CLASSIFYING ARGUMENTS BY SCHEME

Vanessa Wei Feng

Department of Computer Science

University of Toronto

Paper submitted to The University of Toronto

for the degree of Master of Science

November 24, 2010

Abstract

Argumentation schemes are structures or templates for various kinds of arguments. The argumentation scheme classification system that we present in this paper introduces a new task in this field. To the best of our knowledge, this is the first attempt to classify arguments into argumentation schemes automatically.

Given the text of an argument with premises and conclusion identified, we classify it as an instance of one of five common schemes, using general features and other features specific to each scheme, including lexical, syntactic, and shallow semantic features. We achieve accuracies of 63–91% in one-against-others classification and 80–94% in pairwise classification (baseline = 50% in both cases).

We design a pipeline framework whose ultimate goal is to reconstruct the implicit premises in an argument, and our argumentation scheme classification system is aimed to address the third component in this framework. While the first two portions of this framework can be fulfilled by work of other researchers, we propose a syntactic-based approach to the last component of this framework. The completion of the entire system will benefit many professionals in applications such as automatic reasoning assistance.

Acknowledgements

I would like to thank my parents Huaying Sun and Hanjie Feng for their supports throughout my life and especially during my stay in Canada. I have never been away from my hometown Shanghai, China for this long. Things were not only exciting but also difficult for me while learning to live on my own, and I know it must not be easy for my parents either.

I am sincerely grateful to my supervisor Professor Graeme Hirst at the Department of Computer Science, University of Toronto. He is such a gentleman with great humor sense, which makes it always a pleasure to talk to him. He was always so encouraging that makes me confident in pursuing my research objectives. He gave me so much invaluable academic advice, which I truly appreciated, and also left me sufficient freedom for my independent research. As a non-native speaker, who came to an English-speaking country for study for the first time, I had a relatively tough time in writing scientific papers and expressing my ideas sometimes. Professor Hirst took great patience in pointing out my linguistic mistakes in both writing and speaking, and always tried his best to explain those subtle nuances between different wording. Especially during the completion of this paper, he must have put much effort in proofreading, and thinking about better document organization for countless versions of this paper. I am also thankful to my second reader Professor Suzanne Stevenson, who took time and

effort reading my paper during her precious sabbatical and gave invaluable feedback which would always be helpful for my future research life. I must acknowledge the generous financial support of the funding from the Natural Sciences and Engineering Research Council of Canada and from the University of Toronto.

Finally, my gratitude to all my friends in Toronto. As a graduate student in a foreign country, I have to say the bonds between friends are more or less looser compared to the situation in my own country. But the occasional gatherings with them still constitute one of the best parts of my life in Toronto. I am also indebted to all my friends in Shanghai. Though they could only get in touch with my by chatting online during their late nights, they never failed to remind me of those lovely times we spent together.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The study of argumentation has a long history in philosophy, rhetoric, and logic, where a number of argumentation theories have been proposed and developed. Recent research has started to focus on the interdisciplinary area lying between artificial intelligence and theories of argumentation, in the hope that real-life applications, such as reasoning assistance, legal case mining, and judgment prediction can benefit professions such as lawyers, philosophers, and politicians, with enhanced reasoning and arguing skills, more efficient argument searching and more effective argument understanding.

While most research interests focus on argument detection, conclusion/premise classification and argument structure visualization, we investigate a new task in this field: the classification of arguments by the argumentation schemes that they use. An argumentation scheme, informally, is a framework or structure for a (possibly defeasible) argument; we will give a more-formal definition and examples in Section 1.1.3. Our work is motivated by the need to determine the *enthymemes* (unstated or implicitly stated premises) that arguments written in natural language normally draw on, since enthymemes often complicate matters, as they usually serve as unstated assump-

tions that must be true for the premises to lead to the conclusion. We believe that first identifying the particular argumentation scheme that an argument is using will help to bridge the gap between stated and unstated propositions in the argument, because each argumentation scheme is a relatively fixed "template" for arguing.

Our ultimate goal is to construct the enthymemes in an argument by the following method: First classify its argumentation scheme, then fit the stated propositions into the corresponding template, and from this infer the enthymemes (see Section 5.2 for a detailed example). Since determining the unstated assumptions is an integral part of understanding, supporting, or attacking the entire argument, constructing enthymemes is an important problem in argument understanding.

In this chapter, we will first briefly introduce the background of the argumentation theory that we use as the theoretical framework for our work; then we will describe the dataset we use in our work; after that, we will refer to the related work in argument visualization and argument mining; last, we will present the overall framework of our argumentation scheme classification system.

1.1 Background

1.1.1 Definitions of arguments

Arguments are commonly seen in everyday life: when people try to persuade others to accept some particular idea, or to convince others that some particular statement is untrue, they tend to present their reasoning in the form of arguments. An example of argument is shown in Example 1[1].

[1]This example is reproduced from `http://www.arg.dundee.ac.uk/projects/araucariadb/viewargument.php?id=48`.

Example 1. ***An example of argument***

Pumping ground water indiscriminately without recharging is creating havoc. Because when the water table comes down, it affects the drinking water and tubewells get dislodged, and such reduction of level of water is provoked by indiscriminate pumping.

Typically, there is a final statement in an argument, which is the proposition that the speaker advocates — such a statement is usually known as conclusion. In Example 1, "Pumping ground water indiscriminately without recharging is creating havoc" is the conclusion. In order to enhance the audience's confidence in the conclusion, the speaker needs to present some basic facts which are readily accepted by the audience, and additional logical relations which are able to lead from the facts to the final conclusion — these basic facts along with the underlying logic are usually known as premises, and in particular the logical relations are also known as warrants. In Example 1, the fact is conveyed by the proposition "such reduction of level of water is provoked by indiscriminate pumping" and the warrant is stated as "when the water table comes down, it affects the drinking water and tubewells get dislodged".

More formally, an argument is constituted by a set of propositions, which include the following three components:

Definition 1. ***Components of a typical argument***

1. *Argument premise(s)*
2. *Argument warrant*
3. *Conclusion*

The argument warrant is the underlying logic which bridges the facts conveyed in argument premise(s) and the final statement conveyed in conclusion. In other words, the

warrant is the key ingredient which makes the whole argument reliable and persuasive, and ultimately, makes the proposition proposed to the speaker acceptable by the audience. Such function of warrant can be realized by various kinds of logical relations, such as the consequential relation (detailed discussion about logical relations is in 1.1.3) in Example 1.

The order of *argument premise*, *argument warrant*, and *conclusion* is not necessarily as they are in Definition 1 — it depends on the particular intention of the speaker: if the speaker intends to emphasize his or her ultimate statement, he or she might present the conclusion at the very beginning of the argument followed by several supporting reasons; if the speaker wants to make the audience accept the final statement more naturally, he or she might present several well-accepted facts first, and then introduce the conclusion by valid reasoning upon those facts.

1.1.2 Enthymemes

Though a typical argument should have the three components described in Definition 1, sometimes arguments written in natural languages may lack the argument warrant. For example, it would be very awkward to see the argument in Example 2 in real life; rather, normally people would just state the argument by saying "We are not having a picnic today, because it is raining outside", leaving the warrant "if it rains we would not have a picnic" implicit. The reason is that the proposition expressed in the warrant is so widely acknowledged that even without stating it explicitly, the underlying logic of the argument is still perceivable enough.

Example 2. ***An example of awkward argument***

We are not having a picnic today, because it is raining outside and if it rains we would not have

a picnic.

This phenomenon of implicit warrant is quite common with respect to arguments written in natural languages, but other premises may be left unstated or implicitly stated as well. Unstated premises and implicitly stated premises are not directly mentioned in the text, but are usually so widely accepted that are expected to be born in the reader's mind naturally, or are supposed to be easily inferrable from texts stated elsewhere. Such unstated or implicitly stated premises are called *enthymemes*, and arguments with enthymemes are said to be *enthymematic*.

1.1.3 Argumentation schemes and scheme-sets

Argumentation schemes are different structures or templates for forms of arguments, based on the nature of the underlying logical relations. The arguments need not be indefeasible; on the contrary, most argumentation schemes are for *presumptive* or *defeasible arguments* (Walton and Reed, 2002). For example, *argument from sign* is a commonly used scheme in everyday arguments; a definition of the scheme is given in Example 3 and Example 4 shows an example of this scheme.

Example 3. ***Definition of argument from sign***

Premise*: A is true in this situation.*

Premise*: Event B is generally indicated as true when its sign, A, is true in this kind of situation.*

Conclusion*: B is true in this situation.*

Critical Questions

CQ_1*: What is the strength of the correlation between A and B?*

CQ_2: Are there any events other than B that would more reliably account for A?

Example 4. ***Example of argument from sign***

Premise*: She felt uncomfortable after eating ice cream.*

Premise*: Feeling uncomfortable after having dairy products generally indicates that the person is lactose-intolerant.*

Conclusion*: She is lactose-intolerant.*

Critical Questions

CQ_1: What is the strength of the correlation between ***feeling uncomfortable after eating ice cream*** *and* ***being lactose-intolerant****? Is it confirmed that being lactose-intolerant would necessarily make people feel uncomfortable after having dairy products?*

CQ_2: Are there any events other than ***being lactose-intolerant*** *that would more reliably account for* ***her feeling uncomfortable after eating ice cream****? For example, is it possible that she is allergic to the nuts in the ice cream other than the ice cream itself?*

It has been shown that argumentation schemes are useful in evaluating common arguments as fallacious or not (van Eemeren and Grootendorst, 1992): an argument is regarded as valid if it matches all the requirements imposed by the scheme. In order to judge the weakness of an argument, a set of critical questions (i.e., CQ_1 and CQ_2 in the example above) is asked according to the particular scheme that the argument is using, and failure to satisfactorily answer any of these critical questions suggests that there may be exceptions with regard to this argument; thus, this argument cannot be deemed as indefeasible.

A list of such argumentation schemes is called a scheme-set[2], and the number of schemes in any particular scheme-set remains indefinite and unstable, since new schemes keep being found while some old schemes are found similar enough to be merged. Walton's set of 65 argumentation schemes (Walton *et al.*, 2008) is one of the best-developed scheme-sets. The five schemes listed in Table 1.1 are the most commonly used schemes defined in Walton's scheme-set, in descending order in terms of their frequencies, and they are the focus of the scheme classification system that we will describe in this paper.

Argumentation schemes are effective in analyzing written arguments which have a relatively formal structure, especially those in the legal domain, since legal texts usually have a relatively strict formal style in which the reasoning process in an argument of how the premise logically leads to the conclusion is clearly recorded. Mochales and Ieven (2009) point out that, "...classification in types of argumentation is useful for identifying problematic types of argumentation, which are often associated with fallacies."

1.1.4 Types of argumentation structures

In argumentation theories, the structure of arguments is an important concept, which is the way the protagonist or the antagonist puts forward his or her statements in the process of arguing. In the taxonomy of argumentation scheme theory, there are two different argumentation structures: *linked argumentation* and *convergent argumentation*.

A *linked argument* (LA) has two or more inter-dependent premise propositions, all of

[2]Though Walton *et al.* (2008) proposed several classification systems for argumentation schemes, upon which a taxonomy hierarchy could be built, there is no complete system to include all schemes yet. So here we still define a scheme-set as a list of argumentation schemes rather than any ontology-like concept.

Argument from example	
Premise	In this particular case, the individual a has property F and also property G.
Conclusion	Therefore, generally, if x has property F, then it also has property G.
Argument from cause to effect	
Major premise	Generally, if A occurs, then B will (might) occur.
Minor premise	In this case, A occurs (might occur).
Conclusion	Therefore, in this case, B will (might) occur.
Practical reasoning	
Major premise	I have a goal G.
Minor premise	Carrying out action A is a means to realize G.
Conclusion	Therefore, I ought (practically speaking) to carry out this action A.
Argument from consequences	
Premise	If A is (is not) brought about, good (bad) consequences will (will not) plausibly occur.
Conclusion	Therefore, A should (should not) be brought about.
Argument from verbal classification	
Individual premise	a has a particular property F.
Classification premise	For all x, if x has property F, then x can be classified as having property G.
Conclusion	Therefore, a has property G.

