041 EDUs.

3.1.3 Related Work

Related work in discourse segmentation can be broadly classified into 3 types: models that make use of (1) well-defined rules to obtain EDUs; (2) feature-driven traditional machine learning algorithms; and (3) deep learning. Likewise, models can be also be classified on the basis of whether they perform (1) token or boundary classification i.e. decide if a token represents the beginning (boundary) of a new EDU or not; and (2) sequence tagging i.e. use B-O tagging schemes to classify each token in the input as the beginning of a new EDU (B)

or continuation of a previous EDU (I). While several segmenters have been built over the years, this section describes the most important or frequently analyzed ones.

Soricut and Marcu (2003) introduced a probabilistic model called SPADE for performing sentence-level discourse segmentation. They argue that injecting syntax into the decision making process can help perform accurate discourse segmentation. The model makes use of lexical and syntactic features such as POS tags, lexical heads of tokens as obtained from constituency parse trees, etc. to decide if a token represents the beginning of a new EDU or not. Using syntactic parse trees extracted via the Charniak parser (Charniak, 2000), the model achieved an F-score of 84.7.

Subba and Di Eugenio (2007) follow Soricut and Marcu (2003) and model discourse segmentation as a token classification problem. They use feed-forward neural networks with a rich set of lexical and syntactic features like POS tags, syntactic parse tree features and discourse cues to classify a token as the beginning of a new EDU or the continuation of a previous EDU. They improved the F-score obtained by Soricut and Marcu (2003) to 86.0.

Hernault et al. (2010) modelled the task as sequence classification. They used the same set of lexical and syntactic features as used by the work described above and trained a Conditional Random Field (CRF) to classify tokens in a sentence with B-I tags. Performing segmentation as sequence labelling helped improve the performance significantly: specifically strong gains were observed in the precision score, leading to an improved F-score of 89.0. Further, (Bach et al., 2012) built a reranking model on top of Hernault et al. (2010) using subtree feature to further enhance the F-score to 91.0.

While these models achieved decent F-scores, the underlying issue of data insufficiency prevented them from doing even better. To assuage this problem, Wang et al. (2018) and Lin et al. (2019) proposed leveraging sentence representations from ELMo (Peters et al., 2018). Wang et al. (2018) suggested using a restricted form of self-attention that minimizes the size of window of tokens for carrying out segmentation and improved model performance.

Likewise, Lin et al. (2019) suggested using pointer networks (Vinyals et al., 2015) to encode and decode the sentence being segmented. With BERT (Devlin et al., 2019) emerging as a popular language model for obtaining universal input representations, Muller et al. (2019) introduced TonY: a model that leverages pre-trained representations from BERT and contextualized character embeddings to perform accurate discourse segmentation and achieve an impressive F-score of 96.0.

Some important lessons that can be learned from previous work done is that the quality of segmentation depends on (1) smartly leveraging syntactic and lexical features; and (2) utilizing pre-trained embeddings from universal sentence encoders like BERT and ELMo.

3.1.4 Dissertation Contributions

Following observations on previous research, this chapter proposes leveraging BERT's structure and jointly learning syntactic features like part-of-speech tags and dependency parse tree features to perform discourse segmentation. Main contributions of this chapter are:

1. Discourse segmentation is cast as a token classification problem, as opposed to sequence tagging. This allows BERT to attend to one token at a time and not the entire sequence enabling it to make better decisions. It is shown, both qualitatively and quantitatively that representing the input in this fashion helps significantly improve the performance of discourse segmenter. Performance gains are particularly observed in the precision score, which is a common drawback faced by all previously designed segmenters.

2. A simple multi-task learning approach is suggested that uses the intermediate layers of BERT to carry out part-of-speech tag prediction, and dependency relation classification. This helps model performance, particularly for languages other than English. Unlike related work that uses POS tags or parse trees as features, this work suggests learning them in a multi-task fashion. Performance gains are observed, particularly

for low-resource languages like Basque and Portuguese Brazilian, which highlights the importance of learning syntactic features and even points out some limitations of a model as complex as BERT in working with such languages.

3. Experiments were performed for different languages to demonstrate the cross-lingual effectiveness of the framework. Multilingual BERT [1] was used as the sentence encoder. Previous research work has focused mostly on English. This work suggests a model that can work well with any language, boasting very good results across all.

4. A qualitative explanation is provided via error analysis. This provides deeper insights into the models behaviour. Specifically, it is shown that proposed model enhancements allow it work better with longer sentences and capture syntactic phenomena that are not identified by baseline models.

3.2 Model Description

Figure 3.1 provides a logical view of the proposed model and its components. Subsequent subsections describe the model and related details.

3.2.1 Multilingual BERT Encoder

To obtain sentence representations, the model makes use of multilingual BERT as its encoder. To work with multiple languages, Devlin et al. (2019) released multilingual BERT: a single language model that was pre-trained from shared multi-lingual corpora in more than 100 languages; using a shared multi-lingual vocabulary. Despite being a shared language model that could potentially raise concerns about its effectiveness, it was empirically shown that this model gives surprisingly good results for many language tasks (Pires et al., 2019; Wu and Dredze, 2019).

[1] https://www.github.com/google-research/bert

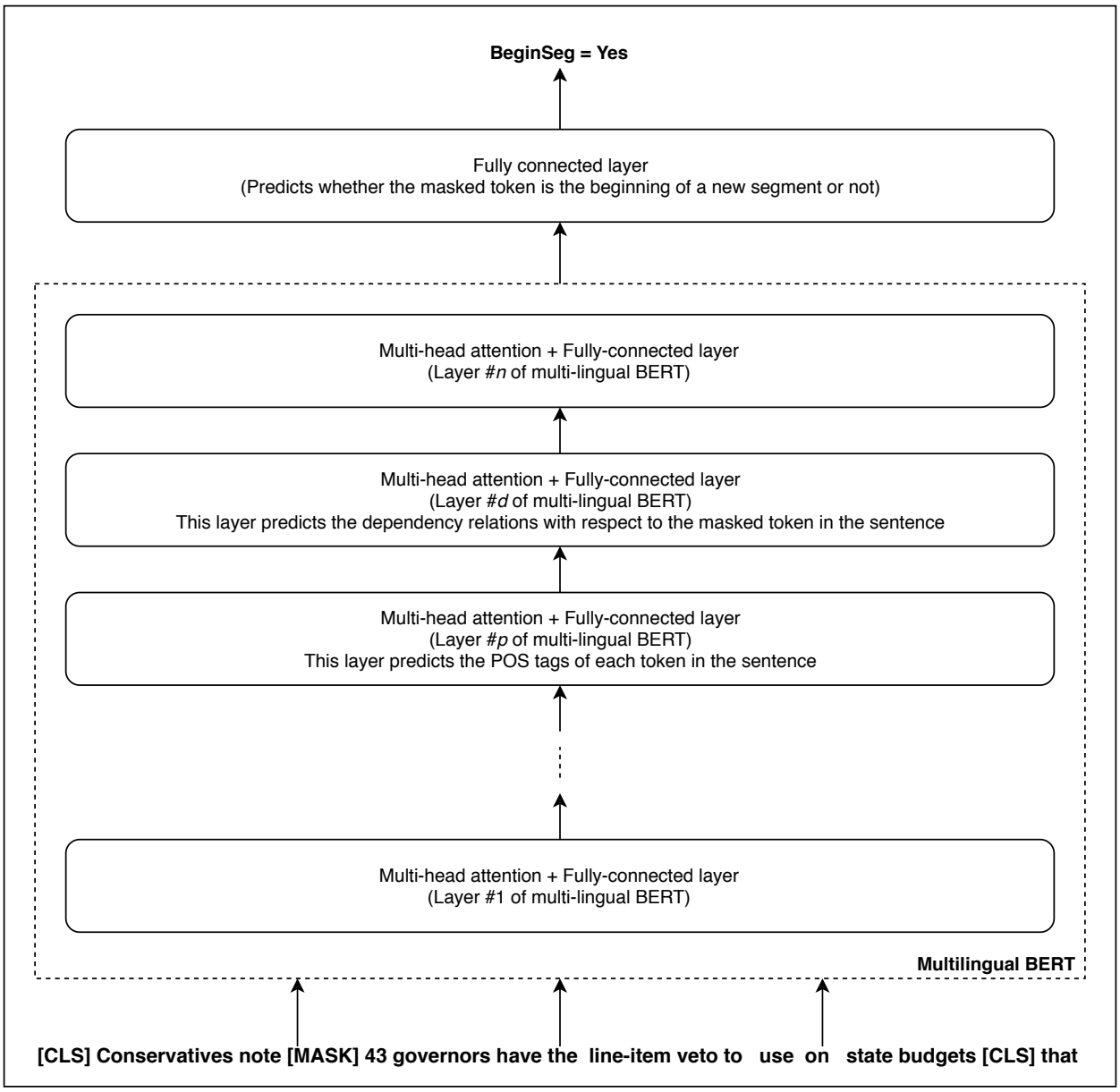

Figure 3.1. Model Architecture for Discourse Segmentation: The model attempts to classify if the masked word 'that' represents the beginning of a new segment or not. The intermediate layers carry out feature prediction i.e. the POS tags and dependency relations of all tokens with respect to masked token in the sentence.

3.2.2 Input representation

As suggested by previous research, the problem can be either formulated as a sequence tagging problem or as a token classification problem. Typically, deep learning frameworks

cast this as a sequence tagging problem i.e. as a B-I tagging problem where B marks the beginning of a segment and I indicates the continuation of a previous segment. Specifically, if BERT is to be used, a sequence of tokens $t_1, t_2, \ldots t_n$ would be represented as:

$$[\text{CLS}]\ t_1, t_2, \ldots t_n\ [\text{SEP}]$$

where [CLS] and [SEP] are special tokens used by BERT.

The proposed model casts this problem as token classification. Specifically, given a sequence of tokens $t_1, t_2, \ldots t_n$, for each token t_i, the input is to BERT is represented as:

$$[\text{CLS}]\ t_1, t_2, \ldots\ [\text{MASK}]\ \ldots t_n\ [\text{SEP}]\ t_i$$

where the token t_i is replaced by a special [MASK] token.

A similar idea was applied to sentence boundary detection (Schweter and Ahmed, 2019) where a window of k characters was defined for markers such as periods, question marks, exclamation points, etc., and a deep network was trained to identify if the marker represents a sentence boundary or not. In this case, markers are well-defined (i.e. a sentence must end with a period, a question mark, an exclamation point or quotation marks). Unfortunately, for discourse segmentation, such markers are not well-defined which requires checking all tokens in a sentence for EDU boundaries.

It can be conjectured that casting the problem this way allows BERT to make better decisions as it attends to each token and not the full sequence. This is particularly useful for discourse segmentation as Wang et al. (2018) observed that demarcating a segment requires only on a small window of neighbouring tokens. Trying to tag the full sequence may introduce unnecessary noise and can lead to errors.

3.2.3 Joint Learning of Syntactic Features

Syntactic features are very helpful in learning language tasks. Strubell et al. (2018) particularly observed that jointly learning syntactic features improved the performance of semantic role labeling systems. The idea is extended to discourse segmentation, by jointly predicting the following features for the sentence:

1. Part-of-speech tags of all tokens in the sentence
2. Dependency parent, child(ren) and sibling(s) of the masked token
3. Dependency relation of the parent with masked token

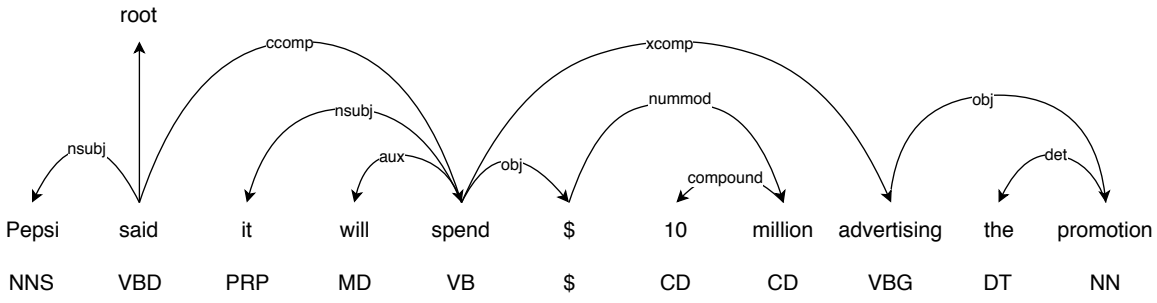

Figure 3.2. An illustrative example showing the dependency parse tree and features extracted for each sentence.

An example is provided in Figure 3.2. The following features are extracted: part-of-speech tags of all tokens; and the dependency parent (and corresponding relation), child(ren) and siblings of the masked token. For example, if the token 'spend' is masked, the classifier is trained to learn CCOMP relation between 'said' and 'spend'; CHILD relations with respect to the tokens 'it', 'will', '$', and 'advertising'; SIBLING relation with respect to the token 'said' and NOREL with respect to every other token.

Joint training offers many advantages. Some of these have been listed in Chapter 2. Here, it must be additionally noted that one can make use of gold POS tags and parse trees

to provide signals for multi-task learning. This is particularly advantageous, as Braud et al. (2017) observed, system-generated parse trees adversely affect segmentation as opposed to gold trees that improved performance considerably.

As illustrated in Figure 3.1, the first p layers of BERT learn part-of-speech tags of tokens under consideration. This layer passes information learned to the upper layers: d layers learn dependency parse tree based features. The final layer ($n = 12$ in case of BERT) provides the final contextualized representations that are fed to the decoder for performing classification.

3.2.4 Decoder

The design of the decoder is fairly simple. A fully connected layer is placed on top of BERT that accepts the final hidden representation of the sentence and predicts whether the masked token represents the beginning of a new segment or not. The softmax activation function is utilized to convert the linear layers output to a probability distribution over labels B and I.

3.3 Experimental Results

3.3.1 Data

To test the performance of the model, experiments were performed with datasets in 5 different languages:

1. The RST-DT corpus (Carlson et al., 2003) in English (eng.rst.rstdt)
2. The Potsdam Commentary corpus (Stede and Neumann, 2014) in German (deu.rst.pcc)
3. The Dutch Discourse Treebank (Redeker et al., 2012) (nld.rst.rstdt)
4. The Cross-document Structure Theory News Corpus (Cardoso et al., 2011) in Portugeuse Brazilian (por.rst.cstn)

5. Basque Discourse Treebank (Iruskieta, Aranzabe, de Ilarraza, Lersundi, and de Lacalle, Iruskieta et al.) (eus.rst.rstdt)

Some statistics on how data is distributed in these corpora is provided in Table 3.1.

Table 3.1. Description of how the data is distributed in each dataset. Notice that the amount of training data available for languages like Dutch and Basque is too small.

Dataset	**Training**			**Validation**			**Test**		
	# Docs	# Sents	# EDUs	# Docs	# Sents	# EDUs	# Docs	# Sents	# EDUs
eng.rst.rstdt	309	6,672	17,646	38	717	1,797	38	929	2,346
deu.rst.pcc	142	1,773	1,788	17	207	275	17	213	294
nld.rst.nldt	56	1,202	1,350	12	257	347	12	248	344
por.rst.cstn	110	1,595	1,772	14	232	552	12	123	265
eus.rst.rstdt	84	990	1,517	28	350	604	28	320	593

3.3.2 Setup and Code

This tool was implemented in PyTorch [2]. The API provided by researchers at HuggingFace[3] was used to fine-tune pre-trained BERT. All experiments were performed using NVIDIA-GTX 1080 Ti GPUs. The code snippet shown below provides the core logic behind all ideas described in the previous section:

```
import torch.nn as nn

class BERTForDiscourseSegmentation(nn.Module):
    def __init__(self, language_model, pos_dict, dep_dict,
    hidden=768, classes=2):
        super(BERTForDiscourseSegmentation, self).__init__()
        self.language_model = language_model # BERT
        self.segment_classifier = nn.Linear(hidden, classes)
        self.pos_classifier = nn.Linear(hidden, len(pos_dict))
```

[2]https://pytorch.org

[3]https://github.com/huggingface/transformers

```
        self.dep_classifier = nn.Linear(hidden, len(dep_dict))
        self.loss = nn.CrossEntropyLoss()
        self.num_pos = len(pos_dict)
        self.num_dep = len(dep_dict)

    def forward(self, input_ids, input_masks, input_segments,
    input_labels=None, input_pos=None, input_parent=None, pos=10,
    dparent=10):
        pooled_output = self.language_model(input_ids,
        attention_mask=input_masks,
        token_type_ids=input_segments)[2]
        predicted_relations =
        self.segment_classifier(pooled_output[12])[:, -1]
        if input_labels is not None:
            predicted_tags =
            self.pos_classifier(pooled_output[pos])[:,-1]
            predicted_dep_rels =
            self.dep_classifier(pooled_output[dparent])
            return self.loss(predicted_relations.view(-1, 2),
            input_labels.view(-1)) +
            self.loss(predicted_dep_rels.view(-1, self.num_dep),
            input_parent.view(-1)) +
            self.loss(predicted_tags.view(-1, self.num_pos),
            input_pos.view(-1))

        return predicted_relations
```

For training, batches of size 16 were constructed. Cross-entropy was used for calculating network loss. The Adam optimizer (Kingma and Ba, 2014) was used for updating network weights. The learning rate is set to $3e-5$. To tune the hyper-parameters, 10% of the training data was held out as validation set: all hyper-parameters were tuned on this validation set.

Experiments were conducted for different values of $p, d \in \{9, 10, 11\}$, but no significant difference in the F-scores (± 0.15) were observed. The best results for each dataset are reported in this dissertation.

3.3.3 Empirical Results

Empirical results for discourse segmentation are reported in Table 3.2. As a baseline, segmentation is cast as a BI tagging problem, following the guidelines provided by Devlin et al. (2019). One can see that by casting it instead as token classification (Token), the F-score improved significantly. In fact, simply casting the problem as token classification got us very close to the state-of-the-art for the English RST-DT corpus (Muller et al., 2019). Likewise, training for part-of-speech tags (Post) and dependency relations (Depend) also improved the F-score for each language. The best improvements were observed for the German and Dutch datasets; with F-scores improving by 4.47 and 2.87 points respectively.

Table 3.2. Empirical results and ablation study: As is evident from the obtained results, casting the problem as a token classification (Token) problem led to significant improvement. Likewise, training for POS tags (Post), dependency relations (Depend) or both (Both) improved model performance across all languages.

Model	eng.rst.rstdt			deu.rst.pcc			nld.rst.rstdt			por.rst.cstn			eus.rst.rstdt		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	86.86	90.41	88.60	84.91	91.84	88.24	84.87	88.08	86.44	84.67	83.40	84.03	75.85	77.46	75.66
Token	95.45	94.67	95.06	**94.76**	86.05	90.20	**97.69**	86.05	91.49	**92.88**	88.68	90.73	**87.25**	80.78	83.89
Post	96.17	96.21	96.19	93.34	95.58	94.45	93.86	93.31	93.59	87.72	**94.34**	90.91	84.63	84.49	84.56
Depend	94.41	**97.19**	95.78	92.74	95.58	94.14	95.73	91.28	93.45	90.98	91.32	91.15	88.87	82.13	85.36
Both	95.24	95.48	95.36	93.81	92.86	93.33	93.02	93.02	93.02	91.60	90.57	91.08	86.02	81.96	83.94
Ensemble	**96.32**	97.02	**96.67**	92.81	**96.60**	**94.67**	94.72	**93.90**	**94.31**	89.53	93.59	**91.51**	85.59	**85.16**	**85.38**

It can also be observed that training either for POS tags or for dependency relations improves the F-score more significantly as compared to training for both (Both). A likely explanation for this is that training for both leads to poor generalization thereby leading to comparatively poor improvements. To assuage the problem, number of training iterations were increased, but this led to severe overfitting. In general, ensembling (Baseline and Both

are not considered during ensembling) helped and gave better scores when compared to these models in isolation.

3.4 Analysis of Performance on English dataset

3.4.1 Comparison with related work

The analysis section begins by first comparing the results obtained for the eng.rst.rstdt corpus with previous work done. To ensure fair comparison, results are reported for discourse segmentation at the sentence-level and not at the document-level (Braud et al., 2017). Table 3.3 reports the performance of the designed model and other competing systems. A description of these systems has been provided in previous sections and is not repeated here for the sake of brevity.

Table 3.3. Performance of the proposed model and other systems on the RST-DT dataset. Results are reported assuming parse trees are extracted using the BLLIP parser (as used by authors in the paper)

Model	P	R	F
Soricut and Marcu (2003)	84.1	85.4	84.7
Subba and Di Eugenio (2007)	85.5	86.6	86.0
Hernault et al. (2010)	91.0	87.2	89.0
Bach et al. (2012)	91.5	90.4	91.0
Feng and Hirst (2014b)	92.8	92.3	92.6
Wang et al. (2018)	92.9	95.7	94.3
Lin et al. (2019)	94.1	96.6	95.4
Muller et al. (2019)	95.3	96.8	96.0
This research	**96.3**	**97.0**	**96.7**
Human	98.5	98.2	98.3

As is evident from Table 3.3, state-of-the-art results were obtained on the RST-DT corpus, beating the previous state-of-the-art model by an absolute 0.7 points and by a relative 7.2 points. It can also be observed that many of these systems were high-recall systems i.e. they end up predicting more EDUs than necessary. The proposed system achieved a higher

precision than all, again beating the state-of-the-art by an absolute 1.0 points or relative 10 points.

3.4.2 Sentence length v/s Number of errors

Next, a comparison of the proportion of errors made by the models with respect to the length of the sentence i.e. the number of tokens in the sentence is carried out. For the purpose of evaluation, a sentence is considered to be incorrectly segmented regardless of whether the type of error (Bach et al., 2012) is over (i.e. a sentence is segmented when it should not be) or miss (a sentence is not segmented when it should be). In Figure 3.3, a graph is provided showing the proportion of errors made by the models with respect to the sentence length. The proportion of errors is calculated as the ratio of sentences that were incorrectly tagged to the total number of sentences, grouped by the sentence length.