Table 1.1: The five most frequent schemes and their definitions in Walton's scheme-set

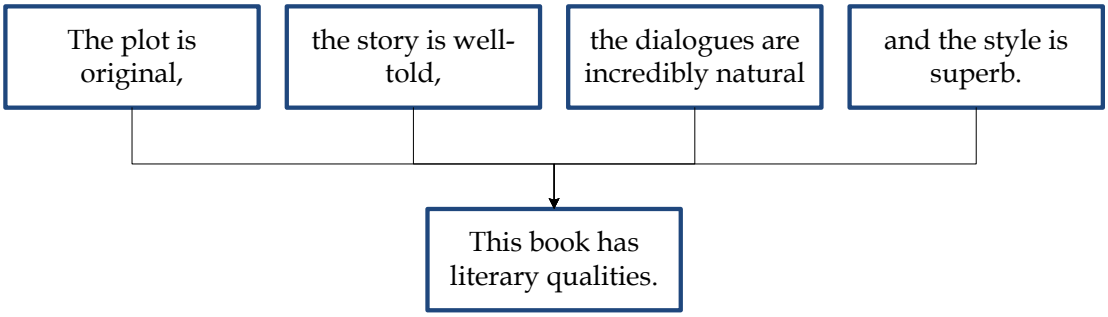


Figure 1.1: Linked argumentation structure

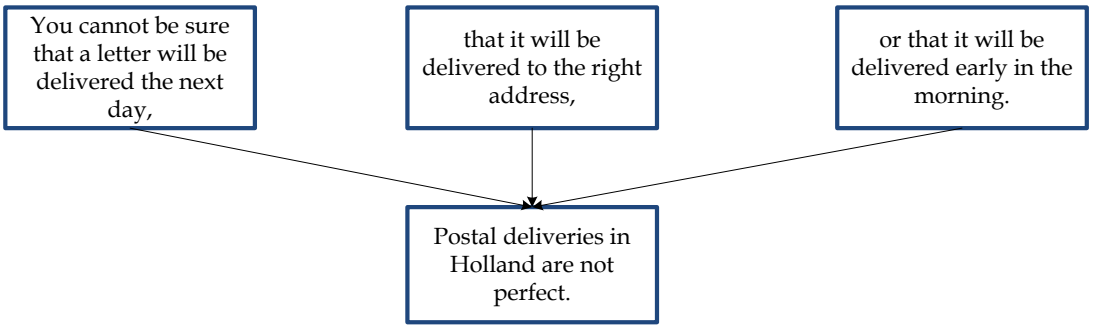


Figure 1.2: Convergent argumentation structure

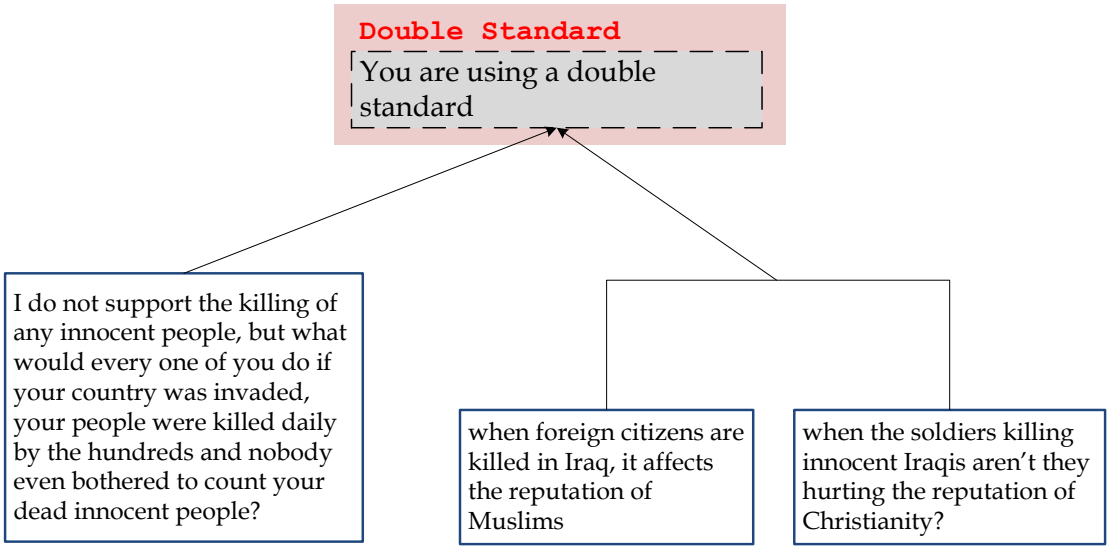
which are necessary to make the conclusion valid. An example of linked argumentation is shown in Figure 1.1[3], in which all of the four qualities: 1) the plot is original, 2) the story is well-told, 3) the dialogues are incredibly natural, and 4) the style is superb are necessary to make the conclusion "This book has literary qualities" convincing.

Conversely, a *convergent argument* (CA) requests only one premise proposition to make the conclusion valid. However, different CAs can share a common conclusion proposition. An example of convergent argumentation is shown in Figure 1.2[4], in which the three premise propositions are independent of each other, and any of them is sufficient to make the conclusion "Postal deliveries in Holland are not perfect" convincing.

Sometimes an argument is not purely linked or convergent, but contains both of

[3]This figure is reproduced from Mochales and Ieven (2009).

[4]This figure is reproduced from Mochales and Ieven (2009).



Argument Text

I am really surprised by some people's comments here. Why are we not looking for the real reasons behind what's going on? Was there any beheading and hostage taking before the US-UK invasion for Iraq? [I do not support the killing of any innocent people, but what would every one of you do if your country was invaded, your people were killed daily by the hundreds and nobody even bothered to count your dead innocent people? premise1] And why is it that [when foreign citizens are killed in Iraq, it affects the reputation of Muslims premise2], [when the soldiers killing innocent Iraqis aren't hurting the reputation of Christianity? premise3][Why such double standards? conclusion]

Redar Hawleri, Erbil (Kurdistan-Iraq) http://news.bbc.co.uk/1/hi/world/middle_east/3691548.stm

Figure 1.3: An example of complex argumentation structure

these two components, for example, a complex argument presented in Figure 1.3[5]. The left branch of this tree consists of a convergent argument, in which there is exactly one premise associated with one conclusion; the right branch with two child propositions consists of a linked argument, in which the combination of these two separate discourses serves as the premise of the conclusion as a whole. In this example, the convergent argument formed by the left branch and the linked argument formed by the right branch share the same conclusion, i.e., the root proposition of this tree.

[5]This figure is produced by me using the argument visualization tool Araucaria 3.1 (see Section 1.3.4).

1.2 AraucariaDB

One of the primary difficulties for automatic argument analysis is that suitable annotated corpora are still very rare, in spite of the endeavor of many researchers. Presently, there are only two annotated training corpora available for researchers working in this field: AraucariaDB[6] and European Court of Human Rights (ECHR) human rights documentation (HUDOC)[7]. In our work, we choose AraucariaDB as our training corpus, because AraucariaDB contains general arguments gathered from various sources while ECHR contains only legal arguments recorded in a specific court. However, Palau and Moens (2009) used both of these copora in their work (see 1.3.3 for details).

AraucariaDB is an online repository of arguments collected by scholars from around the world, which includes approximately 660 manually annotated arguments of several different types, such as *Newspaper* and *Parliamentary records*, collected from various real-life sources, such as *The Age*, *Indian parliamentary debates*, etc. Table 1.2 describes the distribution of different attributed argument sources in the entire Araucaria dataset; however, the majority of the arguments are unclassified as to source (i.e., 83.08%).

1.2.1 Argumentation scheme-sets used in annotation

There are three different scheme-sets used in the annotations in Araucaria: Walton's scheme-set (Walton *et al.*, 2008), Katzav-Reed's scheme-set (Katzav and Reed, 2004a), and Pollock's scheme-set (Pollock, 1995), most arguments in Araucaria are marked up according to only one of them.

[6] http://araucaria.computing.dundee.ac.uk/doku.php#araucaria_argumentation_corpus

[7] http://www.echr.coe.int/ECHR/EN/Header/Case-Law/HUDOC/HUDOC+database

Argument source	Occurrences in dataset	% Occurrences in dataset
Cause information	6	0.9
Weekly magazine	6	0.9
Legal	15	2.3
Discussion forum	18	2.7
Parliamentary records	18	2.7
Newspaper	49	7.4
Unclassified	550	83.1
Total	662	100.0

Table 1.2: Argument source distribution in Araucaria

AraucariaDB provides an associated XML-styled scheme-set file (*.scm) defining the restrictions on the set of argumentation schemes for each scheme-set (the number of defined schemes, whether critical questions are included, etc.), which also serves as the guideline for human annotators. Although the inventors of those argumentation schemes keep refining their scheme definitions from time to time, AraucariaDB adopts only a subset of those rather complex schemes to meet the requirement of annotating real-life arguments: 30, 25, and 7 schemes are selected from Walton's, Katzav-Reed's and Pollock's scheme-set respectively to be included in AraucariaDB. In addition, Walton's and Katzav-Reed's scheme-sets also include critical questions in their .scm files; thus annotators are able to mark up critical questions beside the basic premises and conclusions.

Our experimental dataset is composed of only those arguments annotated in accordance with Walton's scheme-set, within which the five schemes in shown in Table 1.1

constitute 61% of the total occurrences. We choose only those arguments annotated in accordance with Walton's scheme-set because there are obvious disadvantage of the arguments annotated in the other two scheme-set. For Pollock's scheme-set, there are too few annotated arguments available there. For Katzav-Reed's scheme-set, there is only one occurrence of convergent argument while the majority of the arguments are annotated as linked arguments. Since such extremely skewed distribution of linked arguments and convergent arguments is generally impossible, it suggests that annotation provided by those abiding Katzav-Reed's scheme-set might not be accurate enough.

1.2.2 Example data

An example of data from Araucaria is shown in Figure 1.4. The original text of this argument is shown in TEXT. What follows TEXT is AU, an argument unit, composed of a conclusion proposition followed by optional premise proposition(s) in a linked or convergent structure, in which each conclusion or premise proposition can be further defined as a hierarchical collection of smaller AUs. INSCHEME is the particular scheme (e.g., "Argument from Consequences") of which the current proposition is a member; enthymemes that have been made explicit by the analyst are annotated as "missing = yes" and "offset = -1".

1.2.3 Argumentation structure and argumentation scheme

One interesting observation of the Araucaria dataset is that as far as the five most frequent schemes in Table 1.1 are concerned, there exists a strong correlation between the distribution of linked argumentation structure vs. convergent argumentation structure and the particular argumentation scheme that argument is using. This observation

```
<TEXT>If we stop the free creation of art, we will stop the free viewing of art.</TEXT>
<AU>
  <PROP identifier="C" missing="yes">
   <PROPTEXT offset="-1">
     The prohibition of the free creation of art should not be brought about.
   </PROPTEXT>
   <INSCHEME scheme="Argument from Consequences" schid="0" />
  </PROP>
  <LA>
   <AU>
    <PROP identifier="A" missing="no">
     <PROPTEXT offset="0">
       If we stop the free creation of art, we will stop the free viewing of art.
     </PROPTEXT>
     <INSCHEME scheme="Argument from Consequences" schid="0" />
    </PROP>
   </AU>
   <AU>
    <PROP identifier="B" missing="yes">
     <PROPTEXT offset="-1">
       The prohibition of free viewing of art is not acceptable.
     </PROPTEXT>
     <INSCHEME scheme="Argument from Consequences" schid="0" />
    </PROP>
   </AU>
  </LA>
</AU>
```

Figure 1.4: Example of argument markup from Araucaria, showing argument units and argumentation scheme

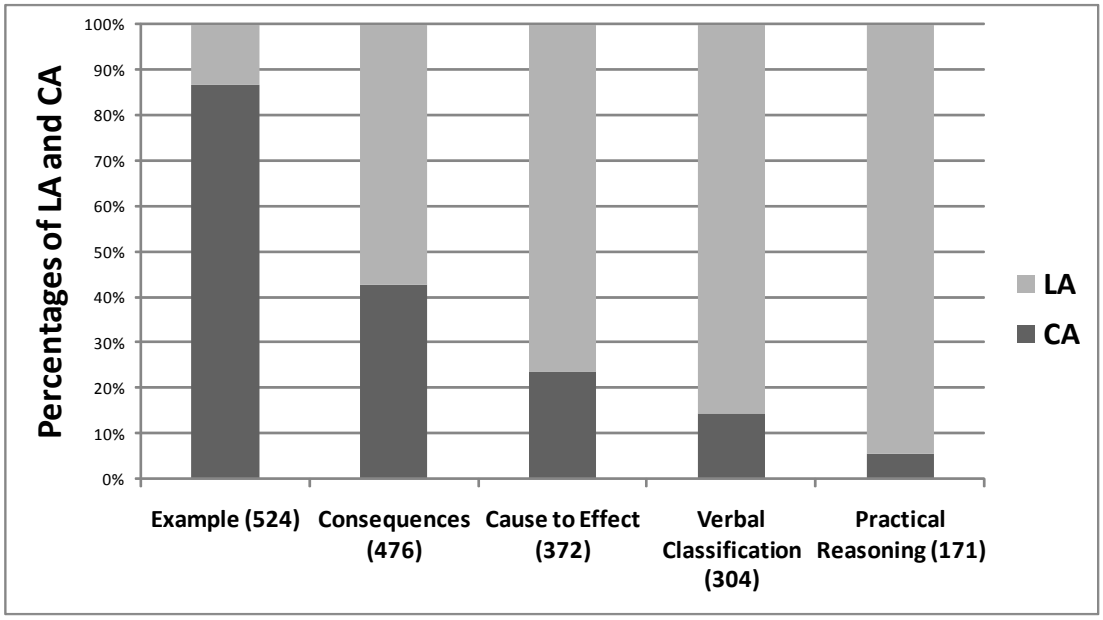


Figure 1.5: Distribution of linked arguments vs. convergent arguments in the five most frequent schemes in Walton's scheme-set. The bracketed number after each scheme name denotes how many arguments belonging to that scheme are in the dataset.

is depicted in Figure 1.5, in which we can see that for each of the five most frequent schemes, the distribution of LAs (linked arguments) and CAs (convergent arguments) is generally very skewed, especially for *argument from example* (nearly 90% of the arguments associated with that scheme are convergent arguments) and *practical reasoning* (over 90% of the arguments associated with that scheme are linked arguments), although for *argument from consequences* they are split almost evenly.

Strictly speaking, according to the definition of each argumentation scheme in Table 1.1, the number of premise propositions is already predefined. For example, *argument from example* should have exactly one premise, while *argument from cause to effect* should have exactly two premises. So theoretically there should be only one possible argumentation structure for a particular scheme — either linked or convergent, but not both — since the classification of argumentation structure is simply based on the number of

required premises[8] in an argument (see Section 1.1.4). If in its definition, there are two premise propositions, then arguments belonging to this scheme should be linked; if there is only one premise proposition in its definition, then arguments belonging to this scheme should be convergent. In this way, the observation above is not surprising, but actually to some extent reflects the fact that naturally occurring arguments may not strictly follow the theoretical scheme template, and some annotation error may be involved as well. Nevertheless, the observation above suggests that for an argument of interest, the argumentation structure might be a good indicator of the particular scheme it is using.

1.2.4 Limitations of AraucariaDB

As an annotated argumentative corpus, AraucariaDB has several limitations:

1. **A rather small size**: There are only approximately 660 annotated arguments in the corpus, which would bring about difficulties to supervised training procedures due to the lack of sufficient training data.

2. **No strict uniform guidance in annotation**: AraucariaDB is an online repository of arguments collected by scholars from around the world. Though the associated scheme-set files (*.scm) provide the basic framework in which annotators should work, there is no strict quality control over submitted arguments. Therefore, sometimes we can find very conspicuous errors in the dataset, for instance

[8]We consider "required premises" to be those which are necessary and sufficient for the argument of interest to be logically valid. So even if there is more than one premise proposition in the argument, but those premises are not dependent on each other, we split the argument into several convergent sub-arguments, rather than treat it entirely as a linked argument.

the argument shown in Example 5 is falsely annotated as using *argument from example* instead of *argument from verbal classification*.

Example 5. ***An example of argument from verbal classification but annotated as argument from example***

The argument text*: Small arms in the hands of honest citizens is a good thing. Because most people are good, having them armed is a benefit to society, because it provides a control on the tiny minority of bad people, who are always going to be armed regardless of laws.*

Premise*: Having them armed is a benefit to society. (**Individual premise**: a has a particular property F.)*

Premise (implicit)*: What is a benefit for the society is a good thing. (**Classification premise**: For all x, if x has property F , then x can be classified as having property G.)*

Conclusion*: Small arms in the hands of honest citizens is a good thing (**Conclusion**: Therefore, a has property G.).*

3. **Low agreement among different annotators**: Most arguments in the corpus are marked up by only one annotator; and for those very few arguments which have more than one annotator, no conflict resolution exists between these annotators if they had different opinions upon the particular argumentation scheme the argument is using, or what the missing premise should be.

Although AraucariaDB has the above limitations, it is nonetheless one of the best argumentative corpora available to date.

1.3 Related work

Research in argumentation is still very limited in the field of AI, and most of the work to date is still far from mature. We believe the lack of significant progress is due to the fact that argument analysis can be highly complicated, not only because it involves knowledge, skills, and experience from a number of other domains, such as law, philosophy, and education, but also because even in the realm of computational linguistics, a fully developed argument analysis theory has to interact with theories belonging to many related research areas as well, for instance, knowledge representation, information retrieval, and discourse analysis.

Another complication is that, like many application in computational linguistics, argument analysis is highly domain-specific, so generally we cannot expect to see equally satisfactory results when the same approaches by which one specific domain, such as law, have achieved success are applied to another domain.

Though research in argumentation can benefit various professions, the legal domain has been the research focus of researchers and scholars for a long time. In this section, we will first present three representative research directions targeting legal texts: Judith Dick's conceptual representation and retrieval of legal cases (Dick (1987), Dick (1991a), Dick (1991b)) (see Section 1.3.1), Brüninghaus and Ashley's legal reasoning and prediction (Katzav and Reed, 2004b) (see Section 1.3.2), and Mochales and Moens's argument mining (Mochales and Ieven (2009), Mochales and Moens (2008), Mochales Palau and Moens (2009), Moens *et al.* (2007), Palau and Moens (2009)) (see Section 1.3.3). Each of these aims to tackle a different perspective in this field while sharing motivations. Then we will present research in argumentation visualization, whose target is more general arguments other than legal texts, with detailed descrip-

tion of one representative implementation: Araucaria (Reed and Rowe, 2004) (see Section 1.3.4).

1.3.1 Dick's argument representation and conceptual retrieval of legal cases

Judith Dick (Dick (1987), Dick (1991a), Dick (1991b)) started to work on argument representation and conceptual retrieval in legal case texts long ago, though at that time no argumentative corpora such as AraucariaDB were available. She was devoted to building an automatic legal case retrieval system, which could not be satisfactorily addressed by traditional information retrieval techniques and must employ more sophisticated knowledge representation and retrieval beyond simple keyword matching.

Legal cases are recorded decisions of past cases. A case text includes the description of the facts, the statements proposed by both parties, and the reasons why the judge reached the final decision. In case law system, past cases are one of the major resources for a lawyer when preparing for a new case. Usually, a lawyer first constructs an initial claim in mind, which is based on the current case and the party of which he or she is representative, and then searches for related legal concepts and past cases which would favor his or her client. The lawyer mush also anticipate what his or her opponent would cite to attack his or her points.

Rather than focusing on legal reasoning or prediction (see 1.3.2), Dick's work aims at providing an intelligent retrieval system for legal searchers. Her ultimate retrieval system would be able to fulfil many aspects of a searcher's potential request, but fundamentally, she is motivated by the particular need of a legal profession, who generally has some unnamed but outlined ideas in mind (for example, a claim that he or she tries to make viable, or an abstract phenomenon that is representative of his or her current

case), and wishes to find out related legal concepts or cases.

The following three are the major reasons why Dick thinks traditional information retrieval systems are not sufficient for tasks in legal domain[9]:

- First, traditional evaluation methods only care about precision and recall. Besides that they are often very subjective when it comes to the judgement of document relevance, precision and recall also fail to characterize how effective a system is: the quality of the information returned.

- Secondly, traditional information retrieval systems generally employ a term-matching mechanism, which is extremely unsuitable for tasks in legal domain, even with extensions such as wild cards, fuzzy matching, or synonym substitution. It can be explained in Example 6 [10]. In other words, what lawyers want is a conceptual retrieval system, rather than a document retrieval system: documents containing the requested keywords generally are not relevant, and in most cases, documents without mentioning the keywords include information valuable to lawyers.

Example 6. ***An example of how keyword matching is unsuitable for legal text searching***

There is a story of a Vermont justice of the peace before whom a suit was brought by one farmer against another for breaking a churn. The justice took time to consider, and then said that he had looked through the statutes and could find nothing about churns, and gave judgment for the defendant. The same state of mind is shown in all our common digests and textbooks. Applications of rudimentary rules of contract or tort are tucked away under the head of Railroads or Telegraphs or go to swell treatises on historical subdivisions, such as Shipping or Equity, or are

[9] In (Dick, 1991a), she discussed other limitations of keyword representation and retrieval systems

[10] This example is copied from Dick (1991a).

gathered under the arbitrary title which is thought likely to appeal to the practical mind, such as Mercantile law. (Holmes, 1997)

Dick argued that since a conceptual retrieval system is needed, an advanced representation of legal texts, which is able to transcribe the meaning from textual representation to a conceptual representation and make inferences between those constructed concepts, is a must. She proposed to construct such a knowledge base by adopting Sowa's conceptual structures (Sowa, 1984) with extension of Somers's case grid (Somers, 1987) for text analysis and transcription. In the framework of Sowa's conceptual structures, concepts are recorded as frames with various fields denoting the restriction and information contained in that concept, and texts are represented as conceptual graphs, which are the logical forms that state the relationship between concepts and represent meaning as a whole. Somers's case grid is an extension to Sowa's conceptual structures, which is aimed to conquer some limitations that Sowa's theory posed. Somers defined concepts in a verb-centred case grammar fashion, and used a two-dimensional grid to represent meaning. By following such approaches, Dick was able to transform arguments from the natural language expressed in case texts, to a more computable format — conceptual graphs (in their linear form).

To access the concepts in a knowledge base (kb) constructed as above, Dick suggested using the Lexical Option Generator (LOG) (Miezitis, 1988) algorithm for frame matching. The retrieval system is designed to answer different kinds of questions, from matching a factual concept in the kb, to matching a relevant legal concept in the kb by another precisely named or outlined legal concept.

Dick's work was a pioneering attempt at her time, when argument representation and retrieval had not received much attention in computational linguistics. Her pro-

posed retrieval system promised lawyers a much more effective method of searching, which would override the traditional term-matching searching mechanism and meet the actual needs of lawyers. From this perspective, there is still significant value in her work even in today's world, since though twenty years have passed, research on this topic has not witnessed much progress yet. However, from a more realistic view, though the Lexical option generator (LOG) searching algorithm is not hard to implement if a fully constructed knowledge base is available, the actual difficulty of applying Dick's proposed system in real life lies in how to automatically extract information and reasoning procedure from a text written in natural languages and transform them into corresponding conceptual representation. In (Dick, 1991a), she presented four example arguments and described in detail how she constructed conceptual graphs from them. It is not a straightforward task since it involves human judgement when ambiguity encountered, and more importantly, it needs a full understanding of the underlying logic of the case of interest. Given the large volume of case texts, it cannot be trivial to construct a knowledge base containing transformed cases in an automatic matter.

1.3.2 Brüninghaus and Ashley's legal reasoning and prediction

Similar to Dick, Brüninghaus and Ashley were also interested in how to best represent text cases for argument generation. However, unlike Dick, who was motivated by the particular needs of lawyers in legal searching, Brüninghaus and Ashley were inspired by the possibility of predicting the outcome of a new case based on the decisions of previous similar cases. (Brüninghaus and Ashley, 2003) presented a prototype system that can reason with text cases and predict the outcome based on its analysis of the generated argument.

Their system has two components: SMILE (SMart Index LEarner) (Brüninghaus and Ashley (1999), Brüninghaus and Ashley (2001)) and IBP (Issue-Based Prediction) (Brüninghaus and Ashley (2003)), in a pipeline fashion (see Figure 1.6).

SMILE

For argument representation, they followed the case indexing convention from the CATO system (Aleven and Ashley, 1997) — each case is marked up in terms of Factors (stereotypical fact patterns that represent the strengths or weaknesses in a plaintiff's claim), together with squibs and summaries of the case facts. Trained on 146 Trade Secret Law cases from CATO, SMILE is thus built to automatically assign applicable Factors using text classifiers, given raw text cases as input.

They found replacing individual names by roles in the case texts helped improve better text indexing; however, the F-Measures for the Factor assignment are still far from competent for commercial application: ranging from 0 to 0.7, with average of around 0.25.

IBP

IBP is a program predicts the outcome of case-based arguments, given Factor representation of a case, which is inspired by how courts make their decisions based on the facts in trade secret cases.

In its model, IBP relates Factors to issues, in order to determine which issues are raised in the case. Then IBP relies on its domain model to make predictions based on the combination of evidence from all applicable issues. If there are issue-related Factors for both the plaintiff and the defendant, IBP uses a case-based reasoning method based

on Hypo (Ashley, 1988) and CATO to determine which side is favored in this case.

By adding certain syntactic and semantic information, IBP has been shown to perform significantly better than machine learning methods and prediction methods based on the CATO program in prediction outcomes; however, the small size of its database still prevents it from being claimed as reliable prediction for legal practice.

Integration of IBP and SMILE

Figure 1.6[11] depicts the integration of IBP and SMILE: with a case text submitted as input, SMILE generates the Factors assigned to the case and feeds them into IBP. IBP then generates a case-based analysis and prediction for the case.