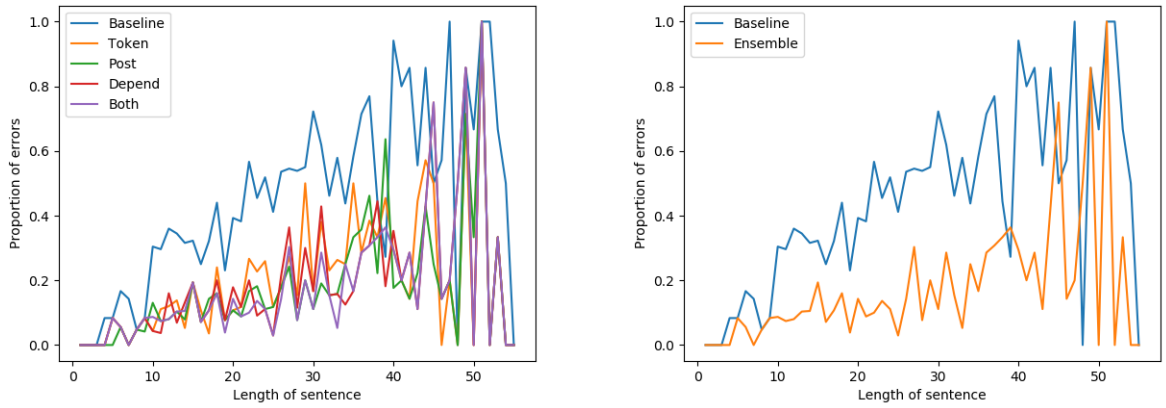

Figure 3.3. Graphs showing the proportion of errors made v/s the length of sentence.

As suspected, the baseline model performs poorly when the sentences are longer. However, formulating the problem in an alternate fashion and injecting syntax make the model perform much better. This effect is more discernible for sentence lengths between 30 and 45, indicating that the baseline model could not segment a large fraction of longer sentences correctly.

3.4.3 Frequent Error Patterns

Table 3.4 lists the 8 most frequent tokens that were incorrectly segmented; and the total number of errors made by each model. As one can infer from Table 3.4, all models give fewer errors than the baseline model. The reduction in total number of errors is more than 62% highlight the efficacy of the models. Additionally, training for syntax further reduced these errors by more than 5%.

Table 3.4. Table showing the number of errors made by all models when tagging the 8 most frequent tokens.

Token	**Absolute number of Errors**				
	Baseline	Token	Post	Depend	Both
and	40	21	15	20	16
that	32	11	8	6	7
to	31	22	18	21	16
the	17	9	9	8	7
as	14	4	2	3	1
in	10	3	4	3	2
for	10	7	7	6	6
if	10	3	4	3	3
Total	608	231	187	201	200

On mapping these errors to rules (Carlson and Marcu, 2001), the following were found to be most frequently violated:

1. **Confusion between infinitival complements and infinitival clauses**: Infinitival components of verbs are never fragmented into separate EDUs, whereas infinitival clauses are segmented only if that clause functions as the satellite of a PURPOSE relation. The model often confuses infinitival complements for infinitival clauses, which leads to tagging errors.

2. **Coordination in Sentences and Clauses**: Coordinated sentences and clauses are broken into separate EDUs, while coordinated verb phrases are not. Additionally,

when coordination occurs in subordinate clauses, segmentation depends on whether or not the subordinate construction would normally be segmented as an EDU if it were a single clause, rather than a number of coordinated clauses. The model made some errors in identifying such patterns.

3. **Confusion among correlative subordinators**: Correlative subordinators consist of a combination of two markers, one in the subordinate clause and the other in the superordinate clause. Examples include 'as ... long as', 'either ... or', etc. These should be broken into separate EDUs, provided the subordinate clause contains a verb. There was some confusion in correctly identifying such constructs and thus performing accurate segmentation.

4. **Punctuation**: Punctuation symbols often indicates segment boundaries. However, there may be cases where EDUs are not segmented. For instance, parenthetical expressions are usually segmented as EDUs, but if the expression is used to indicate missing information, segmentation must not be carried out. Likewise, phrases separated by semi-colons and commas are not EDUs.

While modeling segmentation as token classification helped remove these errors, injecting syntax helped remove these errors further. In particular, it is observed that jointly training for part-of-speech tags helped remove punctuation errors and resolve confusions between infinitival complements and clauses. Likewise, jointly training for dependency relations helped remove errors related to coordination and correlative subordinators. Concrete examples of each error type are provided in Table 3.5.

3.5 Conclusions

Results obtained and analysis performed show how incorporating syntax into the model helped achieve better results. In particular, the joint learning of syntactic features allowed

Table 3.5. Some common errors made by the model are highlighted. It is also shown learning syntactic features helped eliminate such errors.

1	**Segments**: [With 700 branches in Spain and 12 banking subsidiaries, five branches and 12 representative-offices abroad, the Banco Exterior group has a lot] [**to** offer to a potential suitor.] **Explanation**: Infinitival complements of verbs are not segmented as separate EDUs. However, both Baseline and Token confuse the infinitival clause in the sentence for a infinitival complement and end up leaving the sentence as a single EDU. Training for dependency relations allowed the model to identify this correctly as an infinitival clause and perform correct segmentation.
2	**Segments**: [The government directly owns 51.4%] [**and** Factorex, a financial services company, holds 8.42%] **Explanation**: The baseline makes an error as it cannot identify that the sentence contains a superordinate and a subordinate clause; and therefore must be segmented. Training for both part-of-speech tags and dependency relations allowed the model to correctly identify these as two different clauses and segment them.
3	**Segments**: [A private market like this just isn't big **enough**] [**to** absorb all the business.] **Explanation**: The baseline model fails to identify the comparative 'enough . . . to' as a correlative and does not segment the sentence. However, training for syntactic features allowed the model to correctly identify this construct and hence perform correct segmentation.
4	**Segments**: [On the Big Board, Crawford & Co., Atlanta,] [**(CFD)**] begins trading today] **Explanation**: Both baseline and token incorrectly assume that the parenthetical expression expresses missing information and must not be segmented. However, by predicting the POS tag of CFD as NNP which is the same as the POS tags of the words preceding it, learning POS tags allowed the model to correctly segment this sentence.

the model uncover complex syntactic patterns that could not be captured by simply fine-tuning BERT. Further, the use of syntactic features helped achieve solid gains for languages such as German and Dutch; highlighting both the importance of syntax; and also certain limitations of multilingual BERT.

Several complex cases of discourse segmentation could be effectively captured by the proposed model. Having knowledge of sentence-level semantics may help identify such nuanced patterns even better. This was in fact empirically proven by Lin et al. (2019) who jointly carried out discourse segmentation and coherence relation classification, observing an incremental improvement in model performance.

A potential drawback of the system is that the time taken to tag a full sequence is quite large as the model performs sentence segmentation in $O(n)$ time while other models take $O(S)$ time, n being the number of tokens in the document, and $S << n$ being the number of sentences in the document. However, with a sufficiently large batch size; and the availability of multiple GPUs, this bottleneck can be practically resolved by performing sentence segmentation in parallel.

CHAPTER 4

DISCOURSE PARSING

To recapitulate, in the context of RST (Mann and Thompson, 1988), a parser operates in two stages: (1) Discourse segmentation, where the text is partitioned into EDUs; and (2) Tree induction, where a discourse tree is made from segments. Chapter 3 described a model for discourse segmentation. This is an important pre-requisite to carrying out tree induction. This chapter describes a parsing model for RST-style discourse parsing.

4.1 Introduction

4.1.1 Problem Definition

Discourse parsing refers to the task of associating text with a hierarchical discourse tree that illustrates how its components are logically connected to each other. The tree is structured such that segments form its leaves, subtrees represent text spans, and arcs label relations between them. An important characteristic of this tree is that it is hierarchical in nature: granularity plays a very important role in depicting the level of the relation. Lower rungs of the tree indicate relations between segments or EDUs and are largely intra-sentential in nature. While upper rungs of the tree indicate higher level relations occur between text spans and inter-sentential in nature. For a much bigger document, one may find relations between even larger spans of text such as groups of sentences to paragraphs.

To understand this, consider an example shown in Figure 4.1. Leaf nodes (labelled as 1, 2 and 3) represent segments and edges between leaves and/or subtrees represent relations.

From the perspective of semantics, the discourse tree provides a condensed view of the meaning of text. While lower rungs show low-level semantic relations between small spans of text, upper levels of the tree capture higher level relations between larger spans of texts. The root of the tree represents the main idea or the theme of the text. For instance, in the

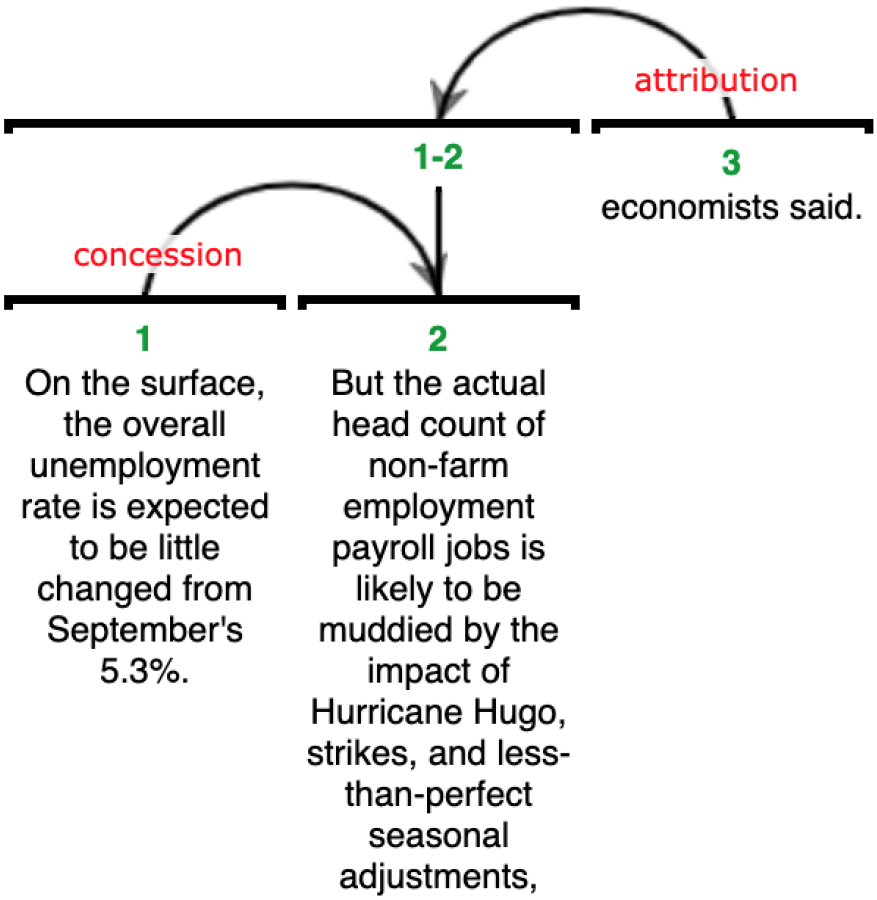

Figure 4.1. An example showing the discourse tree for a small document (Zeldes, 2016).

tree shown in Figure 4.1, the ATTRIBUTION relation represents the main idea of the text, while the CONCESSION relation describes a relation between two smaller spans of text i.e. leaf nodes 1 and 2.

Additionally, discourse parsing directly finds application in many language tasks, like text summarization (Uzêda et al., 2008), machine translation (Joty et al., 2017) and question answering (Verberne, 2007) among others.

While the original RST defines more than 75 relations that can potentially hold between text spans, this dissertation focus on the most commonly studied relations for each language. For instance, research efforts are focused on 13 most commonly found relation types for the

English dataset (Carlson and Marcu, 2001). Definitions and examples of each relation type can be found in Appendix B.

4.1.2 Challenges to Discourse Parsing

Text-level discourse parsing is a challenging problem for several reasons. Inducing the discourse tree requires a thorough understanding of the document's structure and how its components organize text. It involves several complex and interacting factors, touching upon all layers of linguistic analysis, from syntax, semantics up to pragmatics. Additionally, in order to perform accurate parsing, one requires knowledge of related linguistic tasks like coreference resolution, sentence selection and semantic relation classification (Braud et al., 2016).

As a direct consequence of this issue, coming up with labelled data for relation classification is equally difficult and time-consuming (Carlson and Marcu, 2001). The sparsity of training data makes it difficult to construct and train neural network based parsers.

Likewise, in some cases, more than one relation may hold between two textual segments. For example, a causal and a temporal relation may hold between two segments simultaneously. The goal of the annotators was to label each node in a rhetorical structure tree with only one relation. To resolve ambiguities, annotators preferred a local comparison of saliency of relations as a criterion (Carlson and Marcu, 2001). For example, TEMPORAL was regarded as more general, and thus as less salient, than CONDITION. This may cause parsers to confuse between two relations: this can be visualized in the confusion matrix depicted in Lin et al. (2019) where the model confused EXPLANATION for CAUSE; BACKGROUND for TEMPORAL and so on.

It is also worth noting that unlike constituency parsing where clear production rules are utilized to define phrases and sentences, a similar strategy cannot be used here directly (Li et al., 2014). This prevents one from coming up with a deterministic parsing algorithm that

often relies on such rules. However, previous work has shown that if such production rules were to be defined and used as features, they can slightly improve model performance (Ji and Eisenstein, 2014).

4.1.3 Related Work

Related work in discourse parsing can be classified on the basis of what model is used to classify the relations; and what parsing framework is used to induce the tree structure.

Parsing Frameworks

Discourse parsers may also be classified on the basis of what framework they use to construct the tree i.e. either they use transition-based systems or chart parsing. Transition-based systems construct the tree greedily by performing a series of shift-reduce actions in either a top-down or bottom-up fashion. While the latter use a ranking function to score trees and identify the best-scoring representation using a dynamic programming algorithm like CKY. Transition-based systems offer the advantage of inducing the tree in $O(n)$ time where n is the number of segments or EDUs in the document; while chart parsing frameworks take $O(n^3M)$ time where M is the number of relations labels, usually a corpus-dependent constant.

An important lesson learned from previous research is that it is more time-efficient to use transition-based parsing over chart parsing. It has been empirically observed that despite being a greedy approach, transition-based parsing does not yield significantly poor results when compared to chart parsing.

Previous Research

Related work in text-level discourse parsing can be broadly categorized into two types: (1) traditional feature-based models and (2) models that leverage neural networks. Feature-based models rely on manually crafted features extracted from constituency and/or dependency parse trees and make use of traditional machine learning models like SVMs to perform

discourse parsing. Neural nets on the other hand implictly learn these features and are capable of providing better text representations than feature-based classifiers. This section describes the most popular or commonly analyzed text-level discourse parsers.

Feng and Hirst (2014a) uses a feature-driven bottom-up greedy parser with linear-chain CRFs that exploits two levels of granularity in a document to perform discourse parsing. Ji and Eisenstein (2014) use a feature-driven shift-reduce parser with an SVM classifier to induce the discourse tree. Li et al. (2016) uses hierarchical LSTMs with a chart parsing framework to encode the document and use this representation to generate the discourse tree. Braud et al. (2016) perform discourse parsing in a multi-task learning setup where they learn related tasks like coreference resolution, temporal relation extraction, etc. to improve model performance. Mabona et al. (2019) propose a generative model using word-level beam search for discourse parsing.

Following previous research, it can be conjectured that it is important to capture three phenomena to perform accurate discourse parsing: (1) Structure i.e. how different components come together to compose the text (Li et al., 2016); (2) Context i.e. what role does a word or segment play in the context in which it occurs (Kishimoto et al., 2020) and (3) Syntax i.e. how a discourse tree can be composed from the syntactic parse trees of its constituent sentences (Yu et al., 2018).

4.1.4 Dissertation Contributions

With respect to discourse parsing, main ideas proposed in this chapter can be summarized thus:

1. To capture structure and context, the problem is modelled using hierarchical attention networks or HANs (Yang et al., 2016). HANs are useful here as they model the document's layout by attending to multiple levels of the text hierarchy i.e. by differentially focusing on words that make up segments; which in turn constitute the full

text. This allows one to specifically attend to those parts of a text span that contribute significantly to its meaning while giving less importance to the rest.

2. To capture context and as well to assuage the data insufficiency problem, the BERT language model (Devlin et al., 2019) is leveraged by fine-tuning it along with other layers of the model. As with discourse segmentation, BERT is expected to provide high quality text representations that can be utilized to perform accurate parsing.

3. To capture syntax, syntactic features like part-of-speech tags and dependency parse trees are jointly learned while training the model. Previous research has empirically shown that learning such features can help improve discourse parsing (Yu et al., 2018), particularly for low-resource languages (Braud et al., 2017) like Basque and Portuguese-Brazilian. This has also been shown in Chapter 3, where learning such features helped improve the performance of the discourse segmenter.

4.2 Model Description

A pictorial representation of the model used to design the parser is shown in Figure 4.2. For defining this model, the following notation is utilized throughout the rest of this chapter:

Define a text span T as a large chunk of text that consists of multiple segments. Assume T has m segments and each segment s_j contains n words. Word w_{ij} represents the ith word in the jth segment where $i \in [1, n]$ and $j \in [1, m]$. The HAN encoder constructs representations for words and segments in a bottom-up fashion and then uses them to progressively obtain a document representation for the full text. These representations will then be fed to a decoder that decides which action needs to be performed at a given timestep.

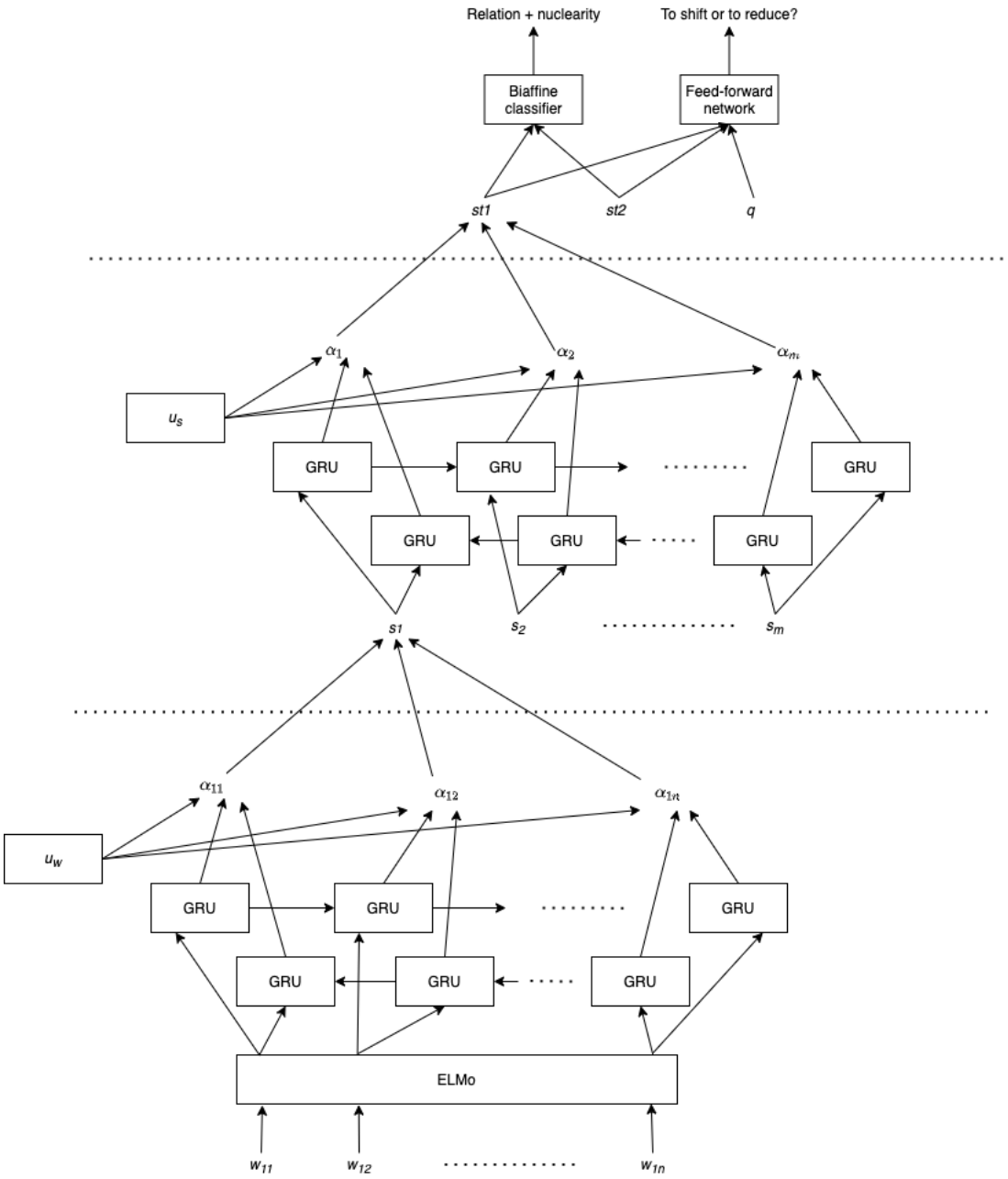

Figure 4.2. Model Architecture for Discourse Parsing: The input to the model will be the encapsulating sentence and not the segment: this has been omitted from the figure for the sake of simplicity.

4.2.1 Shift-Reduce Parsing

A shift-reduce parser (Dyer et al., 2016) is typically implemented using two data structures: a stack and a queue. The stack is used to store partially generated subtrees and the queue is used to hold unprocessed nodes. The parser is initialized by pushing all EDU nodes onto the queue, with the stack being empty. At every timestep, it performs one of these two operations:

1. SHIFT (e): Pop node e from the queue and push it onto the stack.

2. REDUCE (l, d): Pop the first two nodes from the stack, create a subtree with these nodes as its children and push it onto the stack. The relation between the two nodes is l and its nuclearity is $d \in$ {NN, NS, SN}.

Parsing ends when the queue is empty and the stack contains only one tree that represents the full text. The shift-reduce actions performed to get the tree shown in Figure 4.1 are provided in Table 4.1.

Table 4.1. The shift-reduce parser in action

Timestep	Stack	Queue	Action	Explanation
1	{}	$\{e_1, e_2, e_3\}$	SHIFT(e_1)	Push node e_1 onto stack
2	$\{e_1\}$	$\{e_2, e_3\}$	SHIFT(e_2)	Push node e_w onto stack
3	$\{e_1, e_2\}$	$\{e_3\}$	REDUCE(Concession, NS)	Pop nodes e_1, e_2 from stack and create subtree $e_{1:2}$
4	$\{e_{1:2}\}$	$\{e_3\}$	SHIFT(e_3)	Push node e_3 onto stack
5	$\{e_{1:2}, e_3\}$	{}	REDUCE(Attribution, SN)	Pop nodes $e_{1:2}, e_3$ from stack and create subtree $e_{1:3}$
6	$\{e_{1:3}\}$	{}	-	Parsing ends: $e_{1:3}$ represents the full tree

4.2.2 Multilingual BERT

To design the encoder, the parser makes use of BERT (Devlin et al., 2019). In order to work with languages other than English, the parser uses multilingual BERT. To obtain the contextualized embeddings for a sequence of words, the encapsulating sentence (not segment) $S' = \{w_1, w_2, ...w_l\}$ is considered. The input is presented to multilingual BERT as:

$$[\text{CLS}] \quad w_1 \quad w_2 \ldots \quad w_l$$

where [CLS] is a special token used by BERT. If the sequence of words one is interested in is represented as $w_i, w_{i+1}, ...w_j$, only the hidden states associated with these words from the tensor output by BERT are considered.

There are several reasons for representing the input in this fashion. For example, one may input the sequence $w_i, w_{i+1}, ...w_j$, instead of its encapsulating sentence $S^{'}$ to BERT. However, segments are not well-defined syntactic units (like sentences) (Carlson and Marcu, 2001) and previous work has shown that this leads to a significant drop in model performance (Xu et al., 2020). Likewise, instead of building a HAN on top of BERT's output, one may also consider fine-tuning BERT directly for tree induction by inputting text spans to the model. However, BERT has a limitation that input sequences cannot be longer than 512 tokens (or sub-tokens) and this condition is violated in this case as text spans are a lot bigger. While there are work-arounds to handle larger inputs, obtained text representations may not be structure-aware. The next subsection describes how HANs can help incorporate structural information.

4.2.3 HAN Encoder

As described previously, the model must decide which of the two actions i.e. Shift or Reduce should be performed at each timestep. To achieve this, text spans associated with the first two subtrees in the stack and the first subtree in the queue are considered and encoded using the HAN model. Specifically, Gated Recurrent Units or GRUs are used to capture sequence information. Likewise, attention is also applied to identify important pieces of information at the word- and segment-level.

Word-level GRU and Attention

Given a segment s_i containing n words $w_{ij}, j \in [1, n]$, the model first encodes the words using the strategy described previously. These embeddings are then fed to a bidirectional GRU that captures sequential information across the segment. The final representation for a word h_{ij} is obtained by concatenating the forward and backward hidden states associated with the bidirectional GRU; as shown below:

$$
\begin{aligned}
x_{ij} &= \text{BERT}(w_{ij}) \\
\overrightarrow{h_{ij}} &= \overrightarrow{\text{GRU}}(w_{ij}) \\
\overleftarrow{h_{ij}} &= \overleftarrow{\text{GRU}}(w_{ij}) \\
h_{ij} &= [\overrightarrow{h_{ij}}; \overleftarrow{h_{ij}}]
\end{aligned}
$$

Next, the parser uses the attention mechanism to figure out which words are more important within the text span at a given timestep. To do this, the representation h_{ij} is first passed through a linear layer to obtain a representation u_{ij}. To obtain the importance weight of a word α_{ij}, multiply the vector u_{ij} with a context vector u_w and pass it through a softmax function. The final representation of a text segment is obtained by multiplying the hidden state representation h_{ij} and the weight vector α_{ij}.

$$
\begin{aligned}
u_{ij} &= \tanh(W_w h_{ij} + b_w) \\
\alpha_{ij} &= \frac{\exp(u_{ij}^T u_w)}{\sum_j \exp(u_{ij}^T u_w)} \\
s_i &= \sum_j \alpha_{ij} h_{ij}
\end{aligned}
$$

The PyTorch code associated with this is shown below:

```
def matrix_mul(input, weight, bias=False):
```

```
    feature_list = []
    for feature in input:
        feature = torch.mm(feature, weight)
        if isinstance(bias, torch.nn.parameter.Parameter):
            feature = feature + bias.expand(feature.size()[0], bias.size()[1])
        feature = torch.tanh(feature).unsqueeze(0)
        feature_list.append(feature)

    return torch.cat(feature_list, 0).squeeze()

def element_wise_mul(input1, input2):
    feature_list = []
    for feature_1, feature_2 in zip(input1, input2):
        feature_2 = feature_2.unsqueeze(1).expand_as(feature_1)
        feature = feature_1 * feature_2
        feature_list.append(feature.unsqueeze(0))
    output = torch.cat(feature_list, 0)
    return torch.sum(output, 0).unsqueeze(0)

class WordAttNet(nn.Module):
    def __init__(self, bert, embed_size=768, hidden_size=100):
        super(WordAttNet, self).__init__()
        self.word_weight = nn.Parameter(torch.Tensor(2 * hidden_size
        , 2 * hidden_size))
        self.word_bias = nn.Parameter(torch.Tensor(1, 2 * hidden_size))
        self.context_weight = nn.Parameter(torch.Tensor(2 * hidden_size, 1))

        self.bert = bert
        self.gru = nn.GRU(embed_size, hidden_size, bidirectional=True,
        batch_first=False)
        self._create_weights(mean=0.0, std=0.05)
```

```
def _create_weights(self, mean=0.0, std=0.05):
    self.word_weight.data.normal_(mean, std)
    self.context_weight.data.normal_(mean, std)

def forward(self, input, hidden_state):
    torch.autograd.set_detect_anomaly(True)
    output = self.bert(input)[0].permute(1, 0, 2)
    f_output, h_output = self.gru(output.float(), hidden_state)
    output = matrix_mul(f_output, self.word_weight, self.word_bias)
    output = matrix_mul(output, self.context_weight).permute(1, 0)
    output = F.softmax(output, dim=-1)
    output = element_wise_mul(f_output, output.permute(1, 0))

    return output, h_output
```

Segment-level GRU and attention

Given a text span T containing m segments $s_i, i \in [1, m]$, first use a bidirectional GRU to encode the segments and then use attention mechanism in a similar fashion to obtain a representation for T. The underlying math is depicted in the equations shown below:

$$\overrightarrow{h_i} = \overrightarrow{\text{GRU}}(s_i)$$
$$\overleftarrow{h_i} = \overleftarrow{\text{GRU}}(s_i)$$
$$h_i = [\overrightarrow{h_i}; \overleftarrow{h_i}]$$

The representation u_i is obtained by passing the hidden representations from the GRU through a linear layer. Then, the context vector u_s is multiplied with u_i to obtain the attention vector α_i. The final representation of the text span T will be the product of α_i

and h_i. The equations shown below describe the math behind how attention is calculated at the segment-level:

$$u_i = \tanh(W_s h_i + b_s)$$
$$\alpha_i = \frac{\exp(u_i^T u_s)}{\sum_j \exp(u_i^T u_s)}$$
$$T = \sum_i \alpha_i h_i$$

The PyTorch code associated with segment-level GRU and attention is provided here:

```
class SentAttNet(nn.Module):
    def __init__(self, sent_hidden_size=100, word_hidden_size=100):
        super(SentAttNet, self).__init__()

        self.sent_weight = nn.Parameter(torch.Tensor(2 * sent_hidden_size,
        2 * sent_hidden_size))
        self.sent_bias = nn.Parameter(torch.Tensor(1, 2 * sent_hidden_size))
        self.context_weight = nn.Parameter(torch.Tensor(2 * sent_hidden_size,
        1))

        self.gru = nn.GRU(2 * word_hidden_size, sent_hidden_size,
        bidirectional=True, batch_first=False)
        self._create_weights(mean=0.0, std=0.05)

    def _create_weights(self, mean=0.0, std=0.05):
        self.sent_weight.data.normal_(mean, std)
        self.context_weight.data.normal_(mean, std)

    def forward(self, input, hidden_state):
        torch.autograd.set_detect_anomaly(True)
        f_output, h_output = self.gru(input, hidden_state)
```

```
        output = matrix_mul(f_output, self.sent_weight, self.sent_bias)
        output = matrix_mul(output, self.context_weight).permute(1, 0)
        output = F.softmax(output, dim=-1)
        output = element_wise_mul(f_output, output.permute(1, 0))

        return output, h_output
```

4.2.4 Jointly Learning for Syntax

Syntactic features are very helpful in learning language tasks like semantic role labeling (Strubell et al., 2018) and discourse segmentation. Following the previous chapter, idea is extended to discourse parsing, by using BERT to jointly predict the following features for every sentence in the document:

1. Part-of-speech tags of all tokens in a sentence
2. Dependency parent, child(ren) and sibling(s) of the every token in the sentence
3. Dependency relation of parent token with respect to a token in the sentence

4.2.5 Decoder

The decoder performs two actions: First, it predicts what action must be performed at a given timestep. As stated previously, it works with the span representations of the first two nodes in the stack and the first node in the queue. These representations are denoted as st_1, st_2 and q respectively. Second, it predicts the syntax features associated with a sentence during joint training. For joint learning, simple dense layers are used to make predictions. For the former, two special networks are used that are described below:

Position-wise feed-forward network

This module predicts whether the parser must perform a SHIFT or REDUCE action. First, it concatenates the representations st_1, st_2 and q to obtain r: this is then fed to a two-layer feed-forward network (Vaswani et al., 2017) that performs the prediction as shown:

$$y_s = \text{softmax}(max(0, rW_1 + b_1)W_2 + b_2)$$

Here y_s refers to the softmax distribution over two classes: SHIFT and REDUCE.

Biaffine classifier

The biaffine classifier (Dozat and Manning, 2016) is a two-layer network that takes in the span representations st_1 and st_2 as input and predicts the relation and its nuclearity (assuming the reduce action is to be performed). The first layer is a linear layer with ELU activation that maps the span representations into latent features l_1 and l_2. The second layer is a bi-affine layer that obtains a softmax distribution y_r over the labels:

$$l_1 = \text{ELU}(st_1^T U_1) \quad l_2 = \text{ELU}(st_2^T U_2)$$
$$y_r = \text{softmax}(l_1^T W_{12} l_2 + l_1^T W_1 + l_2^T W_2 + b)$$

This code snippet shows how to implement the biaffine classifier:

```
class LabelClassifier(nn.Module):
    def __init__(self, input_size, classifier_hidden_size, classes_label,
    bias=True, dropout=0.1):

        super(LabelClassifier, self).__init__()

        self.classifier_hidden_size = classifier_hidden_size
```

```
        self.labelspace_left = nn.Linear(input_size,
        classifier_hidden_size, bias=bias)
        self.labelspace_right = nn.Linear(input_size,
        classifier_hidden_size, bias=bias)
        self.weight_left = nn.Linear(classifier_hidden_size,
        classes_label, bias=bias)
        self.weight_right = nn.Linear(classifier_hidden_size,
        classes_label, bias=bias)
        self.nnDropout = nn.Dropout(dropout)
        self.weight_bilateral = nn.Bilinear(classifier_hidden_size,
        classifier_hidden_size, classes_label, bias=bias)

    def forward(self, input_left, input_right):
        labelspace_left = F.elu(self.labelspace_left(input_left))
        labelspace_right = F.elu(self.labelspace_right(input_right))

        union = torch.cat((labelspace_left, labelspace_right), 1)
        union = self.nnDropout(union)
        labelspace_left = union[:, :self.classifier_hidden_size]
        labelspace_right = union[:, self.classifier_hidden_size:]
        output = (self.weight_bilateral(labelspace_left, labelspace_right) +
                  self.weight_left(labelspace_left) +
                  self.weight_right(labelspace_right))

        return output
```

4.2.6 Training Loss

The objective of this network is to reduce the overall cross-entropy loss associated with the model. This comprises of three losses: the structure loss that is associated with predicting

correct shift-reduce actions; the relation loss, associated with the biaffine classifier and the joint training loss, that comes from jointly learning for syntax. The overall loss is given by:

$$\mathcal{L}(\theta_P) = \mathcal{L}_s(\theta_P) + \mathcal{L}_r(\theta_P) + \mathcal{L}_j(\theta_B) + \frac{\lambda}{2}||\theta_P||^2$$

where θ_B and θ_P refers to the parameters of BERT and the entire model respectively, $\mathcal{L}_s$ refers to the loss associated with learning structure, $\mathcal{L}_r$ is to the loss associated with learning relations and $\mathcal{L}_j$ is the loss associated with joint learning. Additionally, L2-norm (as indicated by regularization factor λ) is used to prevent model overfitting.

4.3 Experimental Results and Discussion

4.3.1 Data

To evaluate the effectiveness of the model, experiments were carried out with five different languages. Note that these datasets are the same that were used to evaluate the performance of the segmenter in the previous chapter.

1. The RST-DT corpus (Carlson et al., 2003) in English (eng.rst.rstdt)
2. The Potsdam Commentary corpus (Stede and Neumann, 2014) in German (deu.rst.pcc)
3. The Dutch Discourse Treebank (Redeker et al., 2012) (nld.rst.rstdt)
4. The Cross-document Structure Theory News Corpus (Cardoso et al., 2011) in Portugeuse Brazilian (por.rst.cstn)
5. Basque Discourse Treebank (Iruskieta, Aranzabe, de Ilarraza, Lersundi, and de Lacalle, Iruskieta et al.) (eus.rst.rstdt)

Some statistics on the data present in these corpora are provided in Table 4.2. [1] One can observe that the amount of training data available is very small, particularly for the Dutch and Basque datasets. Likewise, the number of labels to be classified is fairly large.

Table 4.2. Distribution of data in each dataset. #Labels indicate the number of (relation+nuclearity) pairs.

Corpus	#Trees	#Words	#Segments	#Sentences	#Relations	#Labels
eng.rst.rstdt	385	206,300	21,789	8,318	56	110
deu.rst.pcc	173	32,274	2,790	2,193	32	58
nld.rst.nldt	80	27,920	2,345	1,707	31	51
por.rst.cstn	136	27,185	2,589	1,950	32	58
eus.rst.rstdt	85	27,982	2,396	1,660	31	50

4.3.2 Setup and Code

The Adam optimizer (Kingma and Ba, 2014) was used for updating network weights during training. For all experiments, the hidden dimension of the GRU was set to 50: the concatenation of the forward and backward GRUs would therefore result in hidden representations of size 100. Additionally, the first layer in both decoder classifiers used a hidden size of 100.

To tune these hyper-parameters, 10% portion of the training data was held out as validation set: all hyper-parameters were tuned on this validation set. Following previous research (Li et al., 2016), it is assumed that the gold discourse segments are already available while evaluating the parser's performance.

This tool was implemented in PyTorch [2]. The API provided by researchers at HuggingFace[3] was used to fine-tune pre-trained BERT. All experiments were performed using NVIDIA-GTX 1080 Ti GPUs.

[1]The REST API available here was used to convert between different file formats: https://github.com/NLPbox/rst-converter-service

[2]https://pytorch.org

[3]https://github.com/huggingface/transformers

Code snippets for word- and segment-level HANs were shown previously. Combining these together, the final hierarchical model is constructed thus:

```
class HierAttNet(nn.Module):
    def __init__(self, elmo, word_hidden_size, sent_hidden_size,
    batch_size, num_classes,
                 max_sent_length, max_word_length):
        super(HierAttNet, self).__init__()
        self.batch_size = batch_size
        self.word_hidden_size = word_hidden_size
        self.sent_hidden_size = sent_hidden_size
        self.max_sent_length = max_sent_length
        self.max_word_length = max_word_length

        self.word_att_net = WordAttNet(elmo, hidden_size=word_hidden_size)
        self.sent_att_net = SentAttNet(sent_hidden_size, word_hidden_size)

        self.w_1 = nn.Linear(2 * sent_hidden_size, sent_hidden_size)
        self.w_2 = nn.Linear(sent_hidden_size, 2)
        self.rel = LabelClassifier(2 * sent_hidden_size,
        sent_hidden_size, num_classes)
        self.dropout = nn.Dropout(0.1)
        self.word_hidden_state = torch.zeros(2, batch_size,
        self.word_hidden_size)
        self.sent_hidden_state = torch.zeros(2, self.batch_size,
        self.sent_hidden_size)
        self.init_hidden_state()

    def init_hidden_state(self, last_batch_size=None):
        if last_batch_size:
            batch_size = last_batch_size
```

```
        else:
            batch_size = self.batch_size
        self.word_hidden_state = torch.zeros(2, batch_size,
        self.word_hidden_size)
        self.sent_hidden_state = torch.zeros(2, self.batch_size,
        self.sent_hidden_size)
        if torch.cuda.is_available():
            self.word_hidden_state = self.word_hidden_state.cuda()
            self.sent_hidden_state = self.sent_hidden_state.cuda()


    def forward(self, inp: torch.Tensor, do=True):
        torch.autograd.set_detect_anomaly(True)
        if do:
            self.init_hidden_state()
        inp = inp.permute(1, 2, 0, 3)
        tree_list = []
        for tree in inp:
            edu_list = []
            for edu in tree:
                output, self.word_hidden_state = self.word_att_net(edu,
                self.word_hidden_state)
                edu_list.append(output)
            output = torch.cat(edu_list, 0)
            output, self.sent_hidden_state = self.sent_att_net(output,
            self.sent_hidden_state)
            tree_list.append(output.squeeze(0))
        return self.w_2(self.dropout(F.relu(self.w_1
        torch.sum(torch.stack(tree_list), 0))))),
        self.rel(tree_list[0], tree_list[1])
```

4.3.3 Evaluation

The ParseEval metrics (Morey et al., 2018) were used for evaluating the performance of the model. Evaluation is performed along 4 dimensions: (1) Span (S), that measures how accurately the model predicted the structure of the tree, (2) Nuclearity (N), that measures how the accuracy of predicting the nuclearity of a relation, (3) Relation (R) that measures the accuracy of classifying the relations between text spans; and (4) Full (F) that measures the performance associated with predicting both the relation and its nuclearity. Additionally, similar to how dependency parse trees are scored, evaluation metrics can be reported as both unlabelled (ULAS) and labelled attachment scores (LAS). ULAS scores typically do not account for the structure of the tree i.e. who are the parent and children of a node. LAS scoring is stricter as it takes into account the tree's structure i.e. who the parents and children of a node are.

To demonstrate the effectiveness of the proposed solution, results are presented using 4 different encoder styles:

Baseline: This is a simple model that contains a bidirectional GRU as its encoder. The span representations are first embedded using their GloVe vectors (Pennington et al., 2014) and then fed to the GRU that summarizes the contextual information for each word in the span. The output of the last hidden state of the GRU is then fed to the decoder to perform classification.

HAN: This model makes use of the HAN encoder to obtain structure-aware span representations and feeds them to the decoder for further processing. Here as well, GloVe embeddings are used, not BERT to obtain word representations. To train both baseline and HAN, a batch size of 64, a learning rate of 0.001 and the maximum number of iterations of 60 was used to perform a fair comparison.

HAN+BERT: This model uses both the HAN encoder and multilingual BERT to obtain deep contextualized, structure-aware span representations.

HAN+BERT+Joint: Built on top of HAN+BERT, this model jointly captures syntax by learning POS tags and dependency parse trees. To train both HAN+BERT and HAN+BERT+Joint, the initial learning rate was set to 3×10^{-5}, the batch size to 16 and the maximum number of training iterations to 8.

An important distinction between these designs is that the baseline model does not generate structure-aware or contextualized representations, the HAN model generates structure-aware but not contextualized representations. HAN+BERT generates contextualized, structure-aware representations, while HAN+BERT+Joint captures structure, context and syntax. This allows one to isolate the impact of each model component on the parser's performance. Note that the parameters used for training either model are also different. For instance, training Baseline or HAN requires more iterations with a larger learning rate when compared to training HAN+BERT or HAN+BERT+Joint that require very few iterations and a small learning rate. This is done because the former two models need to be trained from scratch, when compared to the latter which only need to be fine-tuned for a few iterations, thanks to BERT.

Table 4.3 provides the experimental results on the datasets for each setting. ULAS scores are reported for each model.

As one can see, leveraging HANs and jointly learning syntax features particularly helped in accurately predicting the tree's structure, as indicated by the span score. This empirically verifies our claim that capturing document structure is useful in accurately predicting discourse trees. Likewise, jointly using HANs and BERT helped improve the accuracy of classifying relations and/or their nuclearity. Best improvements in model performance were

Table 4.3. Experimental results for the five datasets using our model

Dataset	Model	S	N	R	F
eng.rst.rstdt	Baseline	58.2	33.4	22.1	20.9
	HAN	65.3	53.2	41.3	40.6
	HAN + BERT	84.5	64.8	54.9	54.1
	HAN + BERT + Joint	**85.3**	**65.6**	**55.4**	**54.7**
deu.rst.pcc	Baseline	56.3	34.8	13.2	11.8
	HAN	62.1	45.1	34.5	33.6
	HAN + BERT	74.3	56.7	49.7	48.6
	HAN + BERT + Joint	**77.5**	**59.8**	**53.1**	**52.4**
nld.rst.nldt	Baseline	48.9	31.5	18.0	16.5
	HAN	56.7	42.1	31.4	30.6
	HAN + BERT	67.9	53.6	42.5	43.7
	HAN + BERT + Joint	**70.8**	**54.7**	**44.6**	**44.1**
por.rst.cstn	Baseline	51.7	30.4	20.1	19.3
	HAN	58.4	37.6	27.4	26.9
	HAN+BERT	62.3	42.1	31.8	30.9
	HAN + BERT + Joint	**64.1**	**45.6**	**32.4**	**31.7**
eus.rst.rstdt	Baseline	56.2	34.9	18.8	17.5
	HAN	63.4	41.5	31.6	30.9
	HAN+BERT	65.8	44.3	35.7	34.1
	HAN + BERT + Joint	**67.4**	**45.9**	**37.6**	**36.4**

observed with English and German datasets; with joint learning of syntactic features particularly helping German discourse parsing. It is also interesting to note that the performance improvements were not substantial for Basque and Portuguese Brazilian, although this is fairly consistent with the findings of related research work.

It is interesting to note that one can qualitatively analyze the behavior of HANs by looking at the best scoring words and/or segments and the relation classified. By doing this, one can get better insights into predictions made by the model. By leveraging both the contextualized embeddings provided by BERT and the structural knowledge encapsulated by HANs, our model makes better predictions than the baseline. In the next section, an in-depth analysis of the English dataset is provided. Particularly an attempt is made to understand how HANs help in better classifying rhetorical relations between text spans.

4.4 Analysis of Performance on English dataset

This section provides an analysis of the model's performance on the eng.rst.rstdt corpus.

4.4.1 Comparison with existing models

The performance of the model[4] is compared with previous research in Table 4.4.

Table 4.4. Comparison of model performance with existing work. ULAS scores for MRCV19 were not available.