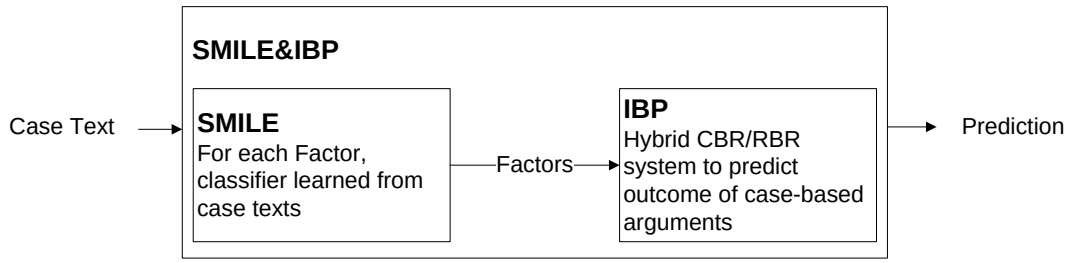


Figure 1.6: Integration of IBP and SMILE

Brüninghaus and Ashley's legal reasoning and prediction system extracts the essential legal factors from case texts, and generates human-readable arguments based on those factors. The outcome of a new case is thus predicted by reasoning upon those factors. Compared with Dick's argument representation and conceptual retrieval system, their system adopts a more coarse-grained processing of texts, since they only care about the fundamental factors which would eventually affect the case outcome, while Dick tends to take care of almost every legal aspect of the case, since even a small legal concept could appear in a lawyer's request later. In spite of the difference in

[11] This figure is reproduced from (Aleven and Ashley, 1997).

their focus, the fundamental methodology of Dick's system and Brüninghaus and Ashley's is very similar: in order to apply automatic concept retrieval (in Dick's system) or argument generation and outcome prediction (in Brüninghaus and Ashley's system), a well-developed knowledge base which contains represented previous cases in their corresponding forms is a must.

1.3.3 Mochales and Moens's argument mining

Similar to Dick and Brüninghaus and Ashley, Mochales and Moens are focused on arguments in the legal domain as well, given that automatic argument mining could be very effective and important to lawyers, since information searches constitutes a large amount of time for them. On one hand, Mochales and Moens are more ambitious because their goal is to filter out arguments from facts, backgrounds, and explanation in a given case text, and further to decompose arguments into their components: premises and conclusions; and on the other hand, their work might not be as useful as Dick's and Brüninghaus and Ashley's unless some postprocessing is applied to those extracted arguments For example, it might be necessary to compare the similarities between the premises or conclusions of the arguments in previous cases and the circumstances in the case that the lawyer currently searches, in order to decide which cases are relevant.

In Mochales and Moens's early work (Mochales and Ieven (2009), Mochales and Moens (2008), Mochales Palau and Moens (2009), Moens *et al.* (2007)), they focused on automatic detection of arguments in legal texts at the sentence level by classifying each sentence into argumentative or not. Each sentence is represented as a vector of shallow features including unigrams, bigrams, and trigrams; adverbs, verbs, and modal auxiliaries; all possible combinations of word pairs; sentence and word length; punctuation

and key words; and parse-tree depth and the number of subclauses. They trained two classifiers, a multinomial naïve Bayes classifier and a maximum entropy model, using the Araucaria corpus (see Section 1.2), and obtained a best average accuracy of 73.75% over 10-fold cross-validation by combining word-pair combinations, verbs, and sentence length using the multinomial naïve Bayes classifier (the baseline is 50%).

In their follow-up work (Palau and Moens, 2009), they focused on argumentative proposition classification: after first classifying the clauses of sentences as argumentative or not using the approaches in their previous work, they further classified each argumentative clause into a premise or conclusion. They used two corpora: AraucariaDB and a corpus of decisions from the European Court of Human Rights (ECHR). Using a support vector machine, classification of clauses as premise or conclusion attained an F_1 measure of 68.12% and 74.07%, respectively.

In addition, Mochales and Moens also designed a context-free grammar for parsing the argumentation structure of their texts, obtaining around 60% accuracy in detecting the argumentation structures while maintaining an F_1 measure around 70% for recognizing premises and conclusions[12].

Intuitively, the task of argument mining is very difficult, because given a text, there seems to be no straightforward way to identify whether there exists some logical relation between arbitrary adjacent propositions. However, since in their work, Mochales and Moens used only shallow features without involving much semantic processing, we cannot expect the overall performance of their argument detector and classifier to be impressively good.

[12]Strangely, the authors did not report the baseline in their paper, though they mentioned the distribution of the number of premise, conclusion, and non-argumentative sentences. However their argumentative proposition classifier is clause-based rather than sentence-based.

Our work is "downstream" from that of Mochales and Moens (the overall framework of our system is described in Section 1.4). Assuming the eventual success of their, or others', research on detecting and classifying the components of an argument, we seek to determine how the pieces fit together as an instance of an argumentation scheme. Moreover, we focus on general sources of arguments, and do not limit ourselves to legal texts.

1.3.4 Reed and Rowe's argumentation visualization tool: Araucaria

In unrestricted texts, the structure of an argument can be complex, constituted by multiple elementary propositions via different conveyance relations. Therefore, argumentation visualization software aiming for better presentation of argument structure has been developed in order to aid students or professionals in domains including law, philosophy, and education to understand the arguments more effectively. Examples of such software are: Athena[13], Rationale[14], Argunet[15], and Araucaria[16]. However, all these visualization tools are not automatic, which prevents them from going beyond analysis of small-scale data. Here we will discuss Araucaria in detail.

Araucaria (Reed and Rowe, 2004)[17] is an argument diagramming tool aimed at helping teaching and studying philosophical writing. Its diagramming is based on the Argument Markup Language (AML) formulated in XML (Reed and Rowe, 2001) (see Figure 1.4 in Section 1.2.2 for an example marked up in AML).

[13] `http://www.athenasoft.org/`

[14] `http://rationale.austhink.com/`

[15] `http://www.argunet.org/debates/`

[16] `http://www.computing.dundee.ac.uk/staff/creed/research/araucaria.html`

[17] Here Araucaria stands for a argument mark-up tool which can be used to create an argument diagram for a given text, not the dataset we discuss in Section 1.2, which contains already annotated arguments.

The process of creating an argument diagram is similar to that of creating a discourse tree in the following aspects, despite that in most cases not all textual units would participate in an argument diagram while in most cases all of them would constitute a discourse tree representing this text:

1. The elementary units in an argument diagram are ideally propositions, which usually appear as clauses in the text. It is also the case with elementary units in a discourse tree.

2. Though Reed and Rowe did not use the term "argumentation tree" to refer to argument diagram, the fundamental structure of an argument is also tree-like, just as the discourse structure.

3. The logical relations between propositions in an argument are strongly correlated with the discourse relation between them. For example, the existence of discourse relations such as Evidence, Motivation, and Justify [18] usually signal supporting reasons to the conclusion, and the existence of discourse relations such as Contrast and Antithesis normally signal refuting reasons to the conclusion.

Figure 1.7 shows the user interface of Araucaria. The user can load a text to analyze, which will appear in the left panel. Components of the argument can be selected by highlighting several text spans, each of which will comprise a node appearing in the right panel. Supporting or refuting relations are drawn by dragging a line from one node to another. Premises can be arranged in serial, convergent, or linked structures, and enthymemes (unstated premises) are denoted by grey nodes.

The support for argumentation schemes is one of the distinguishing features of Araucaria: the user can assign the type of inferential relations between premises and

[18] Here we follow the definitions in RST (Mann and Thompson, 1988).

conclusions, in order to more thoroughly understand the reasoning process. In addition, the user is free to insert critical questions beside each relation, and evaluate the strength of the argument based on whether the argument is able to exclude the exceptions raised in those questions.

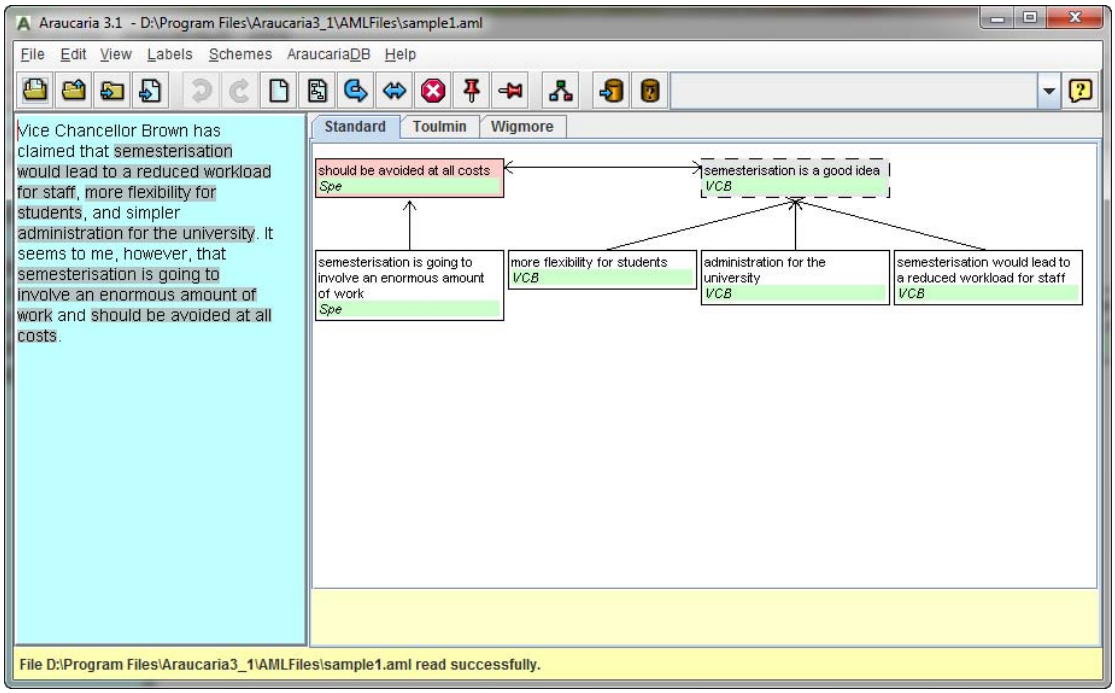


Figure 1.7: Araucaria screenshot

1.4 Overall framework of our work

Our ultimate goal is to reconstruct enthymemes, the unstated premises, in an argument by taking advantage of the stated propositions; and in order to achieve this goal we need to first determine the particular argumentation scheme that the argument is using. This problem is depicted in Figure 1.8.

Our scheme classifier is the dashed round-cornered rectangle portion of this overall framework: its input is the extracted conclusion and premise(s) determined by an argument detector followed by a premise / conclusion classifier, given an unknown text

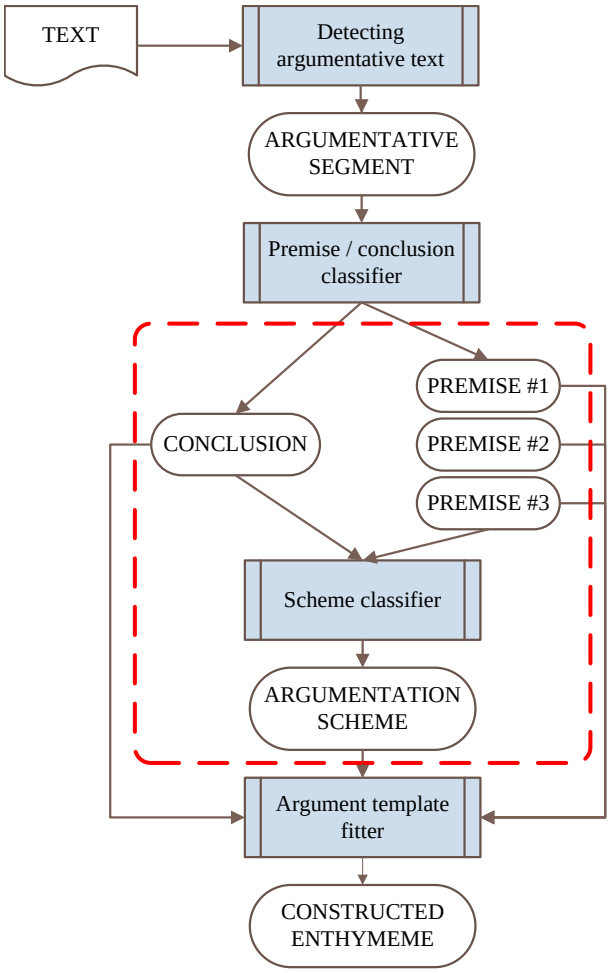


Figure 1.8: Overall framework of this research. This paper only focuses on the dashed round-cornered rectangle portion.

as the input to the entire system. And the portion below the dashed round-rectangle represents our long-term goal — to reconstruct the implicit premise(s) in an argument, given its argumentation scheme and its explicit conclusion and premise(s) as input. Since argument detection and classification are not the focus of this work, we assume here that the input conclusion and premise(s) have already been retrieved, segmented, and classified, as for example by the methods of Mochales and Moens (see Section 1.3.3 for details). The scheme template fitter is the topic of our on-going work. The scheme template is aimed at matching each proposition extracted by the premise / conclusion classifier to each slot in its corresponding scheme template, for example, match a premise of *practical reasoning* into major premise or minor premise.