Model	ULAS				LAS			
	S	N	R	F	S	N	R	F
Feature-based parsers								
Feng and Hirst (2014a)	**84.3**	**69.4**	56.9	56.2	68.6	55.9	45.8	44.6
Ji and Eisenstein (2014)	82.0	68.2	**57.8**	**57.6**	64.1	54.2	**46.8**	**46.3**
Neural parsers								
Braud et al. (2016)	79.7	63.6	47.7	47.5	59.5	47.2	34.7	34.3
Li et al. (2016)	82.2	66.5	51.4	50.6	64.5	54.0	38.1	36.6
Mabona et al. (2019)	-	-	-	-	67.1	**57.4**	45.5	45.0
Our Work	**85.3**	65.6	55.4	54.7	**68.7**	54.8	45.1	44.5

As one can see, our parser achieves competitive results that rivals the performance of both feature-based and neural parsers. It should be noted that our parser achieves the best span score when compared to other models.

4.4.2 Analysis of word-level attention

Define an explicit discourse relation as one that is made apparent through the use of cue words or discourse markers. For example, words like 'because' or 'since' are very good indicators of the Cause relation. It can be conjectured that word-level attention should be able to identify such cue words accurately to predict the correct relation. Table 4.5 displays

[4]To evaluate the model, the educe package: https://educe.readthedocs.io/en/latest/rst-dt.html is used

the lemmas associated with words that were assigned the highest attention scores for each relation type.

Table 4.5. Lemmas of highest scoring words grouped by relation

#	Relation	List of words
1	Attribution	{say, declare, ask, tell}
2	Cause	{as, because, since, to}
3	Condition	{if, until, unless}
4	Contrast	{but}
5	Manner-means	{by}
6	Circumstance	{since, then, by}

Clearly, Table 4.5 is able to identify cue words (Carlson and Marcu, 2001) that explicitly signal a discourse relation. However, this may also lead to some confusion between labels. For example, the word 'since' potentially signifies both Cause and Circumstance relation: here the context in which the word occurs plays a crucial role. This context must be captured at the segment level by the HAN encoder. Likewise, the burden of identifying implicit discourse relations lies on segment-level GRUs and attention as these relations are not made apparent through the use of cue words. Here, the segment-level GRU and attention must identify the segment(s) that are most likely indicators of the relation to be classified.

4.4.3 Analysis of segment-level attention

Table 4.6 provides some examples of predictions made on the eng.rst.rstdt corpus. In these examples, the segments that were ascribed the highest attention scores by the model are highlighted. In other words, one is interested in observing if segment-level attention can take representations provided by the word-level GRU and use it to rank segments according to their relevance.

Specifically, segment-level attention is analyzed from 2 perspectives:

1. Given that word-level attention correctly identifies a cue word or phrase as an explicit discourse marker, can the segment-level attention identify its associated segment as the most important one?
2. Given that the relation is implicit and no cue words can be picked up by word-level attention, can segment-level attention identify the best segment(s) that indicate the relation?

Examples #1 and 2 answer question 1, where word-level attention picks up cue words like 'confirmed' and 'as' correctly and segment-level attention picks up the correct segment(s) and classifies the relation correctly. Likewise, examples #3 and 4 answer question 2, where the relation is implicit and word-level attention cannot pick up any cue words (as they do not exist). In example #3, the correct segment is identified and the relation is also predicted correctly. Whereas, in example #4, the model cannot pick up the correct segment(s) as the most relevant one(s) and mistags the relation as EXPLANATION.

To conclude, while word- and segment-level attention work together to identify explicit relations correctly, the burden of identifying implicit relations lies largely on the segment-level attention.

4.5 Conclusions and Future Work

In this chapter, a simple yet effective neural framework was suggested for performing text-level discourse parsing. Experimental results show that the model is able to accurately generate parse trees for various languages; with their performance rivaling that of several feature-based and neural RST parsers. Additionally, it was shown that learning syntactic features helps learn better parse tree structures when compared to baseline models. Also, using HANs allows one to interpret the behaviour of the model and gain insights into what predictions are made by the parser.

Table 4.6. Qualitative analysis of Segment-level Attention

#	Nucleus	Satellite	True Relation	Predicted	Comments
1	that it received permission late Thursday from U.S. antitrust regulators to increase its Jaguar holdings past the $15 million level.	GM **confirmed** Friday	ATTRIBUTION	ATTRIBUTION	The model correctly predicts the relation as ATTRIBUTION by paying maximum attention to the word 'confirmed' that explicitly indicates this relation. Likewise, the first segment in the nucleus receives more attention than the second as it is a better indicator of the relation.
2	"I'm not losing faith in the market," said Boston lawyer Christopher Sullivan	**as** he watched the market plunge on a big screen in front of a brokerage firm.	CIRCUMSTANCE	CIRCUMSTANCE	Word-level attention correctly identifies the keyword 'as' as a good indicator of this relation between the two text spans. Furthermore, the model correctly identifies the context in which this word is used, pays attention to the correct segment and hence correctly classifies the relation.
3	but by the time Solidarity took office in September,	the damage was done	CIRCUMSTANCE	CIRCUMSTANCE	The model correctly predicts the relation as CIRCUMSTANCE by identifying the segment containing the phrase 'by the time' as an implicit indicator of the temporal circumstance relation between the two text spans
4	Are you kidding? Looking for a job was one of the most anxious periods of my life and is for most people.	Your paper needs a reality check.	TOPIC-COMMENT	EXPLANATION	The model incorrectly predicts the relation as EXPLANATION. It picks the wrong segment as the most important one instead of the one before it, that is more relevant and should have been considered for correctly tagging the relation as TOPIC-COMMENT.

There are several potential avenues for future research. In this model, the focus was only on jointly learning POS tags and dependency parse trees purely due to the non-availability of data/corpora for related linguistic tasks like co-reference resolution or semantic relation classification for languages other than English. However, learning such tasks jointly should further benefit discourse parsing, especially with relation classification. Likewise, one may

also want to leverage the output of an RST parser; and particularly use the attention scores given by the HAN model to improve the performance of related linguistic tasks.

CHAPTER 5

TEMPLATES FOR QUESTION GENERATION

Chapters 3 and 4 described models for discourse segmentation and parsing. The output of the parser is a discourse tree that summarizes the text it represents by depicting discourse relations between its spans. This tree can be leveraged to construct a reading comprehension assessment where deep, inferential questions can be posed to test a reader's understanding of the text. This chapter describes a series of syntactic transformations and a template-driven algorithm for obtaining questions from discourse relations.

5.1 Introduction

5.1.1 Problem Statement

Revisiting the problem of question generation (QG), the objective is to transform a discourse tree into question-answer pairs. Recall that a discourse tree depicts multi-level semantic relations that hold in text: relations described by lower rungs of the tree are intra-sentential and low-level compared to those described by higher rungs of the tree that are inter-sentential and more high-level. Hence, questions generated from low-level relations would test the knowledge of semantic relations at the sentence-level while questions obtained from high-level relations would be more general in scope and test the understanding of the text as a whole.

5.1.2 Previous Work

Previous research efforts in QG have primarily focused on transforming declarations into interrogative sentences, or using shallow semantic parsers to create factoid (Wh-type) questions.

Mitkov and Ha (2003) made use of term extraction and shallow parsing to create questions from simple sentences. Heilman and Smith (2010) suggested a system that over-generates questions from a sentence. First, the sentence is simplified by discarding leading conjunctions, sentence-level modifying phrases, and appositives. It is then transformed into a set of candidate questions by carrying out a sequence of well-defined syntactic and lexical transformations. Then, these questions are evaluated and ranked to identify the most suitable one(s).

Similar approaches have been suggested over time to generate questions, like using a recursive algorithm to explore parse trees of sentences in a top-down fashion (Curto et al., 2012), creating fill-in-the-blank type questions by analyzing parse trees of sentences and thereby identifying answer phrases (Becker et al., 2012) or using semantics-based templates (Lindberg et al., 2013; Mazidi and Nielsen, 2014) to obtain question-answer pairs.

The generation of complex questions from multiple sentences or paragraphs was first explored by Mannem et al. (2010). Discourse connectives such as 'because', 'since' and 'as a result' signal explicit coherence and can be used to generate Why-type questions. Araki et al. (2016) created an event-centric information network where each node represents an event and each edge represents an event-event relation. Using this network, multiple choice questions (and a corresponding set of distractor choices) are generated. Olney et al. (2012) suggested using concept maps to create inter-sentential questions where knowledge in a book chapter is represented as a concept map to generate relevant exam questions. Likewise, Papasalouros et al. (2008) and Stasaski and Hearst (2017) created questions using information-rich ontologies.

Of late, several encoder-decoder models have been used in language generation. Yin et al. (2015) and Du et al. (2017) argue that similar models can be used to automatically translate narrative sentences into interrogative ones. Interested readers can refer to a survey paper written by Pan et al. (2019) to know more about neural question generation methods.

5.1.3 Limitations of previous work

Motivations for using discourse relations for generating questions were revealed in Chapter 1. While a lot of work has been done in the field of QG, previously developed systems suffer from important drawbacks that are redressed in this dissertation:

1. Most earlier QG systems create factoid questions from single sentences. Such questions would not be very meaningful in the context of a large document. For instance, a question like 'What is the county seat of Tarrant County?' is a naive question in the context of a Wikipedia article on Tarrant County as this would simply require one to look up the sentence that contains this information and quote the answer 'Fort Worth'.

2. QG systems often focus on the grammatic and semantic correctness of a question as opposed to its difficulty (Gao et al., 2018). Answering a good question requires one to look beyond the clarity and conciseness of the language in which it is framed. It requires the reader to think through and recall previously learned concepts to arrive at the correct response.

3. Recent research in neural QG (Pan et al., 2019) has focused on constructing questions from more than one sentence. While using neural networks has allowed for the creation of syntactically correct inter-sentential questions, these questions focus on simpler aspects of inter-sentential phenomena such as coreference resolution and sentence selection (Yin et al., 2015). It is imperative to generate inferential questions that test the ability to deduce high-level relations like CAUSE, SOLUTIONHOOD, EVIDENCE and so on (a complete list of these relations was provided in Table 1.4.1).

5.1.4 Dissertation Contributions

To remedy the drawbacks of previous systems, this dissertation suggests using two sources of information, namely a cognitive taxonomy (Bloom, 1964) and discourse theory (Mann and

Thompson, 1988) to devise a model that generates meaningful, inter-sentential questions. Characteristics that set this system apart from previous research include:

- As opposed to generating questions from sentences, it generates questions from entire paragraphs and/or documents. Generation of inter-sentential questions was rarely explored by previous QG systems and this system ranks among the few, handful ones that can generate such questions (Pan et al., 2019).

- The system leverages discourse relations to craft questions. It looks beyond inter-sentential phenomena like coreference resolution and sentence selection to generate meaningful questions that test the ability to deduce semantic relations in text. In fact, questions generated by the system indirectly test the knowledge of coreference resolution as it is often required to uncover discourse relations in text (Kishimoto et al., 2020).

- Generated questions require readers to write detailed responses that may be as long as a paragraph. Datasets like SQuAD (Rajpurkar et al., 2016) often require readers to give the answer as a span of short tokens as found in the text. This makes such questions often trivial as it simply requires the reader to identify which section of the passage contains the required information and simply highlight or quote the response. Writing a long response, on the other hand, requires some thinking and good knowledge of grammar.

- Designed templates are robust. Unlike previous systems that work on structured inputs such as sentences or events, this system should work around any type of input. Experimental results shown in Table 5.5 show that generated questions are grammatically correct and of good quality.

- Systems such as Mannem et al. (2010) make use of cue-words i.e. explicit relations to generate questions. For instance, a cue word such as 'because' is a very likely indicator of the CAUSE relation. Such relations are easier to realize than implicit relations that do not use cue words. The system described in this chapter uses both implicit and explicit relations (Taboada, 2009) to generate questions.

5.2 Approach

The text from which questions are to be generated goes through the mini-pipeline shown in Figure 5.1. A detailed description of each stage of the pipeline is provided in subsequent subsections.

5.2.1 Data Preparation

Here the discourse tree associated with the document is input to the system. All relevant nucleus-satellite pairs are extracted and represented as the triple: RELATION (NUCLEUS, SATELLITE). It should be noted that this works only for mono-nuclear relations (explained in Chapter 1). For multi-nuclear relations, it is advised to convert the subtree into a right-branching binary tree and thereby convert an n-ary relation into a set of binary relations (Morey et al., 2018). This system uses only mono-nuclear relations for question generation.

Prior to applying any syntactic transformations on the text spans, all leading and/or trailing conjunctions, adverbs and infinitive phrases are removed from the text span. Further, if the span begins or ends with transition words or phrases like 'As a result' or 'In addition to', these are removed out as well.

The inherent nature of discourse makes it difficult to interpret text spans as coherent pockets of information. To facilitate the task of QG, text spans containing only one word are ignored. Further, in several cases, it was observed that the questions make more sense

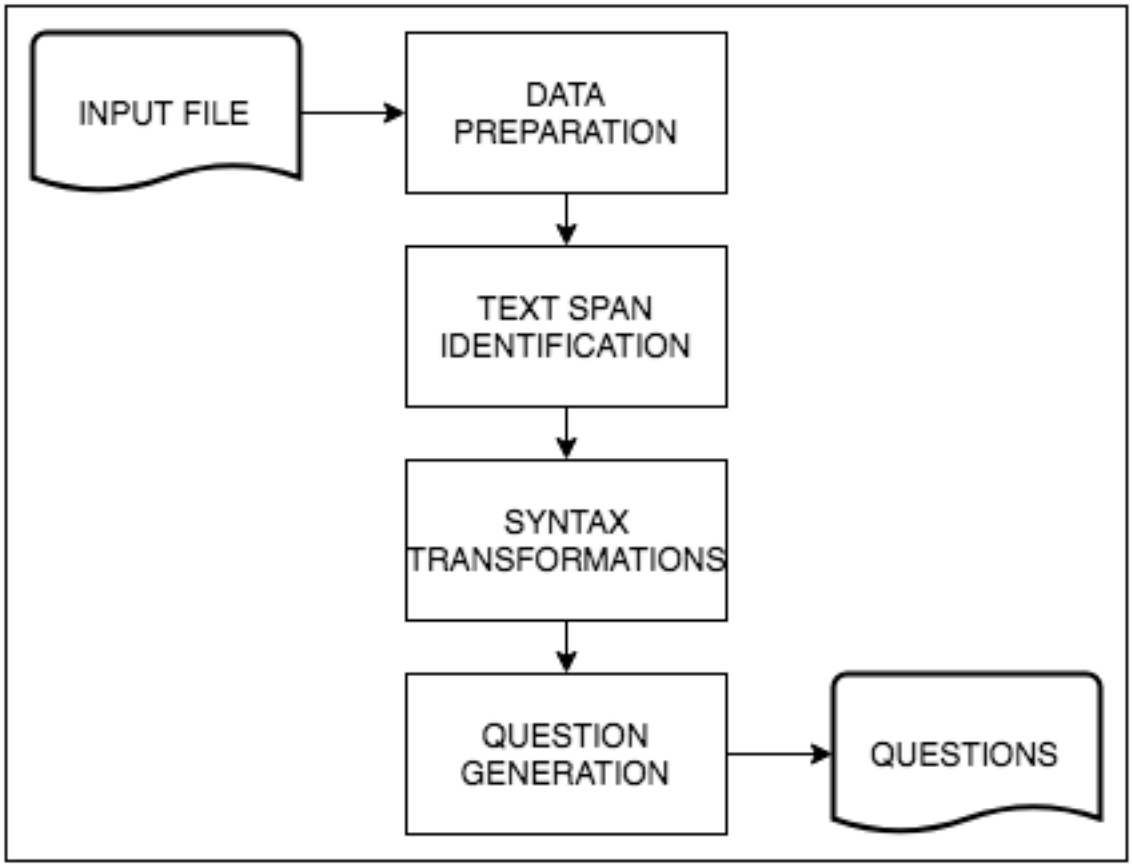

Figure 5.1. A mini-pipeline showing transformations carried out on a document to construct questions.

if coreference resolution is performed: this task was first carried out using automated coreference resolution systems. However, the performance of these systems was poor and the questions generated were vague. Hence this task was performed manually by a pair of human annotators who resolved all coreferents by replacing them with the concepts they were referencing. Two types of coreference resolution are considered: event coreference resolution (where coreferents referring to an event are replaced by the corresponding events) and entity coreference resolution (where coreferents referring to entities are replaced by the corresponding entities). Also, to make the questions sound more natural, some words were randomly

picked from the question and replaced by their synonym (Araki et al., 2016) by consulting WordNet (Glover et al., 1981).

5.2.2 Text-span Identification

Next, each text span is associated with a TYPE depending on its syntactic composition. The assignment of Types to the text spans is independent of the coherence relations that hold between them. Table 5.1 describes these Types with relevant examples.

Table 5.1. Text span Types with relevant examples

Span type	**Characteristic of span**	**Example**
Type 0	A group of many sentences	A bomb exploded in the building. It destroyed its installations.
Type 1	One sentence, or a phrase or clause not beginning with a verb, but containing one	The bomb destroyed the building.
Type 2	Phrase or clause beginning with a verb	destroyed the buildings
Type 3	Phrase that does not contain a verb	destruction of the building

5.2.3 Syntax transformations

If the text span is of Type 1 or Type 2, its parse tree is analyzed to perform a set of simple surface syntax transformations and convert it into a form suitable for QG. First, a dependency parser is employed to find the principal verb associated with the span, its part-of-speech tag and the noun or noun phrase it is modifying. Then, according to the obtained information, a set of syntactic transformations is applied to alter the text. Figure 5.2 describes these transformations as a flowchart. No syntactic transformations are applied on text spans of Type 0 or Type 3. Questions are directly crafted such text spans via templates.

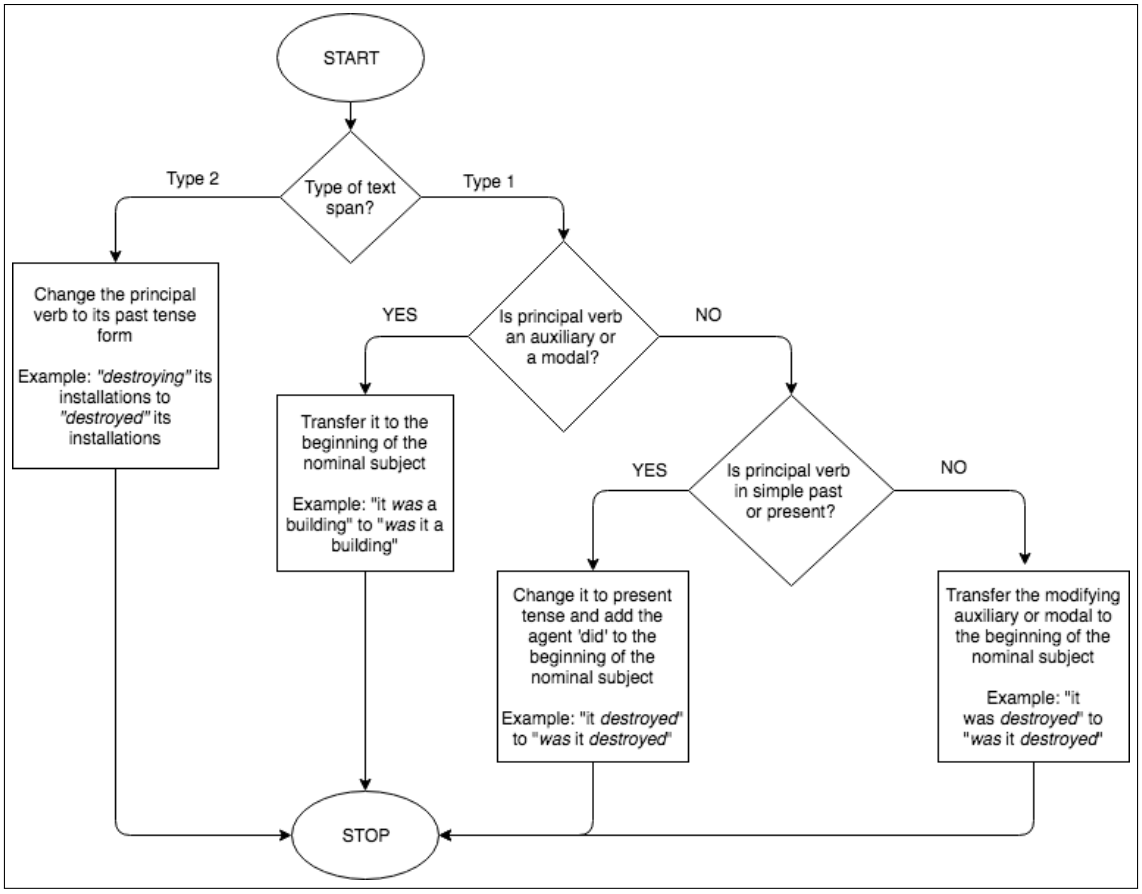

Figure 5.2. Syntactic transformations applied on text spans. These transformations convert the spans to a form suitable for QG.

5.2.4 Question Generation

The transformations described in Figure 5.2 yield a text form suitable for QG . A template is applied to this text to formulate the final question. Table 5.2 defines some of these templates. The design of the chosen templates depends on the relation holding between the spans, without considering the semantics or the meaning of the spans. Note that only 9 relations have been considered for generating questions: other relations such as ELABORATION were considered too trivial or TOPIC-DRIFT were considered too complex for the QG task.

It must be noted that all transformation described upto Section 5.2.2 can be applied to any text span regardless of which domain or relation it is a part of. This makes the system

generic and thereby scalable to any domain as one would only have to define new templates for each relation type.

To understand how the system operates, consider the discourse tree from Figure 1.6. This example shows how the system will generate questions for the causal relations that have been isolated in Figure 5.3.

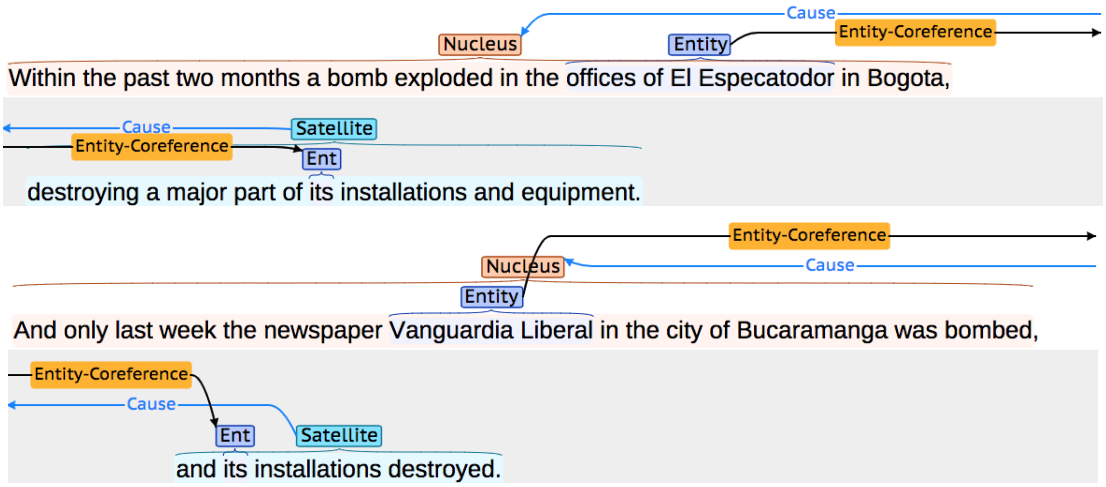

Figure 5.3. Examples of a cause relation from the document

For the first example, begin by associating the satellite: "destroying a major part of its installations and equipment" with Type 2. The principal verb 'destroying' is changed to past tense form 'destroyed' and the pronoun 'it' is replaced by the entity it is referencing i.e. 'the offices of El Especatador', to obtain the question stem: 'destroyed a major part of the installations and equipment of the offices of El Especatador'. The system then uses the template for the cause relation for Type 2 to obtain the question: "What destroyed the installations and equipment of the offices of El Especatador?".

For the second example, begin by associating the satellite: "its installations destroyed" with Type 2. The principal verb is 'destroyed' and the pronoun 'its' is replaced by the entity it is referencing i.e. 'Vanguardia Liberal', to obtain the question stem: 'destroyed installations

Table 5.2. Templates for Question Generation

Relation	**Template for type 0**	**Template for type 1**	**Template for type 2**	**Template for type 3**
EVIDENCE	What evidence is provided to support [Nucleus]?	Why [Nucleus]?	What [Nucleus]?	What caused [Nucleus]?
EXPLANATION	How do we know that [Nucleus]?	Why [Nucleus]?	What [Nucleus]?	What caused [Nucleus]?
BACKGROUND	[Nucleus]. Under what circumstances does this happen?	Under what circumstances [Nucleus]?	What circumstances [Nucleus]?	What circumstances led to [Nucleus]?
CAUSE	[Satellite]. Explain the reason for this statement.	Why [Satellite]?	What [Satellite] ?	Explain the reason for [Satellite]?
CONDITION	[Nucleus]. Under what conditions did this happen ?	Under what conditions [Nucleus]?	What conditions [Nucleus] ?	What conditions led to [Nucleus]?
EVALUATION	[Nucleus]. What lets you assess this fact?	What lets you assess [Nucleus]?	What assessment [Nucleus]?	What assessment can be given for [Nucleus]?
MEANS	How is it that [Nucleus]?	How [Nucleus]?	In what manner [Nucleus]?	How did [Nucleus] happen?
RESULT	[Nucleus]. Explain the reason for this statement.	Why [Nucleus]?	What [Nucleus] ?	Explain the reason for [Nucleus]
SOLUTIONHOOD	[Nucleus]. What is the solution to this problem?	What is the solution to [Nucleus]?	What solution [Nucleus]?	What is the solution to the problem of [Nucleus]?

of Vanguardia Liberal'. The system then uses the template for the cause relation for Type 2 to obtain the question: "What destroyed the installations of Vanguardia Liberal?".

5.2.5 Representative example

A representative example showing a large passage, the associated discourse relation triples and generated questions has been provided in Figure 5.3. It can be seen that questions generated are of varying scopes and sizes, testing on a high-level the understanding of the complete passage and its constituent paragraphs.

5.3 Experimental Results

5.3.1 Data

For the purpose of experimentation, the English RST-DT (eng.rst.rstdt) corpus (Carlson et al., 2003) is used that contains 385 annotated Wall Street Journal articles. Each article is associated with a discourse tree that depicts coherence relations holding between its components. These gold trees are directly used to generate questions: purely for the purpose of evaluation. In the absence of discourse trees, one may want to run a discourse parser prior to applying templates. In Chapter 7, a detailed study of the complete pipeline is provided with respect to the SQuAD dataset (Rajpurkar et al., 2016).

5.3.2 Implementation Details

Part-of-Speech tagging and Dependency parsing were performed using the Stanford CoreNLP package (Manning et al., 2014). Conversion between verb forms was carried out using the powerful linguistics library provided by NodeBox [1]. It is important to note that for the purpose of evaluation, the system uses a heavily annotated corpus with several additional

[1]https://www.nodebox.net/code/index.php/Linguistics

Passage:

Taiwan's USI Far East Corp., a petrochemical company, initialed the agreement with an unidentified Japanese contractor to build a naphtha cracker, according to Alson Lee, who heads the Philippine company set up to build and operate the complex. Mr. Lee, president of Luzon Petrochemical Corp., said the contract was signed Wednesday in Tokyo with USI Far East officials. Contract details, however, haven't been made public. The complex is to be located in Batangas, about 70 miles south of Manila.
The proposed petrochemical plant would use naphtha to manufacture the petrochemicals propylene and ethylene and their resin derivatives, polypropylene and polyethylene. These are the raw materials used in making plastics.
The contract signing represented a major step in the long-planned petrochemical project. At an estimated $360 million, the project would represent the single largest foreign investment in the Philippines since President Corazon Aquino took office in February 1986. It also is considered critical to the country's efforts to both attract other investment from Taiwan and raise heavy industry capabilities. The project has been in and out of the pipeline for more than a decade.
However, workers can't break ground until legal maneuvers to block the complex are resolved, moves which caused the signing to remain questionable up to the last moment. As previously reported, a member of the Philippines' House of Representatives has sued to stop the plant. The legislator, Enrique Garcia, had actively backed the plant, but at the original site in his constituency northwest of Manila. The country's Supreme Court dismissed the suit, but Mr. Garcia late last month filed for a reconsideration.
In addition, President Aquino has yet to sign into law a bill removing a stiff 48% tax on naphtha, the principal raw material to be used in the cracker.
However, at a news conference Thursday, Mrs. Aquino backed the project and said her government was attempting to soothe the feelings of residents at the original site, adjacent to the government's major petroleum refinery in Bataan province. "We have tried our best to tell the people in Bataan that maybe this time it will not go to them, but certainly we will do our best to encourage other investors to go to their province," Mrs. Aquino told Manila-based foreign correspondents.
The project appeared to be on the rocks earlier this month when the other major partner in the project, China General Plastics Corp., backed out. China General Plastics, another Taiwanese petrochemical manufacturer, was to have a 40% stake in Luzon Petrochemical.

Relation Triples:

Relation	Nucleus	Satellite
Background	The project appeared to be on the rocks earlier this month	when the other major partner in the project, China General Plastics Corp., backed out. China General Plastics, another Taiwanese petrochemical manufacturer, was to have a 40% stake in Luzon Petrochemical.
Background	As previously reported, a member of the Philippines' House of Representatives has sued to stop the plant.	The legislator, Enrique Garcia, had actively backed the plant, but at the original site in his constituency northwest of Manila.
Cause	a member of the Philippines' House of Representatives has sued	to stop the plant.
Condition	however workers can't break ground	until legal maneuvers to block the complex are resolved, moves which caused the signing to remain questionable up to the last moment. As previously reported, a member of the Philippines' House of Representatives has sued to stop the plant. The legislator, Enrique Garcia, had actively backed the plant, but at the original site in his constituency northwest of Manila. The country's Supreme Court dismissed the suit, but Mr. Garcia late last month filed for a reconsideration. In addition, President Aquino has yet to sign into law a bill removing a stiff 48% tax on naphtha, the principal raw material to be used in the cracker.
Cause	The proposed petrochemical plant would use naphtha	to manufacture the petrochemicals propylene and ethylene and their resin derivatives, polypropylene and polyethylene. These are the raw materials used in making plastics.

Generated Questions:

Sr. No.	Question
1	Under what circumstances did the project appear to be on the rocks earlier this month?
2	Under what circumstances did Enrique Garcia sue to stop the plant?
3	Why has Enrique Garcia sued?
4	Under what conditions can employees not break ground?
5	Why would the proposed petrochemical plant use naptha?

Figure 5.4. A representative example: the passage and generated questions are shown here

amendments i.e. manually performed coreference resolution to support robust question generation.

5.3.3 Evaluation Criteria

To evaluate the quality of generated questions, the system used a set of criteria that are defined below. Proposed metrics measure both the correctness and difficulty of the question. All the metrics use a two-point scale: a score of 1 indicates the question successfully passed the metric, a score of 0 indicates otherwise.

- Grammatic correctness of questions: This metric checks whether the question generated is only syntactically correct. This metric does not take into account the semantics of the question.

- Semantic correctness of questions: This metric accounts for the meaning of the generated question and whether it makes sense to the reader. It is assumed if a question is grammatically incorrect, it is also semantically incorrect.

- Superfluous use of language: Since the system does not focus on shortening sentences or removing redundant data from the text, generated questions may contain information not required by the student to arrive at the answer. Such questions should be refined to make them shorter and sound more fluent or natural.

- Question appropriateness: This metric judges whether the question is posed correctly i.e. it checks if the question is not ambivalent and makes complete sense to the reader.

- Nature of coherence relation: Coherence relations are classified into two categories: explicit (the relations that are made apparent through using discourse connectives) and implicit (the relations that require a deep understanding of the text). Questions generated through explicit coherence relations are easier to attempt as compared to the ones generated via implicit coherence relations. A score of 1 is assigned to a question generated from an implicit coherence relation and 0 to a question generated from an explicit one.

- Nature of question: This metric checks for the nature of generated question: If both the answer and question are derived from the same sentence, a score of 0 is assigned. Otherwise the score will be 1.

- Number of inference steps (Araki et al., 2016): To evaluate this metric, three semantic concepts are considered: paraphrase detection, entity co-reference resolution and event co-reference resolution. For each concept, the score is 1 if it is required to answer the question and 0 if not. The arithmetic mean of these scores is taken to get the average number of inference steps per question.

As an example, consider some relation triples from the RST-DT corpus. Table 5.3 explains how the generated questions evaluate against some of these criteria.

5.3.4 Results and Analysis

Questions were generated for the entire corpus using the QG system. For the 385 documents contained within the RST-DT corpus, a total of 3609 questions were generated. Table 5.4 describes the statistics for the questions generated for each relation type.

It can be observed that a large fraction of generated questions belong to the Explanation and Background relation types. This is largely because these relations occur more frequently as compared to relations like Solutionhood or Evaluation which are more scarce. More importantly, these are corpus (or discourse parser if used) dependent statistics and have no bearing on the question generation system.

For evaluating the system, 20 questions were sampled for each relation type. Table 5.5 summarizes the results obtained for the system against each relation type. This process was carried out by two evaluators who were familiar with the evaluation criteria, and were well versed with the corpus and nature of generated questions. Table 5.5 reports the mean scores, accounting for the assessment done by each evaluator.

Table 5.3. Examples for metric evaluation

Metric type	**Relation**	**Generated Question**	**Evaluation**
Nature of coherence relation	Nucleus: they are going to be in big trouble with unionists over any Jaguar deal. Satellite: If they try to build it somewhere else in Europe besides the U.K., Relation: Condition	Under what conditions are **General Motors and Ford Motor Co.** going to be in big trouble with unionists over any Jaguar deal?	This is an example of an explicit relation, made apparent through the use of discourse connective 'If' in the satellite
Nature of question	Nucleus: **As a result,** Colombia will earn $500 million less from its coffee this year than last. Satellite: The 27-year old coffee cartel had to be formally dissolved this summer. Relation: Result	Why will Colombia earn $500 million less from its coffee this year than last?	Here, both the question and answer are derived from text spans belonging to different sentences. Thus the score assigned will be 1.
Number of inference steps	Nucleus: **Then,** when it would have been easier to resist them, nothing was done Satellite: and **my** brother was **murdered** by the mafia three years ago Relation: Explanation	Why was **the author's** brother **killed** by the mafia three years ago?	The student should be able to correctly resolve the pronoun 'my' to 'the author' and know that 'killed' is a synonym of 'murdered'. Thus two semantic concepts, paraphrase detection and entity co-reference resolution, are tested here.

An analysis of the results reveals that many questions are syntactically and semantically well-formed. Issues commonly arose due to errors made by the parser; and the inability of NodeBox to convert between verb forms. Additionally, in very few cases, the templates designed were unable to handle all text span Types either due to poor design or because

Table 5.4. Statistics for Generated Questions

Relation type	**Fraction of generated questions**
Evidence	0.0476
Explanation	0.22
Background	0.2447
Cause	0.1513
Condition	0.0601
Evaluation	0.0518
Manner-Means	0.0587
Result	0.151
Solutionhood	0.0139

Table 5.5. Average score for the evaluation criteria. Here R1: Explanation, R2: Background, R3: Solutionhood, R4: Cause, R5: Result, R6: Condition, R7: Evaluation, R8: Evidence, R9: Manner-means. The average scores for each criterion are indicated in the last column.

Evaluation Criteria	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**	**Average**
Grammatical Correctness	0.95	0.94	0.91	0.98	0.98	0.9	0.84	0.95	0.95	0.95
Semantic Correctness	0.95	0.91	0.91	0.88	0.94	0.88	0.8	0.94	0.93	0.93
Superfluity of Language	0.84	0.81	0.77	0.82	0.71	0.9	0.83	0.89	0.84	0.67
Question Appropriateness	0.93	0.83	0.95	0.75	0.78	0.87	0.6	0.9	0.82	0.85
Nature of coherence relation	0.79	0.38	1.0	0.33	0.27	0.22	0.94	0.92	0.91	0.52
Nature of question	0.71	0.37	1.0	0.24	0.24	0.4	0.88	0.75	0.54	0.45
Average # inference steps	0.43	0.46	0.42	0.56	0.39	0.33	0.27	0.49	0.44	0.42

the text span did not follow either definition of the defined Types. For example, some text spans were phrased as questions and some had typographical errors (originally in the text): this led to the generation of unnatural questions. Further, some text spans were arranged in a way such that the main clause appeared after the subordinate clause (For example, the sentence 'If I am hungry, I will eat a cake'): handling such text spans would require us to

modify the text such that the subordinate clause follows the main clause (In this example's case, 'I will eat a cake if I am hungry'). However, to the best of our knowledge, there are no known transformations that allow one to achieve this rearrangement.

Superfluity of language is of concern, as generated questions often contained redundant information. However, identifying redundant information in a question would require a deep understanding of the semantics of the text spans and of the relation that holds between them. Currently, modern discourse parsers are partially capable of handling this aspect (Li et al., 2016) and can perform reasonably well is the nature of relation is explicit.

The latter four metrics depend heavily on the corpus, and not the designed system. Because of its ability to create inter-sentential questions and handle complex coherence relations, the system was given a moderate to good score by both evaluators. Depending on the text and its relations, these scores may vary. Of course, one should expect these scores to increase considerably for a corpus containing many implicit relations between text spans that are displaced far apart in the text.

Table 5.6 provides some statistics on common error sources that contributed to semantic (and/or grammatical) errors in generated questions.

Table 5.6. Common error sources: The percentage of incorrect questions is the ratio of incorrect to total questions with semantic/grammatical errors.

Source of Error	**% of incorrect questions**
NodeBox errors	6.7%
Parsing errors	8.3%
Poor template design	13.3%
Incorrect type identification	6.7%
Clause rearrangement	6.7%
Other minor errors	6.7%

5.3.5 Comparison with previous work

Comparison with previous work is unwarranted largely because these systems focus on performing simple syntactic transformations on single sentences. Nonetheless, to argue in favour

of using discourse relations for QG and understanding their relevance in the context of a big document, a qualitative analysis is carried out with Heilman and Smith (2010).

The quality of proposed system's (abbreviated as QG) questions is compared with Heilman and Smith (2010) (abbreviated as MH) in Table 5.7. To obtain questions using MH, the nucleus-satellite pair is directly input to the system. Questions given by both systems are provided in Table 5.7.

In Table 5.7, both systems generate questions for example# 1 that seem relevant in the context of the CONDITION relation between the Nucleus and Satellite although the wording of questions is different. For example# 2, however, QG generates a more relevant question than MH which asks a fairly trivial question. Examples # 3 and # 4 are inter-sentential relations. MH, due to its inability to work with multiple sentences, ends up asking trivial intra-sentential questions. QG on the other hand recognizes a discourse relation between the nucleus and satellite and generates better quality questions. Clearly, QG is better suited for generating deep, inferential reading comprehension questions when compared to previous research work done.

5.4 Conclusions and future work

There are several avenues for potential research. This dissertation has focused only a subset of relations making up the RST-DT corpus. Templates can also be defined for other relations to generate more questions. Further, Reed and Daskalopulu (1998) argue RST can be complemented by defining more relations or relations specific to a particular domain. One can potentially investigate the effectiveness of encoder-decoder models in obtaining questions from Nucleus-Satellite relation pairs or directly from the document. This should eliminate the need for manually performing coreference resolution and even paraphrasing.

Table 5.7. Comparing questions generated by MH and QG for given nucleus-satellite pairs.

Nucleus	Satellite	Relation	MH	QG
She might want to look into a position at Aetna,	if she was interested in a job that would constantly challenge her.	CONDITION	What might she want to look into if she was interested in a job that would constantly challenge her?	Under what conditions might she want to look into a position at Aetna?
The banking operation had a loss of $8.7 million in the second quarter,	largely because of problem real-estate loans,	CAUSE	What had a loss of $8.7 million in the second quarter?	Why did the banking operation have a loss of $8.7 million in the second quarter?
The government insists that such a possibility is low.	It says that despite loose regulation of the market itself, its longstanding regulation of industry will prevent such crashes.	EXPLANATION	What does the government insist?	Why does the government insist that the possibility of a market crash is low?
The Merc received considerable criticism in 1987.	It was discovered that its compliance director, Kevin P. Conway, who then was responsible for policing the exchange's busy oil and metal pits, "was engaged in other personal business activities on Exchange time," including out-of-state trips.	CAUSE	Who was responsible for policing the exchange's busy oil and metal pits?	Why did the Merc receive considerable condemnation in 1987?

CHAPTER 6

QUESTION DIFFICULTY CLASSIFICATION

Chapter 5 introduced readers to a system that generates questions from texts by using coherence relations. Chapter 5 also proposed some metrics for evaluating the quality of questions generated. While these metrics address how 'meaningful' the question is, they shed little to no light on how difficult it is. This chapter lays the groundwork for defining the difficulty or complexity of a question and describes a simple feature-driven model that labels questions with difficulty levels.

6.1 Introduction

Reading comprehension is widely used in classroom and testing environments to gauge student understanding; it requires a reader to identify high-level semantic relations that hold between text components, and have a deep understanding of the content (Brooks et al., 1977). In order to generate inferential questions for reading comprehension, Chapter 5 described a rule-based system that applies a set of syntactic transformations on relation triples to obtain question-answer pairs. These relations, which include CAUSE, SOLUTIONHOOD, BACKGROUND and so on illustrate how text spans are functionally related to each other.

Questions generated are of varying lengths and scopes, with some derived from implicit coherence relations; and some requiring the reader to give a detailed reply. Unlike factoid questions that expect a simple scan through the text to look for the correct response, these questions are more meaningful and require a deeper understanding of the text.

As a representative example, consider a of text and the questions that follow:

Kidder, Peabody & Co. is trying to struggle back. [Only a few months ago, the 124-year-old securities firm seemed to be on the verge of a meltdown, racked by internal squabbles and

defections.]$_1$ *[Its relationship with parent General Electric Co. had been frayed* ***since*** *a big Kidder insider-trading scandal]*$_3$ ..... *[More than 20 new managing directors and senior vice presidents have been hired since January. The firm's brokerage force has been trimmed and its mergers-and-acquisitions staff increased to a record 55 people ...]* $_2$

Question 1: Why was Kidder on the verge of a meltdown a few months ago?

Question 2: How has Kidder tried to fight back following the issues it was facing months ago?

Question 3: What frayed the relationship between General Electric Co. and Kidder?

Here, Question 1 is an example of an intra-sentential question while Question 3 is an example of an inter-sentential question. Question 2 is derived from an explicit relation made apparent by the use of keyword 'since'.

To understand how relatively complex the questions are with respect to each other, a novel feature-driven approach is suggested that automatically classifies these questions into their difficulty levels. A rich set of syntactic and semantic features is considered that takes into account question-answer pairs and contextual information from the passage to perform classification. The model gave an F-score of 0.68 against the corpus.

6.1.1 Measuring Question Difficulty

The task of measuring question difficulty is inherently subjective: perception of difficulty is influenced by various factors (Torgesen, 2004) such as age, reasoning and inferential skills, extent of conceptual knowledge, ability to perform accurate and fluent reading, etc. Further, native speakers find questions easier compared to those for whom the language of the text is a second language (Van Gelderen et al., 2004). It was also shown (Nunan and Keobke, 1995) that student perception of question difficulty differs from reality: A student may find a

question more demanding if they are intimidated by the task and do not put in appropriate effort to attempt it. Likewise, the question may seem apparently easy if they incorrectly assume an aspect to be the task's key aspect.

Despite the psychological barriers to question difficulty analysis, several techniques have been proposed over the years to measure question complexity. One approach suggested classifying questions via the scope of the document from which they are generated (Mannem et al., 2010): a general level question focuses on almost the entire paragraph, a medium level question concentrates on multiple clauses or sentences, and specific level questions are derived from single sentences.

Likewise, psychometrics (the discipline of study in psychology and education concerned with testing, measurement, assessment and related activities) suggests that the difficulty of a multiple-choice question can be statistically gauged by the proportion of test takers who answered it correctly: the value is estimated empirically by supervising a study before the actual test (Holland and Thayer, 1985) is conducted.