Chapter 2

Methods

2.1 Data preprocessing

Recall from Chapter 1 that our ultimate goal is to reconstruct enthymemes in an argument by taking advantage of the stated propositions; and in order to achieve this goal we need to first determine the particular argumentation scheme that the argument is using. Since the output class domain of our classifier is limited to the five argumentation schemes as in Table 1.1, and the cues available for this classification task are limited to those in the explicitly stated conclusion and premises in an argument, we need to further tailor the data in AraucariaDB as follows to make it suitable for our purpose.

- Extract all arguments annotated in accordance to Walton's scheme-set.
- Break each complex AU node into several simple AUs with one conclusion PROP node, followed by one or more premise PROP nodes, in which no PROP node has embedded AU nodes. For complex AU nodes, the text of the conclusion PROP node is extracted to form the PROPTEXT in the simple AU node.

Argumentation scheme	Occurrences	% Occurrences
Argument from example	149	37.9
Argument from cause to effect	106	27.0
Practical reasoning	53	13.5
Argument from consequences	44	11.2
Argument from verbal classification	41	10.4
Total	393	100.0

Table 2.1: Occurrences of the five most frequent argumentation schemes in the preprocessed dataset

- Extract, from the simple arguments generated in the previous step, every argument whose scheme falls into one of the five most frequent schemes as described in Table 1.1.

- For each argument, remove all PROP nodes with "missing = yes", as these are enthymemes inserted by the annotator; our proposed classifier cannot, in real use, have any access to these. And we ignore any argument for which the conclusion PROP is missing.

The composition of the resulting preprocessed dataset is shown in Table 2.1.

In addition, we also test our classification system on arguments without those enthymemes removed (however, we still ignore any argument with the conclusion PROP missing) and view the performance difference compared to the accuracies of classifying the same arguments with no enthymemes involved. The motivation of this approach: Since the features we use to classify each argumentation scheme are based on our un-

conLoc	the location of the conclusion in the text.
premLoc	the location of the first premise proposition.
conFirst	whether the conclusion appears before the first premise proposition.
gap	the interval between the conclusion and the first premise proposition.
lenRat	the ratio of the length of the premise(s) to that of the conclusion.
numPrem	the number of explicit premise propositions (PROP nodes) in the argument.
type	type of argumentation structure, i.e., linked or convergent.

Table 2.2: List of general features

derstanding of its definition as shown in Table 1.1, with the annotated enthymemes reserved, the complete arguments should fit their corresponding schemes more properly, which allows us to better evaluate our understanding of these argumentation schemes and at the same time evaluate the effectiveness of the feature selection of our system.

2.2 Feature selection

The features used in our work fall into two categories: general features and scheme-specific features.

2.2.1 General features

General features are applicable to arguments belonging to any of the five schemes. As shown in Table 2.2, they include location features, length features, number of premise propositions, and type of argumentation structure.

Most of these features are concerned about the positions of propositions in the argument. The intuition motivating using these features is that we observe a distribution characteristic of different argumentation schemes, with regard to the position of its premise[1] compared with the position of its conclusion in the argument. In particular, the feature **conFirst** captures such a characteristic, as depicted in Figure 2.1, in which premFirst denotes the number of instances in which the premise appear before the conclusion and conFirst denotes the number of instances in which the conclusion appear before the (first) premise. We can see that for almost all these five schemes, the distribution of conFirst and premFirst instances is not balanced, especially for *argument from example*, it is extremely skewed and has a different majority as the other four schemes. Other position-related features are derivatives of **conFirst**, and for the features **conLoc**, **premLoc**, **gap**, and **lenRat**, we have two versions, differing in terms of their basic measurement unit: *sentence*-based and *token*-based.

Another important feature is **numPrem**, which is the number of explicit premise propositions (PROP nodes) in the argument. Although this feature is correlated with the final feautre **type** — it is impossible for convergent arguments to have more than on premise propositions, while strictly speaking linked arguments should have at least two premise propositions, it is not completely equal to **type**. Since for **numPrem**, we only count the number of explicit premises, it is highly possible that for a linked argument, its value of **numPrem** is one, with one or more premise left implicit by the speaker.

The final feature, **type**, indicates whether the premises contribute to the conclusion in a linked or convergent order. However, although this feature is available to

[1]In case there are more than one premise propisitions, we use the position of the first premise in the text.

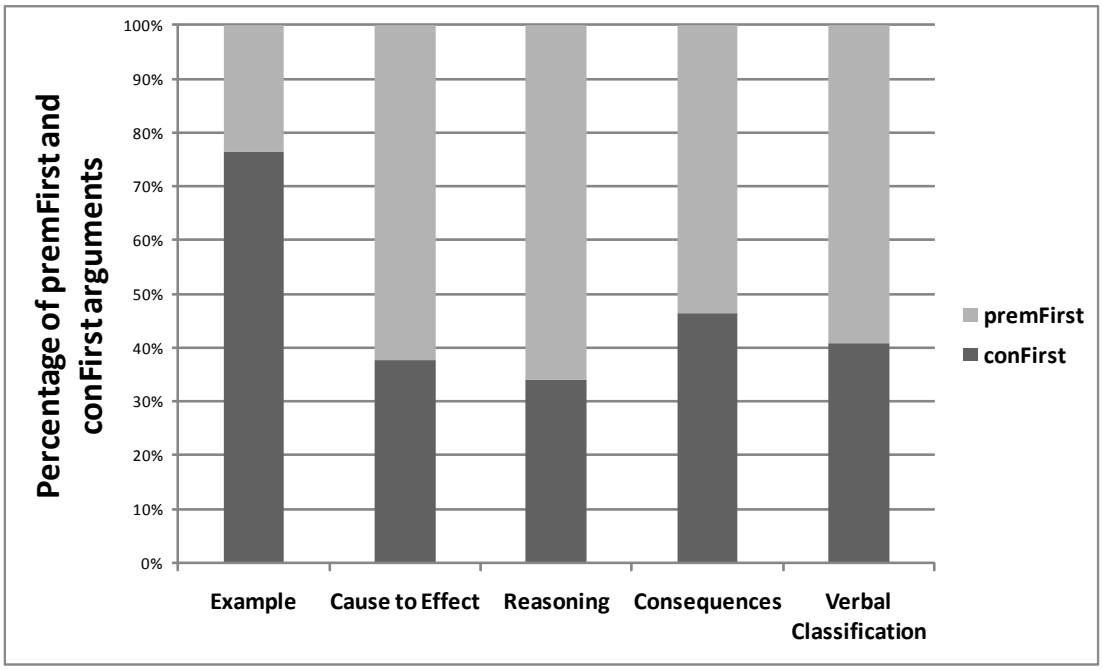


Figure 2.1: Distribution of premise-first arguments vs. conclusion-first arguments in the five most frequent schemes in Walton's scheme-set

us because it is included in the Araucaria annotations, its value cannot be obtained from raw text as easily as other features mentioned above; but it is possible that we will in the future be able to determine it automatically by taking advantage of some scheme-independent cues such as the discourse relation between the conclusion and the premises.

2.2.2 Scheme-specific features

Scheme-specific features are different for each scheme, since each scheme has its special cue phrases or patterns. Scheme-specific features for each scheme are shown in Table 2.3 (see Appendix A for complete lists of keywords and phrases). In our experiments in Chapter 3, all these features are computed for all arguments; but the features for any particular scheme are used only when it is the subject of a particular task. For example, when we classify *argument from example* in a one-against-others setup, we

use the scheme-specific features of that scheme for all arguments; when we classify *argument from example* against *argument from cause to effect*, we use the scheme-specific features of those two schemes.

Features for *argument from example*, *argument from cause to effect*, and *practical reasoning*

For these three schemes, the scheme-specific features are manually compiled cue phrases or patterns which are believed to be good indicators for each scheme. However, we should not treat all potential scheme indicators equally, since those cue phrases and patterns differ in their qualities in terms of precision and recall. For example, if we simply use the total number of cue phrase patterns specific to *practical reasoning* found in an argument to indicate the possibility that this argument belongs to *practical reasoning*, it will not be possible to differentiate two arguments $\mathcal{A}$ and $\mathcal{B}$, where $\mathcal{A}$ contains the keyword "should", which is a relatively common word, while $\mathcal{B}$ contains the keyword "aim", which is relatively rare and more indicative of the scheme. Thus, when incorporating a feature into the entire feature set, we take the distribution characteristics of the entire corpus into account: for each cue phrase or pattern, we compute "confidence", the degree of belief that the argument of interest belongs to a particular scheme, as described below.

For each argument $\mathcal{A}$, a vector $\mathbf{CV} = \{sf_1, sf_2, sf_3\}$ is added to its feature set, where each sf_i indicates the "confidence" c_i of the existence of the specific features associated with each of the first three schemes, $scheme_i$. This is defined in Equation 2.2.1:

$$c_i = \frac{1}{N} \sum_{k=1}^{m_i} (c_{ik} \cdot d_{ik}) \tag{2.2.1}$$

Argument from example 8 keywords and phrases including *for example*, *such as*, *for instance*, etc.; 3 punctuation cues: ":", ";", and "—".
Argument from cause to effect 22 keywords and simple cue phrases including *result*, *related to*, *lead to*, etc.; 10 causal and non-causal relation patterns extracted from WordNet (Girju, 2003).
Practical reasoning 28 keywords and phrases including *want*, *aim*, *objective*, etc.; 4 modal verbs: *should*, *could*, *must*, and *need*; 4 patterns including imperatives and infinitives indicating the goal of the speaker.
Argument from consequences The counts of positive and negative propositions in the conclusion and premises, calculated from the *General Inquirer*[1]: **conPos**, **conNeg**, **prePos**, and **preNeg**.
Argument from verbal classification The maximal similarity between the *central word* pairs extracted from the conclusion and the premise; the counts of *copula*, *expletive*, and *negative modifier* dependency relations returned by the *Stanford parser*[2] in the conclusion and the premise: **conCop**, **conExp**, **conNeg**, **preCop**, **preExp**, and **preNeg**.

[1] `http://www.wjh.harvard.edu/~inquirer/`

[2] `http://nlp.stanford.edu/software/lex-parser.shtml`

Table 2.3: List of scheme-specific features

Here m_i is the number of scheme-specific cue phrases designed for $scheme_i$ (for example, $m_1 = 11$ is the number of scheme-specific features for *argument from example*); c_{ik} is the prior probability $P(scheme_i|cp_k)$ that the argument $\mathcal{A}$ actually belongs to $scheme_i$, given that some particular cue phrase cp_k is found in $\mathcal{A}$; d_{ik} is a value indicating whether cp_k is found in $\mathcal{A}$; and N is the normalization factor, which is the number of scheme-specific cue phrase patterns designed for $scheme_i$ with at least one support (at least one of the arguments belonging to $scheme_i$ contains that cue phrase). There are two ways to calculate d_{ik}, *Boolean* and *count*, as shown in Equations 2.2.2 and 2.2.3, where *matchCnt* is the number of times $\mathcal{A}$ matches cp_k.

$$d_{ik} = \begin{cases} 1 & \text{if } \mathcal{A} \text{ matches } cp_k, \\ 0 & \text{otherwise.} \end{cases} \tag{2.2.2}$$

$$d_{ik} = \begin{cases} matchCnt & \text{if } \mathcal{A} \text{ matches } cp_k, \\ 0 & \text{otherwise.} \end{cases} \tag{2.2.3}$$

Features for argument from consequences

For *argument from consequences*, since the arguer has an obvious preference for some particular consequence (see its definition in Table 1.1), sentiment orientation can be a good indicator for this scheme, which is quantified by the counts of positive and negative propositions in the conclusion and premise(s): denoted as *conPos*, *conNeg*, *prePos*, and *preNeg* respectively. We also experiment with the following three *"sentiment strategies"* of feature selection for representing the arguer's sentiment orientation:

- **c_only**: {*conPos* + *conNeg*}

  In **c_only** sentiment strategy, the counts of positive and negative propositions in

the conclusion are combined as a single number in the entire feature set.

- **cp_combined**: {*conPos* + *prePos*, *conNeg* + *preNeg*}

  In **cp_combined** sentiment strategy, we use the two counts of positive and negative propositions in the argument to represent the arguer's orientation.