6.1.2 Models for Measuring Question Difficulty

Several statistical models have been built over the years to predict question difficulty. Linear SVMs were used (Yahya and Osama, 2011) to automatically classify questions into six classes. It used an expert-curated dataset and a set of lexical features to achieve an accuracy of 0.85. Another study proposed using associative cellular neural network (Namba, 2012) to classify questions from a Java programming course into three levels: easy, standard and difficult. Similar features were used (Hutzler et al., 2014) to design an automated ranking system for an Intelligent Tutoring System. A regressor that makes use of textual features was built (Sheehan et al., 2013) that measures the difficulty of listening comprehension items. The use of word-embeddings with a CNN-based neural network was also suggested (Hsu et al., 2018) to estimate the difficulty of multiple-choice questions.

Previous research work done in question difficulty prediction has focused merely on extracting naive textual features from the question: work described in this document is significantly different from these approaches as it takes into account both the question, the answer and its context in the paragraph for measuring how complex the question is: this allows one to achieve better accuracy in the classification task.

6.2 Problem Definition and Dataset

Formally, define the problem of question difficulty classification as follows:

Given a set of question-answer pairs $(q_i, A_i) \in Q$ generated for a document d, where the answer A_i is obtained from d; and each question is associated with a difficulty level $c_j \in C = \{1, 2, 3\}$, build a model that approximates the function $c_j = f(q_i, A_i)$. The smaller the value of the difficulty level associated with a question, the easier it is.

6.2.1 Refining Questions

Templates described in Chapter 5 were used to craft questions. The system parses through the document to identify coherence relation pairs and applies a set of syntax transformations to convert them into questions. Templates are defined for different types of coherence relations as listed in Appendix B, etc.

To create a training corpus, 125 documents were sampled from the RST-DT dataset and questions were generated using coherence relations for each document. Out of the 1109 questions generated, 894 questions (with/without modification) were included in the final dataset.

Due to the arbitrary nature of discourse, some of the generated questions were erroneous. Common reasons for this included grammatical and/or semantic incorrectness, redundancy in question due to superfluity of language, ambiguity in question meaning, etc. Likewise, some questions made no sense, and some were semantically identical to others. An evaluation

of the questions revealed that 30% of the generated questions had extraneous use of language and 15% of the questions were ambiguous. 9% of the questions were semantically incorrect: an investigation of the sources of errors revealed the major reasons to be parsing errors, inability to handle direct speech, subordinate clause - main clause rearrangement, etc.

To reduce the number of instances of questions having poor quality, some of the generated questions were modified to make them sound more fluent. In general, it was observed that inter-sentential questions had lots of superfluity as templates designed did not account for this. Direct assessment of generated questions showed that 55% of the generated questions had to be modified while 19% of the questions were discarded because they made no sense or were semantically similar to other generated questions.

6.2.2 Annotating Questions

The task of question annotation was performed by a team of 2 annotators who carefully perused each question and gave it a difficulty rating. To increase the κ value, it was ensured that all annotators belonged to the same age group and spoke English as a second language (Van Gelderen et al., 2004).

Thus, each question $q_i \in Q$ associated with document d is associated with a difficulty level c_i on a scale of $1 - 3$. The dataset finally contains the documents, the coherence relation tuples, the question derived from the relation tuples and a difficulty rating. Table 6.1 provides some statistics on the corpus and the inter-annotator agreement. To measure the inter-annotator reliability, Cohen's kappa and Pearson coefficient measures were used. A reasonable agreement of $\kappa = 0.91$ and $\rho = 0.89$ was achieved.

6.2.3 Example

As an example, consider the same example that was given in Chapter 1. The article has been shown once again in Figure 6.1 for better readability. Each question is associated with a difficulty rating.

Table 6.1. Corpus and inter-annotator agreement statistics

Dataset and Question Sizes	
No. of documents	125
No. of Questions selected	894
Average No. of words per Question	12.65
Class Distribution	
Ratio of questions with class 1	0.44
Ratio of questions with class 2	0.37
Ratio of questions with class 3	0.19
Inter-Annotator agreement	
Cohen's kappa κ	0.91
Pearson coefficient ρ	0.89

6.3 Classification

6.3.1 Models for classification and Implementation

Three classifiers: 1. Logistic Regression with L2 regularization, 2. Linear SVM and 3. Random Forest Classifier with 100 decision tree estimators were used for training. Three different classifiers have been used to verify if the effect of features on classification accuracy is independent of the classifier used.

Thee scikit-learn library [1] is used to implement the classifiers. Classification is performed using 10-fold cross validation: micro-averages of the performance metrics are reported for each classifier and feature vector.

6.3.2 Enhancing the baseline

As a baseline, bag-of-words (Yahya and Osama, 2011) and tf-idf representation of questions are chosen as baselines. An analysis of the feature space showed that the classifier ended up giving more importance to irrelevant words such as 'farmer', 'arena' and 'bullock' which could potentially hurt the accuracy. To identify important words that could improve classification

[1] http://scikit-learn.org/

Passage:

Mobil Corp. is preparing to slash the size of its work force in the U.S., possibly as soon as next month, say individuals familiar with the company's strategy. The size of the cuts isn't known, but they'll be centered in the exploration and production division, which is responsible for locating oil reserves, drilling wells and pumping crude oil and natural gas. Employees haven't yet been notified. Sources said that meetings to discuss the staff reductions have been scheduled for Friday at Mobil offices in New Orleans and Denver.
Mobil's latest move could signal the beginning of further reductions by other oil companies in their domestic oil-producing operations. [In yesterday's third-quarter earnings report, the company alluded to a $40 million provision for restructuring costs involving U.S. exploration and production operations. The report says that the restructuring will take place over a two-year period and will principally involve the transfer and termination of employees in our U.S. operations. A company spokesman, reached at his home last night, would only say that there will be a public announcement of the reduction program by the end of the week.]$_3$ [Most oil companies, including Mobil, have been reporting lower third-quarter earnings, largely as a result of lower earnings from chemicals as well as refining and marketing businesses.]$_4$ Individuals familiar with Mobil's strategy say that [Mobil is reducing its U.S. work force because of declining U.S. output.]$_1$
[Yesterday, Mobil said domestic exploration and production operations had a $16 million loss in the third quarter, while comparable foreign operations earned $234 million.]$_4$ [Industry wide, oil production in this country fell by 500,000 barrels a day to 7.7 million barrels in the first eight months of this year. Daily output is expected to decline by at least another 500,000 barrels next year.]$_2$ Some Mobil executives were dismayed that a reference to the cutbacks was included in the earnings report before workers were notified.

Sample Questions:

No.	Question	Class
1	Why is Mobil Corp. reducing its U.S. work force?	1
2	Evaluate the situation of oil production in the Unites States.	2
3	What structural changes is Mobil Corp. undergoing?	2
4	Why is Mobil Corp. reporting lower third-quarter earnings?	3

Figure 6.1. A representative example from the corpus: the passage, generated questions and their difficulty levels are shown

accuracy, the dataset was tokenized to identify all unique unigrams. Their frequencies of occurrence were computed. To reduce the dependency of classifier on irrelevant words, a list

of most frequently occurring tokens (that are not stop words) was compiled and included in the bag-of-words and tf-idf vectors.

The following tokens were identified as relevant features: *why, what, how, circumstance, condition, evidence, when, solution, cause, result.* As one can see, these words seem to be fairly important as they reveal the intent of the question, for example, a question containing the word 'cause' is most likely to test a student on the cause-effect relationship.

Table 6.2 provides the results for the baseline feature representations. Performance improvements were observed for some representation-model combinations with feature reduction. Small improvements in performance were also observed for categorizing questions into classes 1 and 2; however accuracy of performance for class 3 remained virtually the same. A qualitative analysis of the obtained results revealed that a large fraction of questions that contain the identified keywords belonged to classes 1 and 2. Thus, an improvement in accuracy of classification for these classes was expected. However to see performance gains for class 3, special features such as length of the answer, nature of coherence relation, similarity between question and the sentence(s) from which the question is derived, etc. may be required The next sub-section describes these features and their effect on performance.

Table 6.2. Performance metrics for all feature representations and classifiers. Here, bow_q: bag-of-words representation of questions, tfidf_q: tf-idf representation of questions, bow_q* and tfidf_q*: corresponding representations with most frequent tokens.

Baselines	**Logistic Regression**			**Linear SVM**			**Random Forest**		
	P	R	F	P	R	F	P	R	F
bow_q	0.45	0.48	0.46	0.44	0.46	0.45	0.46	0.50	0.48
tfidf_q	0.38	0.48	0.42	0.44	0.49	0.46	0.48	0.51	0.49
bow_q*	0.53	0.54	0.53	0.53	**0.55**	0.53	0.53	**0.55**	**0.54**
tfidf_q*	**0.54**	**0.55**	**0.54**	**0.54**	**0.55**	**0.54**	**0.54**	**0.55**	**0.54**

6.3.3 Features

To improve upon existing systems that measure the difficulty of a question, the following features are considered in addition to each of the four representations described in the previous Section:

1. **Question Length**: This is an integer given by the number of words in the question. A longer question is expected to be easier to understand than a shorter one (Cannell et al., 1981).

2. **Count of complex syntactic structures**: The ability to apply appropriate parsing and inference rules to comprehend a question's meaning may depend on the sentence's syntactic structure. Here, a simple count of the number of clauses and prepositional phrases in the question (Cannell et al., 1981) is taken. The Stanford CoreNLP package (Manning et al., 2014) to obtain constituency parses for counting.

3. **Discourse connectives in the answer**: This is a binary value that indicates whether the coherence relation used to derive the question is implicit or explicit. Discourse connectives in the answer such as 'but', 'since', 'as a result', etc. signal explicit coherence (Taboada, 2009). Questions derived from explicit relations are expected to be easier to answer as opposed to those generated from implicit ones.

4. **Similarity ratio between Question and its Source**: Some of the generated questions were summarized versions of sentences from which they are derived. All coreferents were resolved by replacing them with the concepts they were referencing. Further, some required a complete restructuring of the sentence as they were ambiguous. Identifying such semantic relations (coreference resolution, paraphrase detection, etc.) is particularly useful in gauging the breadth of knowledge of a language (Araki et al., 2016). To quantify the semantic similarity between the question and question

stem, the cosine distance between their vector representations was computed and chosen as an important feature.

5. **Nature of question**: This is a binary feature that checks whether the question is derived from multiple sentences; or from a single sentence or clause, allowing one to determine how well one can identify and differentiate between inter- and intra-sentential discourse relations and draw logical connections between ideas that may be displaced far apart from each other in the text.

6. **Nature of answer**: This is a binary feature that checks whether the expected answer is a single sentence or clause; or comprises multiple sentences. Questions that require a long response are expected to be more difficult than those requiring short and direct answers. The reason is that many of such questions require readers to interpret evaluations and assessments of opinions, identify multiple causes or evidences of an event or detail a solution to some problem or issue.

Changes in experimental results post adding these features are shown in Table 6.3. When features are appended to each of the baseline feature representations, the performance improves considerably. An important reason for this change was identified as the significant enhancement in F-score for Class 3. Hand-designed features were capable of distinguishing tougher questions from the easier ones as now the system recognizes the nature of coherence relations, questions and answers; and the paraphrasing of sentences yielding differently worded questions. The best F-score was observed for the linear SVM classifier with bag-of-words model containing the most frequent tokens. Additionally, to argue for the potency of these features, effectiveness of each of them is investigated by incorporating them one at a time into the model: Table 6.4 shows the results for the best feature-classifier combination.

Table 6.3. Performance metrics for all feature representations: Model abbreviations are identical to those given in Table 6.2.

Features	**Logistic Regression**			**Linear SVM**			**Random Forest**		
	P	R	F	P	R	F	P	R	F
bow_q_f	0.62	0.63	0.62	0.62	0.63	0.63	0.65	0.64	0.64
tfidf_q_f	**0.67**	**0.68**	0.67	**0.67**	0.67	0.67	0.63	0.64	0.64
bow_q_f*	0.66	0.67	0.66	0.67	**0.68**	**0.68**	0.62	0.62	0.62
tfidf_q_f*	0.64	0.65	0.65	0.64	0.65	0.64	0.63	0.64	0.63

Table 6.4. Improvements in F-scores for each class: each cell indicates how the F-score increased with the incorporation of features: Here QL: Question Length, PC: Count of complex syntactic structures, DC: Presence of discourse connectives, CR: Similarity ratio between the question and its source, NQ: Nature of question; and NA: Nature of answer

Features	**Precision**	**Recall**	**Type F1**
Baseline	0.53	0.55	0.53
+QL	+0.02	+0.00	+0.01
+PC	+0.005	-0.01	+0.005
+DC	+0.05	+0.05	+0.05
+CR	+0.03	+0.02	+0.03
+NQ	+0.03	+0.02	+0.03
+NA	+0.02	+0.01	+0.02

6.3.4 Qualitative Analysis

Table 6.5 highlights some of the question-answer pairs and how they fared against models. Each of the mentioned example explain how the feature-driven classifiers are able to identify the properties of question-answer pairs such as nature of question, nature of answer, nature of relation, paraphrasing, etc. to correctly predict the class value of a question as opposed to baselines that misclassify them.

6.3.5 On Differentiating between Classes 2 and 3

It was observed that the scores for class 3 were generally lower than those for labels 1 and 2. Some reasons for this poor performance are:

Table 6.5. Qualitative Analysis of Results.

#	Label	Question	Expected Answer	Comments
1	2	How has Kidder tried to struggle back following the issues it was facing months ago?	More than 20 new managing directors and senior vice presidents have been hired since January. The firm's brokerage force has been trimmed and its mergers-and-acquisitions staff increased to a record 55 people	All baselines misclassified this as 1 probably because they did not realize the answer is a detailed one. However, with features, all classifier correctly classified it.
2	2	Under what conditions could Kidder's hiring binge backfire?	Kidder's hiring binge involving executive-level staffers, some with multiple-year contract guarantees, could backfire unless there are results.	The baselines classified this as 3. However, the features correctly identified the presence of 'unless' in the answer stem and predicted this to be an easier question.
3	2	What refueled speculation that Kidder is getting out of the brokerage business?	Mr. Carpenter this month sold off Kidder's eight brokerage offices in Florida and Puerto Rico to Merrill Lynch & Co.	The baselines predicted the class as 1. However, the question was a summarized version of a much larger nucleus in the relation. Features used the cosine similarity to correctly predict the class as 2.
4	3	How has GE capital started to exploit the synergy between itself and Kidder Peabody?	The Kidder units have worked on 37 investment banking deals this year	Both the baselines and feature-driven classifiers erroneously classified this as class 2.

1. The class-distribution is skewed. As seen in Table 6.1, the dataset contains many examples from classes 1 and 2, however only 19% were from class 3. The classifiers probably did not have enough data to perform reasonably well.

2. Models found it difficult to differentiate between classes 2 and 3. While both classes differed from class 1 in the sense that either most questions were inter-sentential or

were derived from implicit relations; annotators revealed that differentiating between classes 2 and 3 was challenging as it required them to make several judgement calls such as differentiating between how deep the semantics of a relation is.

For example, in Table 6.5, example 4 shows an instance of misclassification: models misclassified this as class 2. Annotators revealed that they had classified this question as class 3 because not only was the question inter-sentential and the relation implicit in nature, but it required a much deeper understanding of the text to arrive at the answer: designed features probably could not to capture this effectively.

6.4 Conclusions and Future Work

Main contributions in this chapter can be briefly summarized as follows:

1. A novel reading comprehension corpus is released in which questions are annotated with a difficulty level.

2. A feature-driven classifier is presented that classifies questions according to their difficulty level. A rich set of semantic features is used to perform this task.

3. As opposed to considering factoid questions, high-level meaningful questions are considered that test a student's understanding of discourse coherence and semantics.

There are several avenues for further research. Reading comprehension questions for generic documents have been considered to make up the corpus: one can consider other sources of data also such as science textbooks, political discourse, medical literature, etc. Likewise, there are techniques that use sources of information other than coherence relations to generate questions from discourse: such questions can also be analyzed for complexity. Additionally, it would also be interesting to test the efficacy of word embeddings and neural networks in performing question difficulty analysis.

CHAPTER 7

PIPELINE IN ACTION

So far, this dissertation has been detailing on discourse parsing, question generation and questions difficulty classification. Recall that these can be performed one after the other i.e. put together as modules in a pipeline to construct a reading comprehension quiz as shown in Figure 1.4. While previous chapters observed and evaluated these modules in isolation, it would be interesting to note how effective the pipeline is in its entirety on a fresh, raw document.

This chapter provides details on experiments performed with the pipeline on the SQuAD dataset (Rajpurkar et al., 2016). It is empirically shown that the pipeline can generate more meaningful questions than the ones already present in the corpus. Not only will this help teachers as a useful tool for automated question generation, but also aid researchers interested in question answering and reading comprehension.

7.1 Experimentation

7.1.1 Data

The Stanford Question Answering dataset or SQuAD (Rajpurkar et al., 2016) is a popular corpus that is used in reading comprehension based question answering research. It consists of 536 Wikipedia articles with more than 100,000 associated questions. To create the corpus, Mechanical Turk crowd-workers were asked to manually craft questions based on paragraphs of text from the articles. Workers were instructed to use their own words while constructing questions. Later, another group of workers was asked to provide answers (as spans of tokens in the text) to these questions.

An example of a paragraph from the corpus and associated questions is provided in Figure 7.1.

Passage:

The Amazon rainforest also known in English as **Amazonia or the Amazon Jungle**, is a moist broadleaf forest that covers most of the Amazon basin of South America. This basin encompasses 7,000,000 square kilometres (2,700,000 sq mi), of which **5,500,000 square kilometres** (2,100,000 sq mi) are covered by the rainforest. This region includes territory belonging to **nine** nations. The majority of the forest is contained within Brazil, with 60% of the rainforest, followed by Peru with 13%, Colombia with 10%, and with minor amounts in Venezuela, Ecuador, Bolivia, Guyana, Suriname and French Guiana. States or departments in **four** nations contain "Amazonas" in their names. The Amazon represents over half of the planet's remaining rainforests, and comprises the largest and most biodiverse tract of tropical rainforest in the world, with an estimated 390 billion individual trees divided into 16,000 species.

Questions:

No.	Question
1	**Which name is also used to describe the Amazon rainforest?**
2	**How many square kilometers of rainforest is covered in the basin?**
3	**How many nations control this region in total?**
4	**How many nations contain "Amazonas" in their name?**

Figure 7.1. A representative example from the SQuAD dataset

Reasons for choosing SQuAD include:

1. SQuAD is a benchmark dataset used to report results in Question Answering research and consists of more than 100,000 manually generated questions. One can compare questions generated by the pipeline with the ones already present. One can hope to empirically show that the pipeline generates deep, inferential questions and are of better quality than the manually crafted ones.

2. SQuAD consists of expository texts i.e. texts that can be used to instruct or educate people which makes it suitable for pipeline evaluation. On the other hand, the RST-DT corpus (Carlson and Marcu, 2001) that was used to train and evaluate the parser contains Wall Street Journal news articles that may not necessarily be purposeful.

3. Compared to RST-DT, SQuAD contains a lot bigger documents with more sentences and tokens. Table 7.1 provides some lexical information on the two corpora. The Stanford CoreNLP package (Manning et al., 2014) was used to obtain this information. As one can see, the SQuAD dataset has more documents that are bigger in size (almost 10 times) with respect to number of sentences and tokens per document.

Table 7.1. A comparison of the SQuAD and RST-DT corpora

Statistic	**RST-DT (Train)**	**RST-DT (Test)**	**SQuAD (Train)**	**SQuAD (Test)**
# Documents	347	38	442	48
# Avg sentences per document	22.73	25.9	219.19	226.67
# Avg tokens per document	564.7	606.21	5833.73	6151.31

Despite being widely used in Question Answering research, SQuAD suffers from the same limitations that were outlined in Chapter 5. These drawbacks include:

1. SQuAD mainly contains sentence-level questions.

2. The corpus contains factoid questions. As one can observe in Figure 7.1, questions asked are fairly trivial and not very meaningful in the context of the entire paragraph (or the document containing that paragraph).

3. Expected answers are a few span of tokens that can be directly found in the text. Du et al. (2017) observed less than 0.17% of questions in the training set require answers that span more than 1 sentence.

These limitations should be redressed by the pipeline. Upon successfully running the entire pipeline, one can hope to generate deep, inferential questions that would be more consequential in the context of the entire paragraph or document. It should be noted that

the pipeline in run on the entire document and not individual paragraphs present in the document. SQuAD requires only the paragraph (not the full document) as context to answer a question. In order to answer questions obtained from the pipeline, the entire document must be taken into account.

7.1.2 Applying the Pipeline

First, the Stanford CoreNLP (Manning et al., 2014) is used to perform basic text processing i.e. tokenization and sentence boundary detection. Then, the entire dataset is lowercased to make it suitable for further processing.

Then, the document progresses through the pipeline by undergoing segmentation and parsing. The discourse trees obtained are fed to the question generation module that spits out question-answer pairs. Finally, the question difficulty classifier ranks them in the increasing order of their complexity.

7.2 Statistics

This section provides statistics on the relation triples extracted and correspondingly, the questions generated. Note that the pipeline is transferred directly, in a zero-shot manner, to the dataset. Therefore, performance metrics like accuracy cannot be reported unless the outputs are manually observed and a human adjudicates them as correct or incorrect. The zero-shot transfer strategy has found successful application in many tasks like event extraction (Huang et al., 2018), visual question answering (Li et al., 2018) among others and should yield good results here as well.

Results on the number of documents, sentences and tokens have already been provided in Table 7.1. Subsequent sections will provide statistics on outputs provided by the segmenter, relation classifier, question generator and difficulty classifier successively.

7.2.1 Discourse Segmentation

Recall that a discourse segmenter identifies elementary discourse units (EDUs) or segments in text that are potential arguments between which relations hold. The segmenter described in Chapter 3 is run on the SQuAD dataset. Note that ensembling is used as described in the chapter to achieve best results. Table 7.2 provides some statistics on the segments obtained from the text.

Table 7.2. Statistics on segments obtained from discourse segmenter

Statistic	**SQuAD (train)**	**SQuAD (test)**
# Documents	442	48
# Avg Segments per Document	542.59	458.54
# Avg Segments per Sentence	1.94	2.02
# Avg Tokens per Segment	13.7	13.41

The statistics shown in Table 7.2 can be compared with the values given for the English RST-DT corpus in Table 3.1 for reference. On an average, the reference dataset has 2.63 segments per sentence and 10.07 words per segment. It can be inferred that the SQuAD dataset has longer and fewer segments (per sentence). However, as the SQuAD dataset is bigger in size, it has more segments per document than the RST-DT corpus.

7.2.2 Discourse Parsing

After obtaining segments from the previous module, a discourse parser is used to induce a discourse tree and subsequently extract relation triples from the tree. Table 7.3 provides statistics on key relations identified by the discourse parser.

A total of 187,838 and 21,962 relation triples were extracted in total for the train and test portions of the SQuAD dataset respectively. Out of these, 9.74% and 10.16% of the

Table 7.3. Statistics on relation triples obtained from discourse parser: only the ones required by the question generation tool are shown here

Relation	SQuAD (Train)	SQuAD (Test)
All Relations	**187,838**	**21,962**
Evidence	2	-
Explanation	1,606	209
Background	3,754	421
Cause	9,144	1,126
Result	67	11
Condition	647	111
Manner-means	1,667	205

relation triples are meaningful as these will be used by the question generation module for constructing question-answer pairs. It should be noted that the parser could not find any Solutionhood and Evaluation relations in the dataset. This may be attributed to 2 possible reasons:

1. These relations occur rarely in text. Since the parser is working with Wikipedia articles, one can suspect the frequency of such relations to be very small to almost zero.

2. Since these relations occur rarely in text, it is probable that the parser never learned a function for them in the first place. To test this conjecture, the output provided by the parser on both the train and test datasets of the RST-DT corpus was analyzed. It was found that the parser could find Solutionhood relation triples with an accuracy of 50% in the train set and 23% in the test set. Likewise, the parser could find Evaluation relation triples with a slightly greater accuracy of 56% in the train set and 31% in the test set. More importantly, the parser often confused these relations for Elaboration, a trivial relation that occurs very frequently. This suggests that the parser needs to be redesigned to appropriately handle cases of less frequent labels.

This completes the analysis of outputs provided by the discourse segmenter and parser. The next step is to transform obtained relation triples into question-answer pairs.

7.2.3 Question Generation

Utilizing the syntactic transformations and templates defined in Chapter 5, discourse trees obtained from the previous step are translated into question-answer pairs. Following Table 7.3, a total of 16,887 questions were generated for the training portion and 2,083 questions were obtained for the test portion of the corpus.

An important difference between the approach used in Chapter 5 and here is that the output of the discourse parser is available in this case. This allows one to make use of the attention weightings given by the HAN model to identify the most important segment(s) for a given text span. This also allows one to use these segment(s) as representatives for large text spans. By doing this, one can hope to see an improvement in the superfluity scores.

To assess these questions, evaluation criteria described in Chapter 5 are used. Here as well, 2 annotators are employed to ascribe a score to the generated questions. Evaluation was performed on 20 randomly sampled documents from the dataset. Table 7.4 tabulates the average scores assigned by the evaluators. For the sake of comparison, scores obtained for the RST-DT corpus are also provided in this Table (these scores are the same as those reported in Table 5.5 and have been included for the sake of clarity).

Table 7.4. Scores assigned by annotators to generated question-answer pairs for each metric defined in Chapter 5.

Relation	**RST-DT**	**SQuAD**
Grammatical Correctness	0.95	**0.96**
Semantic Correctness	0.93	**0.94**
Superfluity of language	0.67	**0.72**
Question Appropriateness	**0.85**	0.75
Nature of coherence relation	0.52	**0.58**
Nature of question	0.45	**0.52**
Average # inference steps	0.42	**0.48**

Table 7.4 reveals that a large fraction of generated questions are grammatically and semantically correct: this further verifies the claim that templates used for crafting questions

are robust enough to handle any type of input and can generate valid questions. However, annotators gave a comparatively low score for question appropriateness. One of the most important reasons identified for this was that the expected answer did not make sense in the context of the question posed: this was likely due to errors made by the parser in classifying a relation between the text spans. Another reason was that since the output of the HAN model is used to isolate most important segments from text spans, it is likely that the HAN misidentified a segment as the more relevant one, resulting in poor appropriateness scores. However, one can hope this score will get boosted considerably once a more accurate parser is available in the future.

As expected, SQuAD generates better-quality questions than the RST-DT corpus. SQuAD gets better scores for nature of coherence relation and question as it contains more implicit discourse relations (relations that are more difficult to realize in the absence of cue words) and longer texts (thereby yielding more inter-sentential questions). Scores assigned by annotators to the pipeline for the SQuAD dataset show a lot of promise and one can hope to advance research in question answering and reading comprehension with this newly created set of questions.

7.2.4 Question Difficulty Classification

Finally, after running the question generation tool and obtaining the reading comprehension quiz, the final step involves ranking or classifying these questions according to their difficulty. To do this, an ensemble of classifiers described in Chapter 6 are utilized to categorize generated questions. Table 7.5 tabulates the ratio of questions grouped by the labels assigned by the classifiers. For the sake of comparison, class distribution for the RST-DT corpus is also provided in this Table (these scores are the same as those reported in Table 6.1 and have been included for the sake of clarity).

Table 7.5. Ratio of questions grouped by the labels assigned by difficulty classifiers. Class distribution for the training set is also included here for the sake of comparison.

Class	RST-DT	SQuAD
1	0.44	0.41
2	0.37	0.34
3	0.19	**0.25**

It is evident from Table 7.5 that more difficult questions are generated from the SQuAD dataset than the RST-DT corpus. This verifies claims that the SQuAD dataset is better-suited than the RST-DT corpus for question generation and pipeline evaluation.

7.2.5 Example

In addition to the metrics described in the previous section, one can also compare questions generated by the system with previous research work done. This study has been skipped purely because such a comparison would be unfair: questions generated by previous systems are intra-sentential and do not use discourse relations as a source of information. One can anticipate that questions generated by the system described in this Chapter will clearly be more superior than the ones suggested by other systems. As an example, consider Figure 7.2 a fairly small paragraph from the SQuAD dataset and questions that follow.

Figure 7.2 reveals that systems like Heilman and Smith (2010) and Du et al. (2017) are able to generate sentence-level questions that are fairly trivial to answer. System described in this document, on the other hand, generates deeper inferential questions that are more difficult to attempt. These questions are inter-sentential and test the ability to deduce discourse relations in text. Answering these questions requires a complete understanding of the text and its meaning.

Some more examples of the output given by the pipeline on paragraphs present in the SQuAD dataset are provided in Appendix C.

Passage:

Kuznets' curve predicts that income inequality will eventually decrease given time. As an example, income inequality did fall in the United States during its High school movement from 1910 to 1940 and thereafter. However, recent data shows that the level of income inequality began to rise after the 1970s. This does not necessarily disprove Kuznets' theory. It may be possible that another Kuznets' cycle is occurring, specifically the move from the manufacturing sector to the service sector. This implies that it may be possible for multiple Kuznets' cycles to be in effect at any given time.

Questions by Heilman and Smith (2010):

No.	Question
1	When does kuznets ' curve predict that income inequality will eventually decrease?
2	Where did income inequality do fall during its High school movement from 1910 to 1940 and thereafter?
3	May it be possible that another Kuznets' cycle is occurring?

Questions by Du et al. (2017):

No.	Question
1	Where did income inequality fall during its high school movement?
2	When did income inequality fall in the us?
3	When did high-school movement begin?

Questions by our system:

No.	Question
1	Under what circumstances did income inequality fall in the US?
2	Why does rise in income inequality not necessarily disprove Kuznets theory
3	Kuznets' curve suggests that income inequality will eventually decrease given time. What evidence is provided in the text to support this claim?

Figure 7.2. An example showing questions generated by previous work and system described in this Chapter.

7.3 Conclusions

Thanks to a deep pipeline, the SQuAD dataset could be complemented with a set of inferential, inter-sentential questions that are more involved than intra-sentential Wh-type

questions. Potential research efforts could be directed towards building automated QA systems that could answer such questions.

While relations described by the RST are generic enough to work with any document or text, domain-specific coherence relations can be defined for areas of interest like Biology, Political Sciences, Math, etc. One can consider defining templates for asking questions around texts written for such domains as well. Likewise, one can also consider other sources of information like images, audio files, videos, etc. for generating questions. The objective should be to generate meaningful questions asked around the semantics of the object under consideration and not ask simple factoids that are fairly trivial to answer.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

This dissertation addressed the tasks of (a) finding rhetorical relations in large texts, (b) using these relations to generate meaningful question-answer pairs and (c) ranking generated questions according to their difficulty. After assembling into a pipeline and running it on the SQuAD dataset, it was empirically shown that generated questions are of better quality than the ones already present or the ones generated by other automated QG systems (Heilman and Smith, 2010; Du et al., 2017).

8.1 Conclusions

Main dissertation conclusions and contributions are summarized below:

1. The construction of text-level discourse parsers is considered a challenging problem for several reasons, such as the sparsity of training data, the inherent nature of discourse, complexity of relation classification, etc. This dissertation did away with most of these challenges by coming up with a deep discourse segmenter and parser that accurately builds discourse trees from text.

    (a) A deep learning model was constructed that leverages BERT's (Devlin et al., 2019) complex structure for performing discourse segmentation. By jointly learning syntactic features and casting the problem as token classification (as opposed to the traditional way of sequence tagging), research described in this document advanced the state-of-the-art and inched closer to human performance (the proposed segmenter achieves an F-score of 96.7 versus human performance of 98.3)

    (b) A deep model leveraging the representational powers of BERT and Hierarchical Attention Networks or HANs (Yang et al., 2016) was designed to obtain syntax,

structure and context-aware representations. These representations were then fed to a shift-reduce parser to perform accurate text-level discourse parsing. Evaluation on benchmark datasets showed that the model rivalled the performance of state-of-the-art feature-driven and neural parsers.

2. Modern automated question generation systems either generate Wh-type questions (Heilman and Smith, 2010), use shallow semantics and/or ontologies for constructing questions (Graesser et al., 2003; Araki et al., 2016; Stasaski and Hearst, 2017) or use deep learning for generating questions from one to few sentences (Du et al., 2017). This dissertation suggested a method for generating inferential, inter-sentential questions that tests the ability to deduce high-level relations between text spans.

3. This dissertation defined what it means for a question to be 'meaningful' by describing a set of metrics that measure its quality. In sharp contrast to measures like grammatical and semantic correctness, these metrics are more helpful in gauging the quality and complexity of a question.

4. A rich semantic feature-driven classifier was described that categorized questions according to their difficulty or complexity. Using these features allows one to capture dependencies that cannot be captured by simple baselines, thereby observing a relative improvement of 26% in model performance.

5. These modules were arranged into a pipeline and applied in succession to a benchmark reading comprehension corpus like SQuAD (Rajpurkar et al., 2016). It was empirically shown that this pipeline can generate questions that are better in quality and complexity in comparison to the ones that are already present. One can hope to advance research in question answering by analyzing such deep, inferential questions.

8.2 Future Work

There are many potential avenues for future work. Each subsection describes how modules making up this dissertation/pipeline can be further enhanced.

8.2.1 Discourse Parsing

This dissertation proposed a simple yet effective deep learning framework that gathered language information from three important lexical sources: syntax, structure and context by using HANs and BERT and jointly learning syntactic features. Potential enhancements that can further improve model performance are:

1. **Joint learning of other tasks**: The model captured syntactic information in a multi-task learning fashion by using information from dependency parse trees. Other sources of information or language tasks can also be considered here. Braud et al. (2016) performed an analysis on importance of related linguistic tasks by jointly learning models for coreference resolution, temporal relation classsification, identifying turns in dialogue, etc. Likewise, Lin et al. (2019) jointly learned models for discourse segmentation and sentence-level discourse parsing. Such sources of information were not considered by this dissertation as these are mostly available for the English language. More importantly, incorporating this information would unnecessarily complicate the model by introducing a lot of parameters into its architecture.

2. **Exploring text hierarchy**: While training the HAN model, two levels of text hierarchy were considered: words and segments. One may also consider other levels of text hierarchy such as characters, sentences, paragraphs, etc. These levels of granularity were explored by Shi et al. (2020) who observed improvements in model performance, especially in the classification of implicit discourse relations.

3. **Utilizing other language models**: The advent of more advanced language models (Zaheer et al., 2020; Brown et al., 2020) that can work with bigger inputs and yield better text representations has significantly enhanced the performance of many complex NLP applications. One may consider the possibility of fine-tuning such models instead of BERT and expect an improvement in performance.

8.2.2 Question Generation

This dissertation argued that modern automated QA systems are not able to generate meaningful, inferential questions as they either focus on single sentences or simple inter-sentential phenomena like coreference resolution. Work described in Chapter 5 was able to do away with such issues by coming up with a bunch of rule-based transformations that convert coherence relation triples into question-answer pairs. Some key limitations of this system and how they can be overcome are described below:

1. **Coreference resolution and paraphrasing**: The designed system relied heavily on automated coreference resolution systems. Unfortunately, these systems could not yield very good results and in most cases, annotators ended up manually resolving coreference. Likewise, in many cases, words were picked from questions at random and replaced by their synonyms to distort the language. Yin et al. (2015) and Du et al. (2017) showed that neural QG systems can be designed such that they are able to automatically resolve coreference resolution and perform paraphrasing to generate coherent, well-formed questions. If the proposed rule-based system were replaced by a neural QG system, then one would not have to worry about coreference resolution and paraphrasing and let the system bear the burden of performing these tasks. This path was not chosen by the dissertation as training a neural QG system requires annotated data and such a corpus that contains inferential, inter-sentential questions is not available. One can hope to use the pipeline described in this document to generate

questions, fix any grammatical or semantic errors and finally use the obtained corpus to train a neural QG system.

2. **Domain adaptation**: This research works with two corpora: the RST-DT corpus (Carlson et al., 2003) and SQuAD dataset (Rajpurkar et al., 2016) that contains articles from the Wall Street Journal and Wikipedia respectively. While the relations described by the RST are generic enough to work with any document or text, specific coherence relations can be defined for other domains like Biology, Political Sciences, Math, etc. The advantage offered by the proposed system is that the syntactic transformations described by Chapter 5 would work with any text, regardless of the domain or relation inventory. However, specific templates would have to be defined for a new relation type. Likewise, the discourse parser would have to be re-trained to account for this new relation. Thus, by simply re-training the parser and adding a new template, one can scale the system to a new domain.

3. **Generating distractors**: This dissertation does not address how to formulate multiple choice questions (Araki et al., 2016). To generate meaningful distractors for such questions, one should consider exploiting discourse structures and look for confusing relation triples in the tree. For instance, a correlation relation can be confused for causation: strategies such as these can be utilized to craft smart distractors. However, to generate such distractors, we clearly need a powerful discourse parser that itself does not get confused between relations it is trying to categorize and more importantly, the presence of discourse structures that would allow us to obtain them in the first place.

4. **Evaluation of answers**: We also have not addressed how to evaluate answers written by students for a particular question. One strategy is to identify spans of tokens in text that correspond to the answer: however, it is very likely that a student may have written the answer in their own words. To account for this, a smart rating system

must be designed that compares the expected answer with the given one (Shen et al., 2019), and perhaps rates it on a scale to indicate how correct the answer is.

5. **Multi-modal question generation**: One can also consider other sources of information like images, audio files, videos, etc. for generating questions. Once again, the objective should be to generate meaningful questions. Several QA datasets have been released by researchers (Zeng et al., 2017; Lei et al., 2018) that ask simple wh-type questions around images and videos. These datasets can also be complemented with meaningful questions like the ones generated by the pipeline. This would help advance research in both multi-modal question generation and answering.

APPENDIX A

PRINCIPLES OF DISCOURSE SEGMENTATION

This appendix summarizes the principles described in Carlson and Marcu (2001) for annotating discourse segments in a document. The list given below gives concrete examples for each principle with explanations (segment boundaries are indicated via bracketing):

1. **Main Clause**: The basic elementary discourse unit or EDU is a clause. For instance:

   (1) [The company will shut down its plant.]

2. **Subordinate clauses with discourse cues**: Some clear-cut examples of sentences containing two clausal EDUs are enumerated below – each of these has a superordinate clause, and a subordinate clause, containing a discourse marker are listed below:

   *(2) [Such trappings suggest a glorious past] [**but** give no hint of a troubled present.]*

   *(3) [**Although** Mr. Freeman is retiring,] [he will continue to work as a consultant for American Express on a project basis.]*

3. **Clausal subjects and objects**: Clausal subjects and objects of verbs should not be treated as elementary discourse units:

   (4) [Deciding what constitutes terrorism can be a legalistic exercise.]

   (5) [Atco Ltd. said its utilities arm is considering building new electric power plants.]

4. **Clausal Complements**: Clausal complements of verbs are normally not fragmented into separate EDUs. However, an exception is made in the case of attribution verbs as shown later. Consider these examples:

   (6) [Ideally, wed like to be the operator of the project and a modest equity investor.]

   *(7) [The company **says**] [it will shut down its plant.]*

An exception is made if the complement is a to-infinitival:

(8) [The company wants to shut down its plant.]

5. **Coordinated sentences**: Coordinated sentences and clauses are broken into separate EDUs

   (9) [Inventories are creeping up;] [car inventories are already high,] [and big auto makers are idling plants.]

6. **Coordination in Superordinate Clauses**: When coordination clauses occur as superordinate clauses, they are treated like coordinated sentences, and should be marked as EDUs.

   (10) [The company will shut down its plant,] [and dismiss several hundred employees.]

7. **Coordination in Subordinate Clauses**: If the subordinate construction is normally segmented as an EDU in the single clause case, then the coordinate clauses are segmented as EDUs:

   (11) [The company announced] [that it will shut down its plant] [and dismiss several hundred employees.]

   If the subordinate construction is not segmented as an EDU in the single clause case, then the coordinate clauses are not segmented as EDUs:

   (12) [The company plans to shut down its plant and dismiss several hundred employees.]

8. **Syntactic Focusing Devices**: When a syntactic focusing device, such as cleft, pseudo-cleft or extraposition creates two clauses out of a single clause, the resulting construction is regarded as a single EDU. An example is given below:

   (13) [It is hard for the company to dismiss several hundred employees.]

9. **Temporal Clauses**: Clausal temporal expressions are EDUs. Temporal clauses triggered by before, after, may have a number of modifiers that are included in the EDU. Example:

   (14) [Just months before dismissing several hundred employees, ...]

10. **Temporal phrases**: Temporal phrases, such as in the morning, in the past several weeks, are not EDUs. Even if the temporal phrase is event-like in nature, it is not marked as an EDU:

    (15) [Just a week after the companys dismissal of several hundred employees, further layoffs were announced.]

11. **Correlative Subordinators**: Correlative Subordinators are marked as separate EDUs. Example:

    (16) [No sooner had they announced the closing of the plant] [than massive protests erupted on the premises.]

12. **Embedded Discourse Units**: Relative clauses, nominal postmodifiers, appositives, parentheticals are treated as embedded EDUs. Embedded units are those which modify a portion of an EDU, or break up another legitimate EDU. Examples:

    (17) [The plant] [that the company will shut down] [is in Ohio.]

    (18) [The plant] [(which is in Ohio)] [will be shut down in October.]

APPENDIX B

DISCOURSE RELATION INVENTORY

This appendix provides definitions and examples of all relations found in the RST inventory. Only the most important ones were enumerated in Table 1.4.1; here all other relation types are described.

1. ANALOGY: Here, two textual spans, often quite dissimilar, are set in correspondence in some respects. An analogy contains an inference that if two or more things agree with one another in some respects, they will probably agree in other respects. In most cases, the relation is multinuclear.

   Example:

   [And just as we did not believe the tendentious claims of the Congressmen and arms-control advocates who visited Krasnoyarsk,] [we are in no way persuaded by the assent to the tainted-meat theory by a U.S. team of scientists who met with Soviet counterparts in Washington last year.]

2. ANTITHESIS: Here, the situation presented in the nucleus comes in contrast with the situation presented in the satellite. The contrast may happen in only one or few respects, while everything else can remain the same in other respects.

   Example:

   [Although the legality of these sales is still an open question,] [the disclosure couldn't be better timed to support the position of export-control hawks in the Pentagon and the intelligence community.]

3. COMMENT: In a COMMENT relation, the satellite constitutes a subjective remark on a previous segment of the text. It is not an evaluation or an interpretation. The comment

is usually presented from a perspective that is outside of the elements in focus in the nucleus.

Example:

Sears said [claims from the storm,] [as expected,] [reduced its third quarter net by $80 million, or 23 cents a share.]

4. Concession: The situation indicated in the nucleus is contrary to expectation in the light of the information presented in the satellite.

   Example:

   [Still, todays highest-yielding money funds may beat CDs over the next year] [even if rates fall]

5. Conclusion: In this relation, the satellite presents a final statement that wraps up the situation presented in the nucleus. The satellite is a reasoned judgment, inference, necessary consequence, or final decision with respect to the situation presented in the nucleus.

   Example:

   China could exhaust its foreign-exchange reserves as early as next year, a Western government report says, unless imports are cut drastically to help narrow the balance-of-payments deficit. According to the report, completed last month, if China's trade gap continues to widen at the pace seen in the first seven months of this year, the reserves would be wiped out either in 1990 or 1991. [A country is considered financially healthy if its reserves cover three months of its imports. The $14 billion of reserves China had in June would cover just that much.] [The report by the Western government, which declines to be identified, concludes that "a near-term foreign-exchange payment problem can be avoided only if import growth drops to below 5% per annum."]

6. Contingeny: Here, the satellite suggests an abstract notion of recurrence or habituality. Hence, the expression of time, place, or condition is not the primary focus.

   Example:

   [They have a life of their own and can be counted on to look good and perform] [whenever a cast isn't up to either.]

7. Definition: Here, the satellite gives a definition of the nucleus.

   Example:

   [Deciding what constitutes terrorism can be a legalistic exercise.] [The U.S. defines it as "premediated, politically motivated violence perpetrated against noncombatant targets by subnational groups or clandestine state agents."]

8. Disjunction: This is a multinuclear relation whose elements can be listed as alternatives, either positive or negative.

   Example:

   [Yet Israel will neither share power with all these Arabs] [nor, says its present prime minister, redraw its borders closer to its pre-1967 Jewish heartland.]

9. Preference: The relation compares two situations, acts, events, etc., and assigns a clear preference for one of the situations, acts, events, etc. The preferred situation, act, event, etc. is the nucleus.

   Example:

   [She has thrown extravagant soirees for crowds of people,] [but prefers more intimate gatherings.]

10. Proportion: A Proportion relation expresses a proportionality or equivalence of tendency or degree between two nuclei. It is always multinuclear.

Example:

"We can't have this kind of thing happen very often. [When the little guy gets frightened,] [the big guys hurt badly.] Merrill Lynch can't survive without the little guy."