- **cp_split**: {*conPos*, *conNeg*, *prePos*, *preNeg*}

  In **cp_split** sentiment strategy, the four counts are treated as separate features in the entire feature set.

We also tried using the "scaled" proposition counts returned by *General Inquirer* instead of using the "raw" counts in our experiments. The "scaled" proposition counts take into account the total numbers of different kinds of propositions in the text, and scales the number of a particular kind of proposition (e.g., positive proposition) with it.

Features for argument from verbal classification

For *argument from verbal classification*, there exists a hypernymy-like relation between some pair of propositions (entities, concepts, or actions) located in the conclusion and the premise respectively. The existence of such a relation is quantified by the maximal similarity between the *"central word"* pairs extracted from the conclusion and the premise. We parse each sentence of the argument with the Stanford dependency parser, and a word or phrase is considered to be a *central word* if it satisfies either of the following conditions, which basically represents the attribute or the action of an entity in a sentence, or the entity itself:

- It is the dependent of any of the following dependency relations: *acomp*, *agent*,

appos, *attr*, *dobj*, *iobj*, *nsubj*, *nsubjpass*, *pobj*, *xcomp*.

- It is the governor of any of the following dependency relations: *agent*, *appos*, *cop*, *dobj*, *iobj*, *nsubj*, *nsubjpass*, *xsubj*[4].

Potentially, a word or a phrase satisfying the requirement above not only represents the core concept conveyed in the sentence, but also is able to act as the proposition that can be "classified" to a similar or more general concept in another sentence according to the definition of *argument from verbal classification*. It can be better explained by the argument presented in the third row of Table 2.4, which is generated from the corresponding scheme template for each component (shown in the first row) by appropriate entity substitution relations (shown in the second row).

The Stanford typed dependencies found in its individual premise and conclusion are:

1. *nsubj*(help, she)
2. *cop*(nice, is)

And the classification relation lies between the governor (help $\rightarrow$ property F) of the first dependency (*nsubj*), and the governor (nice $\rightarrow$ property G) of the second dependency (*cop*). Thus, this classification relation will be extracted by our approach described above.

The maximal similarity between the *"central word"* pairs extracted from the conclusion and the premise is calculated as below:

Two lists **con_cent_words** $\{cw_1, cw_2, \ldots, cw_n\}$ and **pre_cent_words** $\{pw_1, pw_2, \ldots, pw_m\}$ are constructed, in which each cw_i or pw_j is extracted following the criteria above. Then

[4]See Appendix B for definitions and examples of these dependency relations.

	Individual premise	**Classification premise**	**Conclusion**
Template	a has a particular property F.	For all x, if x has property F, then x can be classified as having property G.	Therefore, a has property G.
Substitution	$a \rightarrow$ she $F \rightarrow$ helps others	$x \rightarrow$ people $G \rightarrow$ being nice	$a \rightarrow$ she $G \rightarrow$ being nice
Example	She always helps others.	For all people, if they always help others, then they can be classified as being nice.	Therefore, she is nice.

Table 2.4: An example of argument from verbal classification generated from the scheme template

the similarity between these two lists is used to represent the maximal similarity between the *"central word"* pairs extracted from the conclusion and the premise, as in Equation 2.2.4:

$$sim = \max_{\substack{1 \le i \le n \\ 1 \le j \le m}} \left(jcn\_sim \left(synset_i, synset_j \right) \right) \tag{2.2.4}$$

where

$$synset_i \in \bigcup_{k=1}^{n} \mathbf{S_k}, synset_j \in \bigcup_{l=1}^{m} \mathbf{S_l} \tag{2.2.5}$$

Here $\mathbf{S_k}$ is all the synsets that cw_k belongs to in WordNet which bear the same part-of-speech as cw_k in the conclusion text; and $\mathbf{S_l}$ is all the synsets that pw_l belongs to in WordNet which bear the same part-of-speech as pw_l in the premise text.

Here we use Jiang-Conrath Similarity (Jiang and Conrath, 1997)[5] to compute the similarity between two synsets. Jiang-Conrath Similarity (*jcn_sim*) is an information content–based semantic distance measure that exploits the WordNet semantic network. Jiang and Conrath incorporated the notion of the *lowest common subsumer* or *lowest super-ordinate* (*lso*) of two target concepts in WordNet, as proposed in Resnik (1995). Intuitively, the semantic similarity between the two concepts is directly proportional to how specific the *lso* is — the more general the lso is, the larger the semantic distance between the target nodes. For two concepts c_1 and c_2, their semantic distance is defined in Equation 2.2.6, and their semantic similarity is simply the inverse of their semantic distance. The first bracketed part represents how dissimilar the target concept c_1 is from the *lso*, and the second bracketed part represents how dissimilar the target concept c_2 is from the *lso*. The semantic distance between c_1 and c_2 is thus the sum of these two

[5]Implemented by NLTK 2.0b8 (`http://www.nltk.org/download`)

terms.

$$jc\_dist(c_1, c_2) = \underbrace{IC(c_1) - IC(lso(c_1, c_2))}_{1} + \underbrace{IC(c_2) - IC(lso(c_1, c_2)}_{2} \tag{2.2.6}$$

where $IC(c)$ stands for the information content of a concept c, which is calculated by Equation 2.2.7:

$$IC(c) = -\log_p c \tag{2.2.7}$$

In addition, an arguer tends to use several particular syntactic structures (*copula*, *expletive*, and *negative modifier*) when using this scheme, which can be quantified by the counts of those special relations in the conclusion and the premise(s): denoted as *conCop*, *conExp*, *conNeg*, *preCop*, *preExp*, and *preNeg* respectively, and be incorporated into a feature vector in the following three "*dependency relation*" strategies:

- **single**: conCop + conExp + conNeg + preCop + preExp + preNeg
- **cp_combined**: conCop + preCop, conExp + preExp, conNeg + preNeg
- **cp_split**: conCop, conExp, conNeg, preCop, preExp, preNeg

Chapter 3

Experiments

3.1 Training

We experiment with two kinds of classification: one-against-others and pairwise[1]. We build a pruned C4.5 decision tree[2] (Quinlan, 1993) for each different classification setup, implemented by Weka Toolkit 3.6 (Hall *et al.*, 2009).

3.1.1 One-against-others classification

A one-against-others classifier is constructed for each of the five most frequent schemes, using the general features and the scheme-specific features for the scheme of interest: for example, when we classify *argument from example* in a one-against-others setup, we

[1]We did not fully implement five-way classification for this task. Since our preliminary attempt at five-way classification revealed relatively low accuracies, which might the result of a number of complex causes, we decided to use the two kinds of classification described in this paper to better understand the strength and weakness of our methods.

[2]We also tried support vector machines in our experiments, but their performance was always inferior to that of C4.5 with the same experimental setup.

use the scheme-specific features of only that scheme plus other general features for all arguments. For each classifier, there are two possible outcomes: *target_scheme* and *others*. One-against-other classification thus tests the effectiveness of each scheme's specific features.

3.1.2 Pairwise classification

A pairwise classifier is constructed for each of the ten possible pairings of the five schemes, using the general features and the scheme-specific features of the two schemes in the pair: for example, when we classify *argument from example* against *argument from cause to effect*, we use the scheme-specific features of only those two schemes plus other general features. For each of the ten classifiers, the training dataset is divided into arguments belonging to $scheme_1$ and arguments belonging to $scheme_2$, where $scheme_1$ and $scheme_2$ are two different schemes among the five. Pairwise classification thus tests the differentiating capability of each scheme's specific features.

3.1.3 Data preparation

The distribution of the five most frequent schemes is relatively skewed (149, 106, 53, 44, and 41 respectively belong to the five schemes in the order shown in Table 1.1), which would cause difficulty for the baseline evaluation approach of our classification system. For example, without any further processing, when we classify *argument from verbal classification*-against-*others*, the baseline will be nearly 90%; thus, even a high classification accuracy such as 95% might not be actually interesting. Of course, sampling in the already rather small dataset will inevitably result in even fewer available training data, but we are trying to find a tradeoff between fewer training data and more precise

target_scheme	*# of arguments*
Argument from Example	232
Argument from Cause to Effect	152
Practical Reasoning	72
Argument from Consequences	72
Argument from Verbal Classification	72

Table 3.1: The number of arguments used in each one-against-others classification experimental setup

evaluation.

Therefore, for each of our experimental setups (one-against-others or pairwise classification setup), we create 10 pools of randomly sampled data from the full corpus, each of which has the same size and the same baseline of 50%, and the number of samples belonging to each scheme is proportional to the total number of arguments belonging to that scheme in the dataset. The training-evaluation procedure will be carried out on each individual pool of data for once. Table 3.1 and Table 3.2 list the number of arguments used in each one-against-others and pairwise classification experimental setup.

3.2 Evaluation

We experiment with different combinations of general features and different strategies of incorporating scheme-specific features (discussed in Section 2.2): 4 different combinations for *argument from example*, *argument from cause to effect*, and *practical reasoning* each; 12 different combinations for *argument from consequences* and 6 for *argument from*

	example	*cause to effect*	*reasoning*	*consequences*
cause to effect	126			
reasoning	62	62		
consequences	48	48	48	
classification	52	52	52	48

Table 3.2: The number of arguments used in each pairwise classification experimental setup

verbal classification. The entire set of possible choices for each different strategy is listed below:

- **dependency**: *c_only*, *cp_combined*, and *cp_split* — the strategy of combining special syntactic structures, only applicable to *argument from verbal classification*.
- **sentiment**: *c_only*, *cp_combined*, and *cp_split* — the strategy of combining sentiment proposition counts, only applicable to *argument from consequences*.
- d_{ik}: *Boolean* or *count* — the strategy of combining scheme-specific cue phrases or patterns using either Equation 2.2.2 or 2.2.3 for d_{ik}, applicable to *argument from cause to effect*, *argument from example*, and *practical reasoning*.
- **base**: *sentence* or *token* — the basic unit of applying location- or length- related general features, applicable to all five schemes.

For each experiment, the training-and-evaluation procedure is repeated 10 times over all the 10 pools of randomly sampled data using 10-fold cross-validation[3] — once

[3]We use J48 classifier in Weka Toolkit 3.6 for this training-and-evaluation procedure, with command line "-N 10 -R"

for each individual pool with baseline at 50%. The 10-point average accuracy is used as the evaluation metric.

Chapter 4

Results

We first present the best average accuracy (BAA) of each classification setup. Then we demonstrate the impact of the feature **type** (convergent or linked argument) on BAAs for different classification setups, since we believe **type** is strongly correlated with the particular argumentation scheme (see discussion in Section 1.2.3) and its value is the only one directly retrieved from the annotations of the training corpus. Last, we present the comparison between BAAs of classification using arguments without enthymemes as training data and those using arguments with annotated enthymemes.

4.1 BAAs of each classification setup

4.1.1 BAAs of one-against-others classification

Table 4.1 summarizes the best average accuracies of one-against-others classification for each of the five most frequent schemes. The subsequent four columns list the particular strategies of feature incorporation under which those BAAs are achieved (see Section 3.2 for detailed explanations of these choices), and the entire possible choices

target_scheme	*BAA*	*sentiment*	d_{ik}	*base*	*dependency*
example	90.6	–	–	count	token
cause to effect	70.4	–	–	Boolean/count	token
reasoning	90.8	–	–	count	sentence
consequences	62.9	–	cp_split	–	sentence
classification	63.2	cp_combined	–	–	token

Table 4.1: Best average accuracies (BAAs)(%) of one-against-others classification

and their corresponding average accuracies of each scheme as *target_scheme* are given in Tables C.1 – C.5 in Appendix C:

As Table 4.1 shows, one-against-others classification achieves great accuracy with respect to the two argumentation schemes: *argument from example* and *practical reasoning* — 90.6% and 90.8%, while the BAA of *argument from cause to effect* is only just over 70%. However, for the last two schemes (*argument from consequences* and *argument from verbal classification*), accuracy is only in the low 60s (in the former case using *cp_split* for combining special dependency relations and in the latter using *cp_combined* for combining sentiment proposition counts); there is little improvement of our system over the majority baseline of 50%. This is probably due at least partly to the fact that these schemes do not have such obvious cue phrases or patterns as the other three schemes, and also because the available training data for each is relatively small (44 and 41 instances, respectively). The BAA of each scheme is achieved with inconsistent choices of base and d_{ik}, but the accuracies that resulted from different choices vary only by very little (see Tables C.1 – C.5 for details).

4.1.2 BAAs of pairwise classification

Table 4.2 shows that our system is able to correctly differentiate between most of the different scheme pairs, with accuracies as high as 98% (indicated in boldface). It has poor performance (64%) (indicated by underline) only for the pair *argument from consequences* and *argument from verbal classification*; perhaps not coincidentally, these are the two schemes for which performance was poorest in the one-against-others task. Tables C.6 – C.15 in Appendix C list the possible choices and their corresponding average accuracies of each possible scheme pairing.