11. QUESTION-ANSWER: In such a relation, one textual span poses a question (not necessarily realized as an interrogative sentence), and the other text span answers the question. The relation may be mononuclear or multinuclear, depending on the context. When the question is perceived as more important than the answer, the question is assigned the role of nucleus and the answer is the satellite.

    Example:

    [But are these four players, three of them in their 80s, ready to assume a different role after 88 years, collectively, of service on the high court?] [Every indication is that the four are prepared to accept this new role, and the frustrations that go with it, but in different ways. Justices Brennan and Stevens appear philosophical about it; Justices Marshall and Blackmun appear fighting mad.]

APPENDIX C

EXAMPLES SHOWING PIPELINE OUTPUT

In this appendix, some more examples of questions as generated by the pipeline described in Chapter 7 are provided. The SQuAD dataset (Rajpurkar et al., 2016) was used to craft these questions. In each case, it can be observed that the pipeline generates more general-level questions than the ones posed by humans in SQuAD.

Passage:

There are three major types of rock: igneous, sedimentary, and metamorphic. The rock cycle is an important concept in geology which illustrates the relationships between these three types of rock, and magma. When a rock crystallizes from melt (magma and/or lava), it is an igneous rock. This rock can be weathered and eroded, and then redeposited and lithified into a sedimentary rock, or be turned into a metamorphic rock due to heat and pressure that change the mineral content of the rock which gives it a characteristic fabric. The sedimentary rock can then be subsequently turned into a metamorphic rock due to heat and pressure and is then weathered, eroded, deposited, and lithified, ultimately becoming a sedimentary rock. Sedimentary rock may also be re-eroded and redeposited, and metamorphic rock may also undergo additional metamorphism. All three types of rocks may be re-melted; when this happens, a new magma is formed, from which an igneous rock may once again crystallize.

Questions in SQuAD:

No.	Question
1	An igneous rock is a rock that crystallizes from what?
2	Sedimentary rock can be turned into which three types of rock?
3	When the three types of rock are re-melted what is formed?
4	What are the three major types of rock?
5	What changes the mineral content of a rock?

Questions generated:

No.	Question	Difficulty
1	Why is the rock cycle an important concept in geology?	2
2	Under what circumstances is a rock an igneous rock?	1
3	How does an igneous rock turn into a metamorphic rock?	2
4	Under what circumstances is a new magma formed?	2
5	How may an igneous rock once again crystallize?	2

Passage:

In 1755, six colonial governors in North America met with General Edward Braddock, the newly arrived British Army commander, and planned a four-way attack on the French. None succeeded and the main effort by Braddock was a disaster; he was defeated in the Battle of the Monongahela on July 9, 1755 and died a few days later. British operations in 1755, 1756 and 1757 in the frontier areas of Pennsylvania and New York all failed, due to a combination of poor management, internal divisions, and effective Canadian scouts, French regular forces, and Indian warrior allies. In 1755, the British captured Fort Beausjour on the border separating Nova Scotia from Acadia; soon afterward they ordered the expulsion of the Acadians. Orders for the deportation were given by William Shirley, Commander-in-Chief, North America, without direction from Great Britain. The Acadians, both those captured in arms and those who had sworn the loyalty oath to His Britannic Majesty, were expelled. Native Americans were likewise driven off their land to make way for settlers from New England.

Questions in SQuAD:

No.	Question
1	When did colonial governors meet with General Edward Braddock about attack on the french?
2	How successful was initial effort by Braddock?
3	Why did British operation fail in 1755, 56, 57?
4	In 1755 what fort did British capture?
5	What order did British make of French?

Questions generated:

No.	Question	Difficulty
1	Why did six colonial governors in North America meet with General Edward Braddock, the newly arrived British Army commander?	1
2	Why was the main effort by Braddock a disaster?	2
3	Why did British operations in 1755, 1756 and 1757 in the frontier areas of Pennsylvania and New York all fail?	2
4	Under what conditions were orders for the deportation given by William Shirley, Commander-in-Chief, North America?	3
5	Why were Native Americans likewise driven off?	2

Passage:

The Doctor rarely travels alone and often brings one or more companions to share these adventures. His companions are usually humans, as he has found a fascination with planet Earth. He often finds events that pique his curiosity as he tries to prevent evil forces from harming innocent people or changing history, using only his ingenuity and minimal resources, such as his versatile sonic screwdriver. As a Time Lord, the Doctor has the ability to regenerate when his body is mortally damaged, taking on a new appearance and personality. The Doctor has gained numerous reoccurring enemies during his travels, including the Daleks, the Cybermen, and the Master, another renegade Time Lord.

Questions in SQuAD:

No.	Question
1	How often does the Doctor travel by himself?
2	What enemy of Doctor Who is also a Time Lord?
3	What does Doctor Who do when his body is mortally damaged?
4	What type of beings does Doctor Who usually take with him on his travels?
5	What type of Lord is Doctor Who?

Questions generated:

No.	Question	Difficulty
1	Why are the Doctor's companions usually human?	1
2	Under what circumstances does the Doctor find events that pique his interest?	2
3	How does the Doctor try to prevent evil forces from harming innocent people or changing history?	2
4	Under what circumstances does the Doctor take on a new appearance and personality?	3

REFERENCES

Anderson, L. W., D. R. Krathwohl, P. Airasian, K. Cruikshank, R. Mayer, P. Pintrich, J. Raths, and M. Wittrock (2001). A taxonomy for learning, teaching and assessing: A revision of blooms taxonomy. *New York. Longman Publishing. Artz, AF, & Armour-Thomas, E.(1992). Development of a cognitive-metacognitive framework for protocol analysis of mathematical problem solving in small groups. Cognition and Instruction 9*(2), 137–175.

Araki, J., D. Rajagopal, S. Sankaranarayanan, S. Holm, Y. Yamakawa, and T. Mitamura (2016). Generating questions and multiple-choice answers using semantic analysis of texts. In *COLING*, pp. 1125–1136.

Asher, N., N. M. Asher, and A. Lascarides (2003). *Logics of conversation*. Cambridge University Press.

Bach, N. X., N. L. Minh, and A. Shimazu (2012). A reranking model for discourse segmentation using subtree features. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 160–168. Association for Computational Linguistics.

Becker, L., S. Basu, and L. Vanderwende (2012). Mind the gap: learning to choose gaps for question generation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 742–751. Association for Computational Linguistics.

Bengio, Y., R. Ducharme, P. Vincent, and C. Jauvin (2003). A neural probabilistic language model. *Journal of machine learning research 3*(Feb), 1137–1155.

Bloom, B. S. (1964). *Taxonomy of educational objectives*, Volume 2. Longmans, Green New York.

Braud, C., O. Lacroix, and A. Søgaard (2017). Does syntax help discourse segmentation? not so much. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 2432–2442.

Braud, C., B. Plank, and A. Søgaard (2016). Multi-view and multi-task training of rst discourse parsers. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1903–1913.

Brooks, P. H., D. J. Arnold, and M. Iacobbo (1977). Some cognitive aspects of reading comprehension. *Peabody Journal of Education 54*(3), 146–153.

Brown, T. B., B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei (2020). Language models are few-shot learners.

Cannell, C. F., P. V. Miller, and L. Oksenberg (1981). Research on interviewing techniques. *Sociological methodology 12*, 389–437.

Cardoso, P. C., E. G. Maziero, M. L. C. Jorge, E. M. Seno, A. Di Felippo, L. H. M. Rino, M. d. G. V. Nunes, and T. A. Pardo (2011). Cstnews-a discourse-annotated corpus for single and multi-document summarization of news texts in brazilian portuguese. In *Proceedings of the 3rd RST Brazilian Meeting*, pp. 88–105.

Carlson, L. and D. Marcu (2001). Discourse tagging reference manual. *ISI Technical Report ISI-TR-545 54*, 56.

Carlson, L., D. Marcu, and M. E. Okurowski (2003). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, pp. 85–112. Springer.

Caruana, R. (1997). Multitask learning. *Machine learning 28*(1), 41–75.

Charniak, E. (2000). A maximum-entropy-inspired parser. In *1st Meeting of the North American Chapter of the Association for Computational Linguistics*.

Collobert, R. and J. Weston (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 160–167. ACM.

Collobert, R., J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa (2011). Natural language processing (almost) from scratch. *Journal of machine learning research 12*(Aug), 2493–2537.

Curto, S., A. C. Mendes, and L. Coheur (2012). Question generation based on lexico-syntactic patterns learned from the web. *Dialogue & Discourse 3*(2), 147–175.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186.

Dozat, T. and C. D. Manning (2016). Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Du, X., J. Shao, and C. Cardie (2017). Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

Dyer, C., A. Kuncoro, M. Ballesteros, and N. A. Smith (2016). Recurrent neural network grammars. In *Proceedings of NAACL-HLT*, pp. 199–209.

Feng, V. W. and G. Hirst (2014a). 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)*, pp. 511–521.

Feng, V. W. and G. Hirst (2014b). Two-pass discourse segmentation with pairing and global features. *arXiv preprint arXiv:1407.8215*.

Gao, Y., L. Bing, W. Chen, M. R. Lyu, and I. King (2018). Difficulty controllable generation of reading comprehension questions. *arXiv preprint arXiv:1807.03586*.

Givon, T. (1984). Prolegomena to discourse-pragmatics. *Journal of Pragmatics 8*(4), 489 – 516.

Glover, J. A., B. S. Plake, B. Roberts, J. W. Zimmer, and M. Palmere (1981). Distinctiveness of encoding: The effects of paraphrasing and drawing inferences on memory from prose. *Journal of Educational Psychology 73*(5), 736.

Goldberg, Y. (2016). A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research 57*, 345–420.

Graesser, A. C., D. S. McNamara, and M. M. Louwerse (2003). What do readers need to learn in order to process coherence relations in narrative and expository text. *Rethinking reading comprehension*, 82–98.

Grimes, J. E. (1972). The thread of discourse.

Grosz, B. J. and C. L. Sidner (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics 12*(3), 175–204.

Heilman, M. and N. A. Smith (2010). Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 609–617. Association for Computational Linguistics.

Hernault, H., D. Bollegala, and M. Ishizuka (2010). A sequential model for discourse segmentation. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 315–326. Springer.

Hirschberg, J. and D. Litman (1993). Empirical studies on the disambiguation of cue phrases. *Computational linguistics 19*(3), 501–530.

Holland, P. W. and D. T. Thayer (1985). An alternate definition of the ets delta scale of item difficulty. program statistics research.

Hsu, F.-Y., H.-M. Lee, T.-H. Chang, and Y.-T. Sung (2018). Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques. *Information Processing & Management 54*(6), 969–984.

Huang, L., H. Ji, K. Cho, I. Dagan, S. Riedel, and C. R. Voss (2018). Zero-shot transfer learning for event extraction. In *56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pp. 2160–2170. Association for Computational Linguistics (ACL).

Hutzler, D., E. David, M. Avigal, and R. Azoulay (2014). Learning methods for rating the difficulty of reading comprehension questions. In *Software Science, Technology and Engineering (SWSTE), 2014 IEEE International Conference on*, pp. 54–62. IEEE.

Iruskieta, M., M. J. Aranzabe, A. D. de Ilarraza, M. Lersundi, and O. L. de Lacalle. The rst basque treebank: an online search interface to check rhetorical relations.

Ji, Y. and J. Eisenstein (2014). Representation learning for text-level discourse parsing. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 13–24.

Joty, S., F. Guzmán, L. Màrquez, and P. Nakov (2017). Discourse structure in machine translation evaluation. *Computational Linguistics 43*(4), 683–722.

Kingma, D. P. and J. Ba (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kishimoto, Y., Y. Murawaki, and S. Kurohashi (2020). Adapting bert to implicit discourse relation classification with a focus on discourse connectives. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pp. 1152–1158.

Lei, J., L. Yu, M. Bansal, and T. Berg (2018). Tvqa: Localized, compositional video question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1369–1379.

Li, J., R. Li, and E. Hovy (2014, October). Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 2061–2069. Association for Computational Linguistics.

Li, Q., T. Li, and B. Chang (2016). Discourse parsing with attention-based hierarchical neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 362–371.

Li, Y., Y. Yang, J. Wang, and W. Xu (2018). Zero-shot transfer vqa dataset. *arXiv preprint arXiv:1811.00692*.

Lin, X., S. Joty, P. Jwalapuram, and M. S. Bari (2019). A unified linear-time framework for sentence-level discourse parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4190–4200.

Lindberg, D., F. Popowich, J. Nesbit, and P. Winne (2013). Generating natural language questions to support learning on-line.

Maas, A. L., R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts (2011, June). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA, pp. 142–150. Association for Computational Linguistics.

Mabona, A., L. Rimell, S. Clark, and A. Vlachos (2019). Neural generative rhetorical structure parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2284–2295.

Mann, W. C. and S. A. Thompson (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text 8*(3), 243–281.

Mannem, P., R. Prasad, and A. Joshi (2010). Question generation from paragraphs at upenn: Qgstec system description. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pp. 84–91.

Manning, C. D., M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky (2014). The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.

Mazidi, K. and R. D. Nielsen (2014). Linguistic considerations in automatic question generation. In *ACL (2)*, pp. 321–326.

McDonald, S. and M. Ramscar (2001). Testing the distributioanl hypothesis: The influence of context on judgements of semantic similarity. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, Volume 23.

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mitkov, R. and L. A. Ha (2003). Computer-aided generation of multiple-choice tests. In *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing-Volume 2*, pp. 17–22. Association for Computational Linguistics.

Moldovan, D. and E. Blanco (2012, May). Polaris: Lymba's semantic parser. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, pp. 66–72. European Language Resources Association (ELRA).

Morey, M., P. Muller, and N. Asher (2018). A dependency perspective on rst discourse parsing and evaluation. *Computational Linguistics 44*(2), 197–235.

Muller, P., C. Braud, and M. Morey (2019). Tony: Contextual embeddings for accurate multilingual discourse segmentation of full documents. *NAACL HLT 2019*, 115.

Namba, M. (2012). Intelligent tutoring system with associative cellular neural network. In *E-Learning-Organizational Infrastructure and Tools for Specific Areas*. InTech.

Nunan, D. and K. Keobke (1995). Task difficulty from the learner's perspective: Perceptions and reality. *Hong Kong papers in linguistics and language teaching 18*, 1–12.

Olney, A. M., A. C. Graesser, and N. K. Person (2012). Question generation from concept maps. *Dialogue & Discourse 3*(2), 75–99.

Pan, L., W. Lei, T.-S. Chua, and M.-Y. Kan (2019). Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949*.

Papasalouros, A., K. Kanaris, and K. Kotis (2008). Automatic generation of multiple choice questions from domain ontologies. In *e-Learning*, pp. 427–434. Citeseer.

Pennington, J., R. Socher, and C. D. Manning (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer (2018). Deep contextualized word representations. In *Proc. of NAACL*.

Pires, T., E. Schlinger, and D. Garrette (2019). How multilingual is multilingual bert? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4996–5001.

Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of pragmatics 12*(5-6), 601–638.

Prasad, R., N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. K. Joshi, and B. L. Webber (2008). The penn discourse treebank 2.0. In *LREC*. Citeseer.

Radford, A., K. Narasimhan, T. Salimans, and I. Sutskever (2018). Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Rajpurkar, P., J. Zhang, K. Lopyrev, and P. Liang (2016). Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392.

Redeker, G., I. Berzlánovich, N. Van Der Vliet, G. Bouma, and M. Egg (2012). Multi-layer discourse annotation of a dutch text corpus. *age 1*, 2.

Reed, C. and A. Daskalopulu (1998). Modelling contractual arguments. In *Proceedings of the 4th International Conference on Argumentation (ISSA-98). SICSAT*. Citeseer.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Sacks, H., E. A. Schegloff, and G. Jefferson (1978). A simplest systematics for the organization of turn taking for conversation. In *Studies in the organization of conversational interaction*, pp. 7–55. Elsevier.

Schweter, S. and S. Ahmed (2019). Deep-eos: General-purpose neural networks for sentence boundary detection. In *KONVENS*.

Sheehan, K. M., M. Flor, and D. Napolitano (2013). A two-stage approach for generating unbiased estimates of text complexity. In *Proceedings of the Workshop on Natural Language Processing for Improving Textual Accessibility*, pp. 49–58.

Shen, A., B. Salehi, T. Baldwin, and J. Qi (2019). A joint model for multimodal document quality assessment. *CoRR abs/1901.01010*.

Shi, K., Z. Liu, and N. F. Chen (2020). An end-to-end document-level neural discourse parser exploiting multi-granularity representations.

Soricut, R. and D. Marcu (2003). Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, Stroudsburg, PA, USA, pp. 149–156. Association for Computational Linguistics.

Stasaski, K. and M. A. Hearst (2017). Multiple choice question generation utilizing an ontology. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 303–312.

Stede, M. and A. Neumann (2014). Potsdam commentary corpus 2.0: Annotation for discourse research. In *LREC*, pp. 925–929.

Storey, V. C. (1993). Understanding semantic relationships. *The VLDB Journal 2*(4), 455–488.

Strubell, E., P. Verga, D. Andor, D. Weiss, and A. McCallum (2018). Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 5027–5038.

Subba, R. and B. Di Eugenio (2007). Automatic discourse segmentation using neural networks. In *Proc. of the 11th Workshop on the Semantics and Pragmatics of Dialogue*, pp. 189–190.

Taboada, M. (2009). Implicit and explicit coherence relations. *Discourse, of course. Amsterdam: John Benjamins*, 127–140.

Torgesen, J. (2004). Adolescent literacy, reading comprehension & the fcat. In *CLAS Conference, Naples, FL. Retrieved*, Volume 3, pp. 07.

Uzêda, V. R., T. A. S. Pardo, and M. d. G. V. Nunes (2008). Evaluation of automatic text summarization methods based on rhetorical structure theory. In *2008 Eighth International Conference on Intelligent Systems Design and Applications*, Volume 2, pp. 389–394. IEEE.

Van Gelderen, A., R. Schoonen, K. De Glopper, J. Hulstijn, A. Simis, P. Snellings, and M. Stevenson (2004). Linguistic knowledge, processing speed, and metacognitive knowledge in first-and second-language reading comprehension: A componential analysis. *Journal of educational psychology 96*(1), 19.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin (2017). Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008.

Verberne, S. (2007). Paragraph retrieval for why-question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 922–922.

Vinyals, O., M. Fortunato, and N. Jaitly (2015). Pointer networks. *Advances in neural information processing systems 28*, 2692–2700.

Wang, Y., S. Li, and J. Yang (2018, October-November). Toward fast and accurate neural discourse segmentation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, pp. 962–967. Association for Computational Linguistics.

Wu, S. and M. Dredze (2019). Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 833–844.

Xu, J., Z. Gan, Y. Cheng, and J. Liu (2020). Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Yahya, A. A. and A. Osama (2011). Automatic classification of questions into bloom's cognitive levels using support vector machines.

Yang, Z., D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1480–1489.

Yin, J., X. Jiang, Z. Lu, L. Shang, H. Li, and X. Li (2015). Neural generative question answering. *arXiv preprint arXiv:1512.01337*.

Young, T., D. Hazarika, S. Poria, and E. Cambria (2018). Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine 13*(3), 55–75.

Yu, N., M. Zhang, and G. Fu (2018, August). Transition-based neural RST parsing with implicit syntax features. In *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA, pp. 559–570. Association for Computational Linguistics.

Zaheer, M., G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, et al. (2020). Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*.

Zeldes, A. (2016). rstweb-a browser-based annotation interface for rhetorical structure theory and discourse relations. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 1–5.

Zeng, K.-H., T.-H. Chen, C.-Y. Chuang, Y.-H. Liao, J. C. Niebles, and M. Sun (2017). Leveraging video descriptions to learn video question answering. *ArXiv abs/1611.04021*.

BIOGRAPHICAL SKETCH

Takshak Desai, son of Purvi Desai, was born in Mumbai, India. He completed his bachelor's degree in Computer Engineering from Mumbai University in May 2016 and joined UT Dallas in August 2016 to pursue his master's and PhD degrees. Takshak worked with Dr. Dan Moldovan to research in the fields of discourse parsing and question generation. He interned as an NLP Software Engineer at Lymba Corporation in Richardson, TX in Summer 2019 and remotely as a Data Science Intern at iManage LLC in Summer 2020. He will join iManage at their headquarters in Chicago, IL as a Data Scientist in Summer 2021.

CURRICULUM VITAE

Takshak Desai

April 2021

Contact Information:

Department of Computer Science
The University of Texas at Dallas
800 W. Campbell Rd.
Richardson, TX 75080-3021, U.S.A.

Email: takshak.desai@utdallas.edu

Educational History:

BS, Computer Engineering, Mumbai University, 2012
MS, Computer Science, The University of Texas at Dallas, 2020
PhD, Computer Science, The University of Texas at Dallas, 2021

Discourse Parsing and its Application to Question Generation
PhD Dissertation
Department of Computer Science, The University of Texas at Dallas
Advisor: Dr. Dan I. Moldovan

Employment History:

Data Science Intern, iManage LLC, Chicago, IL, May 2020 – August 2020
Software Engineer Intern, Lymba Corporation, Richardson, TX, May 2019 – August 2019

Publications[1]:

1. Desai, Takshak, Udit Deshmukh, Mihir Gandhi, and Lakshmi Kurup. "A hybrid approach for detection of plagiarism using natural language processing." In Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, pp. 1-6. 2016.

2. * Desai, Takshak, Parag Dakle, and Dan Moldovan. "Generating questions for reading comprehension using coherence relations." In Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications, pp. 1-10. 2018.

3. * Desai, Takshak, and Dan I. Moldovan. "Towards Predicting Difficulty of Reading Comprehension Questions." In The Thirty-Second International Flairs Conference. 2019.

[1]Publications marked with an asterisk are relevant to this dissertation

4. * Desai, Takshak, Parag Pravin Dakle, and Dan Moldovan. "Joint Learning of Syntactic Features helps Discourse Segmentation." In Proceedings of The 12th Language Resources and Evaluation Conference, pp. 1073-1080. 2020.

5. Dakle, Parag Pravin, Takshak Desai, and Dan Moldovan. "A study on entity resolution for email conversations." In Proceedings of The 12th Language Resources and Evaluation Conference, pp. 65-73. 2020.

6. * Desai, Takshak, and Dan Moldovan. "Structure, Syntax and Context-aware RST Discourse Parsing" In Proceedings of the 2021 IEEE 15th International Conference on Semantic Computing (IEEE ICSC 2021), pp. 155-162. IEEE, 2021.