One interesting result observed in Tables C.11 and C.14 is that when our classifier is able to differentiate two argumentation schemes (in this case, *argument from consequences* versus *practical reasoning* and *argument from verbal classification* versus *practical reasoning*) with extremely high accuracy (97.9% and 98.3%), there is no difference between various feature incorporation strategies in terms of the average accuracies. On one hand, it indicates that the cue phrases we utilize as scheme-specific features are sufficient in disambiguate between these scheme pairings, which explains why different choices of basic units in position-related features do not effect the performance; and on the other hand, it shows that those cue phrases can appear in a single proposition at most once, so different strategies of d_{ik} (**count** or **Boolean**) do not effect the performance either.

4.2 Impact of type on classification accuracy

As we can see from Tables 4.3 and 4.4, for both one-against-others and pairwise classifications, incorporating **type** into the feature vectors improves classification accuracy in

	example	*cause to effect*	*reasoning*	*consequences*
cause to effect	80.6			
reasoning	93.1	94.2		
consequences	86.9	86.7	**97.9**	
classification	86.0	85.6	98.3	64.2

Table 4.2: Best average accuracies (BAAs)(%) of pairwise classification

most cases: the only exception is that the best average accuracy of one-against-others classification between *argument from cause to effect* and other is obtained without involving **type** into the feature vector — but the difference is negligible, i.e., 0.5 percentage points with respect to the average difference. In terms of average difference, **type** has most significant impact on the two schemes **argument from example** (14.7 points) and **practical reasoning** (7.2 points), while it has relatively small impact on the other three schemes. We speculate that this might be the result of two possible causes:

1. These two schemes are also the two schemes which have most unbalanced linked and convergent argument distribution and are in different polarity (*argument from example* has many more convergent arguments than linked arguments, while *practical reasoning* has many more linked arguments than convergent arguments). Therefore, intuitively, adding **type** would increase the capacity to differentiate these two schemes with other schemes.

2. Best average accuracies of the other three schemes are around 60% to 70% in their respective one-against-others experiments, which suggests that those scheme-specific features might not be very effective to characterize the distinct properties of those schemes; and sometimes they might even mislead our classifier — in

such case, even if **type** is indeed a valuable feature, its usefulness would not be so obvious when combined with other features.

Similarly, for pairwise classifications, as shown in Table 4.4, **type** has significant impact on BAAs, especially on the pairs of *practical reasoning* versus *argument from cause to effect* (17.4 points), *practical reasoning* versus *argument from example* (22.8 points), and *argument from verbal classification* versus *argument from example* (20.2 points), in terms of the maximal differences; but it has a relatively small impact on *argument from consequences* versus *argument from cause to effect* (0.8 point), and *argument from verbal classification* versus *argument from consequences* (1.1 points), in terms of average differences. We can see that, for a given scheme pair, when at least one them is *argument from example* or *practical reasoning* — the two schemes whose BAAs in one-against-others classification differ the most with or without **type** — incorporating **type** leads to significant accuracy improvement (7.1 – 20.1 points average differences), except for *argument from consequences* versus *practical reasoning* (0.8 points average difference), whose best average accuracy is already extremely high (97.9%) without **type**.

4.3 Impact of incorporating annotated enthymemes into training

As mentioned in Section 2.1, we also test our classification system on arguments without their annotator-supplied enthymemes removed, and here we present the performance difference compared to the accuracies of classifying the same arguments with no enthymemes involved.

According to Table 4.5 and 4.6, as far as the best average accuracies (BAAs) are

target_scheme	***BAA-t***	***BAA-no t***	***max diff***	***min diff***	***avg diff***
example	90.6	71.6	22.3	10.6	14.7
cause to effect	70.4	70.9	−0.5	−0.6	−0.5
reasoning	90.8	83.2	8.1	7.5	7.7
consequences	62.9	61.9	7.5	−0.6	4.2
classification	63.2	60.7	2.6	0.4	2.0

Table 4.3: Accuracy (%) comparisons between with and without **type** in one-against-others classification. *BAA-t* is best average accuracy with **type**, and *BAA-no t* is best average accuracy without **type**. *max diff*, *min diff*, and *avg diff* are maximal, minimal, and average differences between each experimental setup with **type** and without **type** while the remaining conditions are the same.

$scheme_1$	$scheme_2$	***BAA-t***	***BAA-no t***	***max diff***	***min diff***	***avg diff***
cause to effect	example	80.6	69.7	10.9	7.1	8.7
reasoning	example	93.1	73.1	22.8	19.1	20.1
reasoning	cause to effect	94.2	80.5	17.4	8.7	13.9
consequences	example	86.9	76.0	13.8	6.9	10.1
consequences	cause to effect	87.7	86.7	3.8	−1.5	−0.1
consequences	reasoning	97.9	97.9	10.6	0.0	0.8
classification	example	86.0	74.6	20.2	3.7	7.1
classification	cause to effect	85.6	76.8	9.0	3.7	7.1
classification	reasoning	98.3	89.3	8.9	4.2	8.3
classification	consequences	64.0	60.0	6.5	−1.3	1.1

Table 4.4: Accuracy (%) comparisons between with and without **type** in pairwise classification. Column headings have the same meanings as in Table 4.3.

concerned, for most classification setups, there is no significant difference between incorporating enthymemes into training or not, except for *argument from consequences*-against-*others* (17.9 points). But there are several points worth discussion:

As shown in Table 4.5, for *argument from verbal classification*-against-others classification — one of the two one-against-others classifications (the other one is *argument from consequences*-against-others) with relatively poor results, incorporating enthymemes helps boost the performance a lot: from 62.9% to 90.8%, while the other four classifications suffer more or less performance reduction with enthymemes involved. We speculate that the reason for the significant difference between classification with and without enthymemes for *argument from verbal classification*-against-*others* is that with enthymemes involved, it should be easier for us to identify the "classification" part, since it usually belongs to common world knowledge, which is thus normally not stated explicitly. This suggests that it is promising to improve the performance on this classification setup with enthymemes excluded by incorporating more world knowledge. However, for the other four schemes, especially *argument from example*, where shallow lexical features are indicative enough, incorporating ethymemes might involve more noise into the feature set as well, which in our speculation explains the decreased accuracy in the other four experimental setups.

As shown in Table 4.6, for pairwise classifications, almost every pairing involving *argument from cause to effect* or *argument from example* suffers performance reduction, which is correlated with their decreased accuracies in one-against-others classifications; and the increased accuracy of *argument from classification* versus *argument from consequences* is also consistent with the boosted performance of *argument from classification*-against-others.

target_scheme	*BAA-no enthy*	*BAA-enthy*	*difference*
example	90.6	83.3	−7.3
cause to effect	70.4	69.2	−1.2
reasoning	90.8	88.2	−2.6
consequences	62.9	90.8	**7.9**
classification	63.2	63.0	−0.2

Table 4.5: Best average accuracy (BAA)(%) comparisons between with and without incorporating annotated enthymemes into training in one-against-others classification, *BAA-no enthy* and *BAA-enthy* are the best average accuracies without and with incorporating annotated enthymemes into training.

$scheme_1$	$scheme_2$	*BAA-no enthy*	*BAA-ethy*	*difference*
cause to effect	example	80.6	78.7	−1.9
reasoning	example	93.1	90.8	−2.3
reasoning	cause to effect	94.2	82.3	**−11.9**
consequences	example	86.9	78.7	−8.2
consequences	cause to effect	87.7	79.6	−8.1
consequences	reasoning	97.9	98.5	0.6
classification	example	86.0	87.4	1.4
classification	cause to effect	85.6	80.3	5.3
classification	reasoning	98.3	98.5	−0.2
classification	consequences	64.2	70.7	6.5

Table 4.6: Best average accuracy (BAA)(%) comparisons between with and without incorporating annotated enthymemes into training in pairwise classification. Column headings have the same meanings as in Table 4.5.

However, it is relatively surprising to find out that for most classification setups, the accuracies actually decrease instead of increase with annotated enthymemes incorporated into training. Intuitively, if our understanding of each argumentation scheme is correct and our feature selection is able to capture the characteristics of each scheme, with enthymemes incorporated, the complete arguments should fit their corresponding schemes more properly, and thus higher accuracies should be expected. This could be a result of both of the following two aspects:

- First, though many enthymemes inserted by the annotators match their corresponding scheme template properly (such as Example 7), some annotator-supplied enthymemes might not be exactly the "properly fitting" enthymemes which would fill up the complete argumentation scheme template perfectly.

  Moreover, if the annotated argumentation scheme is wrong itself, and the inserted enthymemes will make the argument fit the correct scheme (but not the annotated one) more properly; thus our system will falsely treats it as a classification error. For example, the argument shown in Example 5 in Chapter 1 is falsely classified as *argument from example* instead of *argument from verbal classification*; and by inserting the enthymeme, the complete argument fits the argumentation template of *argument from verbal classification* almost perfectly, which unfairly deteriorates the classification accuracy by accident.

  Example 7. ***An example of high-grade annotator-supplied enthymeme***

  Individual premise*: A typical cluster bomb is a container that opens in mid-air and scatters up to 200 bomblets over an area the size of half a soccer field.*

  Classification premise (inserted by the annotator)*: Bombs that do not destroy only their target but involve in their effects the surrounding area are not precision weapons.*

***Conclusion**: Even in their new, "wind-corrected" form, cluster bombs are not precision weapons.*

- Secondly, the features used in our system are basically shallow features, which try to represent the semantic information in an argument by extracting its syntactic and lexical characteristics. Given richer information in an argument by incorporating enthymemes into training, those features may encounter more difficulty since more noise is also involved in this process.

4.4 Conclusions

The argumentation scheme classification system that we have presented in this paper introduces a new task in research of argumentation. To the best of our knowledge, this is the first attempt to classify argumentation schemes.

In our experiments, we have focused on the five most commonly used schemes in Walton's scheme-set, and conducted two kinds of classification with different feature incorporation strategies: one-against-others classification and pairwise classification. In one-against-others classification, we achieved over 90% best average accuracies for the two schemes *argumentation from example* and *practical reasoning*; and in pairwise classification, we are able to obtain 80% to 90% best average accuracies for most scheme pairs. The performance of our classification system on the two schemes *argument from consequences* and *argument from verbal classification* are relatively lower, possibly due to the lack of sufficient training examples and the insufficient semantic information captured by our features. In the middle was the performance on *argument from cause to effect*; we speculate that it might be improved by employing alternative causal relation patterns.

We also have compared classification performance between using the feature **type** or not. In most experimental setups (one-against-others or pairwise classification), incorporating **type** actually improves accuracy, which is consistent with our intuition motivating the use of **type** as one of our features. In one-against-others classification, for *argument from example* and *practical reasoning*, incorporating **type** helps boost performance a lot: 14.7 and 7.7 points on average; but there is no significant difference for the other three schemes, in terms of average differences. The reason why **type** has inconsistent impact on different scheme might be two-fold: on one hand, the five schemes are not all equivalent with respect to the distribution of linked and convergent arguments, so the effectiveness of **type** is thus inequivalent for different schemes; and on the other hand, since the scheme-specific features of the other three schemes (*argument from cause to effect*, *argument from consequences*, and *argument from verbal classification*) are more difficult to distinguish from others, it might add complexity to the evaluation of **type**'s effectiveness. In pairwise classification, the impact of **type** is consistent to that in one-against-others: when either *argument from example* or *practical reasoning* is in a particular scheme pair, incorporating **type** leads to a large performance boost, except for *argument from consequences* versus *practical reasoning*, whose best average accuracy stays almost the same after **type** incorporated, because it is already able to achieve extremely high accuracy without **type** (97.9%).

Moreover, we have analyzed the influence of enthymemes in classifying argumentation schemes, by comparing the best average accuracy differences between using annotator-supplied enthymemes in training or not. The results are counter-intuitive, because for most classification setups, incorporating enthymemes into training decreases performance rather than increases it. Especially for those schemes — *argument from example* and *practical reasoning* — on which our classifier is able to achieve

high accuracy without annotator-supplied enthymemes, their best average accuracies of one-against-others classification suffer greater reduction (7.3 and 2.6 points); however, for *argument from verbal classification*, one of the two schemes on which our classifier can only achieve slightly better accuracies than the majority baseline without annotator-supplied enthymemes, incorporating enthymemes help boost the best average accuracy in one-against-others classification (7.9 points). The comparison results of pairwise classification are basically consistent with the situation in one-against-others: most scheme pairs involving *argument from verbal classification* are subject to improved best average accuracies, while other pairs generally experience reduced accuracies. Such counter-intuitive results can be caused by a number of reasons; however, we speculate the following two might be the most possible ones:

1. As discussed in Section 1.2.4, the correctness of annotation in Araucaria is not guaranteed, since the task of constructing missing premises is particularly difficult, the quality of those annotator-supplied enthymemes may not be reliable enough for a fair evaluation on the impact of enthymemes. Especially when an annotator made a mistake in assigning scheme to an argument in the first place, incorporating enthymemes might push our classifier toward a wrong direction even more.

2. For those schemes whose best average accuracies are already above 80s in their one-against-others classifications (*argument from example* and *practical reasoning*), since the general and scheme-specific features are sufficiently effective to characterize those schemes even based on only explicitly stated propositions, adding enthymemes into the equation may import unnecessary noise to our classifier as well.

Chapter 5

Future Work

In this chapter, we will discuss several potential extensions to our current work.

5.1 Automatic classification of the feature type

In future work, we will look at automatically classifying **type** (i.e., linked or convergent argument), as **type** is the only feature directly retrieved from annotations in the training corpus at the moment and it has a strong impact on improving classification accuracies (see Section 4.2). Though the automatic classification of **type** is not trivial, we believe that it will be possible to do so using only the available input — the segmented explicit conclusion and premise of an argument — as in our preliminary system.

One potential approach to automatic classification of **type** is to explore the discourse relations between the explicit conclusion and premise(s) in an argument. Recall the definitions of linked argument (LA) and convergent argument (CA) as in Section 1.1.4:

- A *linked argument* (LA) has two or more inter-dependent premise propositions, all

of which are necessary to make the conclusion valid.

- A *convergent argument* (CA) has exactly one premise proposition, which is sufficient to make the conclusion valid.

We propose an ideal criterion for classifying **type**:

We assume that given an ideal discourse parser, an argumentative text has been segmented into a sequence of elementary discourse units (edus) and all discourse relations between these discourse unit spans have been analyzed, then: if there exists a *partial cross rhetorical relation* (PCRR), then this is a *LA*; otherwise, it is a *CA*. We define *partial cross rhetorical relation* as in Definition 2:

Definition 2. *A **partial cross rhetorical relation** occurs if either of the following two cases holds:*

1. *The nucleus of this relation covers a proper subset of conclusion edus, and the satellite covers a subset of premise edus.*

2. *The satellite of this relation covers a proper subset of conclusion edus, and the nucleus covers a subset of premise edus.*

In our preliminary exploration using Hilda_0.9.5[1] (duVerle and Prendinger, 2009) as the discourse parser, this criterion is able to achieve 100% precision but very low recall. The resulting low accuracy is partly due to the fact that progress in discourse parsing is still far from "ideal" as in our assumption. Moreover, our manual study of the mistakes made by our criterion suggests that employing only discourse relations may not be sufficient, more semantic information needs to be incorporated as well, such

[1] `http://nlp.prendingerlab.net/hilda/`

as which part of the conclusion is sufficiently supported by which premise proposition, but this is beyond the state of the art in computational linguistics.

5.2 Argumentation scheme template fitting

As depicted in Figure 1.8, the component following the argumentation scheme classifier is the template fitter, whose function is to map each explicitly stated conclusion and premise into the corresponding position as in its scheme template and to extract necessary information for enthymeme reconstruction. This stage should largely involve world knowledge in order to realize the expected goal. For example, consider the following argument from our dataset, which uses *practical reasoning*:

Example 8. ***An example of*** **practical reasoning**

Premise*: The survival of the entire world is at stake.*

Conclusion*: The treaties and covenants aiming for a world free of nuclear arsenals and other conventional and biological weapons of mass destruction should be adhered to scrupulously by all nations.*

Definition 3. ***Practical reasoning***

Major premise*: I have a goal G.*

Minor premise*: Carrying out action A is a means to realize G.*

Conclusion*: Therefore, I ought (practically speaking) to carry out this action A.*

We should be able to fit the *Premise* and the *Conclusion* into the *Major premise* and the *Conclusion* in the definition of *practical reasoning*, and extract the necessary information to construct the following conceptual mapping relations:

1. *Survival of the entire world* $\longrightarrow$ a goal G

2. *Adhering to the treaties and covenants aiming for a world free of nuclear arsenals and other conventional and biological weapons of mass destruction* $\longrightarrow$ action A

Therefore we will be able to reconstruct the missing *Minor premise* in this argument:

> Carrying out *adhering to the treaties and covenants aiming for a world free of nuclear arsenals and other conventional and biological weapons of mass destruction* is a means to realize *survival of the entire world*.

Of course, some minor adjustment might be needed to make the generated premise more comprehensible, but this is not the focus of this study.

The example above explains how enthymemes in an argument can be constructed by first figuring out the particular argumentation scheme the argument is using and then applying template fitting to map the stated propositions into its corresponding template. The ultimate goal of our research is to present these reconstructed enthymemes to users such as lawyers and students studying philosophy, and allow them to evaluate the plausibility of the argument or prepare for attacks based on the reconstructed enthymemes. In this sense, we are not responsible for verifying the reliability of those reconstructed enthymemes; thus, purely syntactic information extraction and slot filling is sufficient here.

5.3 Enhancing the pipeline framework

Completion of our scheme classification system will be a step towards our ultimate goal of constructing the enthymemes in an argument by the procedure depicted in Figure 1.8. Because of the significance of enthymemes in reasoning and arguing, this is

crucial to the goal of understanding arguments. But given the still-premature state of research of argumentation in computational linguistics, there are many practical issues to deal with first, such as improving the general performance of each step in this entire procedure, and the construction of richer training corpora.

To prevent the errors made in the previous stages from being multiplied after going through the entire process, we may incorporate a feedback mechanism as an enhancement to the proposed pipeline framework: the result of later stages, such as the classified argumentation scheme, could be used to re-evaluate the extracted conclusion and premise(s).

References

Vincent Aleven and Kevin D. Ashley. Teaching case-based argumentation through a model and examples empirical evaluation of an intelligent learning environment. In *Artificial Intelligence in Education*, pages 87–94, Amsterdam, The Netherlands, 1997. IOS Press.

Kevin D. Ashley. *Modelling legal argument: reasoning with cases and hypotheticals*. PhD thesis, University of Massachusetts, Amherst, MA, USA, 1988.

Stefanie Brüninghaus and Kevin D. Ashley. Toward adding knowledge to learning algorithms for indexing legal cases. In *ICAIL '99: Proceedings of the 7th international conference on artificial intelligence and law*, pages 9–17, New York, NY, USA, 1999. ACM.

Stefanie Brüninghaus and Kevin D. Ashley. Improving the representation of legal case texts with information extraction methods. In *ICAIL '01: Proceedings of the 8th international conference on artificial intelligence and law*, pages 42–51, New York, NY, USA, 2001. ACM.

Stefanie Brüninghaus and Kevin D. Ashley. Predicting outcomes of case based legal arguments. In *ICAIL '03: Proceedings of the 9th international conference on artificial intelligence and law*, pages 233–242, New York, NY, USA, 2003. ACM.

Judith Dick. Conceptual retrieval and case law. In *Proceedings, First International Conference on Artificial Intelligence and Law*, pages 106–115, Boston, May 1987.

Judith Dick. *A conceptual, case-relation representation of text for intelligent retrieval*. PhD thesis, Faculty of Library and Information Science, University of Toronto, April 1991. Published as technical report CSRI-265.

Judith Dick. Representation of legal text for conceptual retrieval. In *Proceedings, Third International Conference on Artificial Intelligence and Law*, pages 244–252, Oxford, June 1991.

David duVerle and Helmut Prendinger. A novel discourse parser based on support vector machine classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 665–673, Suntec, Singapore, August 2009. Association for Computational Linguistics.

Roxana Girju. Automatic detection of causal relations for question answering. In *Proceedings of the ACL 2003 workshop on multilingual summarization and question answering*, pages 76–83, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

Oliver Wendell Holmes. The path of the law. *Harvard Law Review*, 110(5), 1997.

Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and

lexical taxonomy. In *International Conference Research on Computational Linguistics (ROCLING X)*, pages 19–33, 1997.

Joel Katzav and Chris Reed. A classification system for arguments. Technical report, Department of Applied Computing, University of Dundee, 2004.

Joel Katzav and Chris Reed. On argumentation schemes and the natural classification of arguments. *Argumentation*, 18(2):239–259, 2004.

William C. Mann and Sandra A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text – Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988.

Mara Miezitis. Generating lexical options by matching in a knowledge base. Master's thesis, Department of Computer Science, University of Toronto, October 1988. Published as technical report CSRI-217.

Raquel Mochales and Aagje Ieven. Creating an argumentation corpus: do theories apply to real arguments?: a case study on the legal argumentation of the ECHR. In *ICAIL '09: Proceedings of the 12th International Conference on Artificial Intelligence and Law*, pages 21–30, New York, NY, USA, 2009. ACM.

Raquel Mochales and Marie-Francine Moens. Study on the structure of argumentation in case law. In *Proceedings of the 2008 Conference on Legal Knowledge and Information Systems*, pages 11–20, Amsterdam, The Netherlands, 2008. IOS Press.

Raquel Mochales Palau and Marie-Francine Moens. Automatic argumentation detection and its role in law and the semantic web. In *Proceedings of the 2009 Conference on Law, Ontologies and the Semantic Web*, pages 115–129, Amsterdam, The Netherlands, 2009. IOS Press.

Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. Automatic detection of arguments in legal texts. In *ICAIL '07: Proceedings of the 11th International Conference on Artificial Intelligence and Law*, pages 225–230, New York, NY, USA, 2007. ACM.

Raquel Mochales Palau and Marie-Francine Moens. Argumentation mining: the detection, classification and structure of arguments in text. In *ICAIL '09: Proceedings of the 12th International Conference on Artificial Intelligence and Law*, pages 98–107, New York, NY, USA, 2009. ACM.

John L. Pollock. *Cognitive Carpentry: A Blueprint for How to Build a Person*. Bradford Books. The MIT Press, May 1995.

J. Ross Quinlan. C4.5: Programs for machine learning. *Machine Learning*, 16(3):235–240, 1993.

Chris Reed and Glenn Rowe. Araucaria: Software for puzzles in argument diagramming and XML. Technical report, Department of Applied Computing, University of Dundee, 2001.

Chris Reed and Glenn Rowe. Araucaria: Software for argument analysis, diagramming and representation. *International Journal of Artificial Intelligence Tools*, 14:961–980, 2004.

Philip Resnik. Using information content to evaluate semantic similarity in a taxanomy. In *Proceedings of the 14th International Joint Conference for Artificial Intelligence (IJCAI-95)*, pages 448–453, 1995.

Harold Somers. *Valency and Case in Computational Linguistics (Edinburgh Information Technology)*. Edinburgh University Press, 1987.

J. F. Sowa. *Conceptual structures: information processing in mind and machine*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1984.

Frans H. van Eemeren and Rob Grootendorst. *Argumentation, Communication, and Fallacies: A Pragma-Dialectical Perspective*. Routledge, 1992.

Douglas Walton and Chris Reed. Argumentation schemes and defeasible inferences. In *Workshop on Computational Models of Natural Argument, 15th European Conference on Artificial Intelligence*, pages 11–20, Amsterdam, The Netherlands, 2002. IOS Press.

Douglas Walton, Chris Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

Appendices

Appendix A

Complete lists of scheme-specific features

Note: All keywords and phrases in the following tables are represented in regular expression format.

Keywords and phrases	Punctuation
(for)?example(s)?	:
(a)?like	;
such as	—
as well(as)?	
corresponding(ly)?	
(for)instance(s)?	

Table A.1: Scheme-specific features for *argument from example*

Keywords and phrases		Conditional patterns
according(ly)?	consequen((ce) | (tly))	Any modal verb following "if"
result(ing)?	so	Any modal verb following "when"
(be)?cause(of)?	ma(k | d)e it	if . . . going to
would be	further	when . . . going to
will be	fact(s)?	
since	explain(s | (ed))?	
effect	in th(e | (at))? case	
affect(s)?	suppose	
beg(i | a)n to	thus	

Table A.2: Scheme-specific features for *argument from cause to effect*

Keywords and phrases		**Imperative and infinitive patterns**
want	have to	. . . to VB . . .
solution	way to	[BOS][1]Do VB . . .
purpose	make sure	[BOS]Don't/Do not VB . . .
objective	allow	[BOS]VB . . .
goal	cause	**Modal verbs**
destination	sake	should
intention	desigining	need
aim	design	must
idea	allow	could
wish	recommend mind	doubt
intent	ensure	
terminus	choose	
choice	support	

[1]Beginning of the sentence

Table A.3: Scheme-specific features for *argument from cause to effect*

Appendix B

Definitions of selected Stanford typed dependencies

***acomp*: adjectival complement**[1]

An adjectival complement of a VP is an adjectival phrase which functions as the complement (like an object of the verb); an adjectival complement of a clause is the adjectival complement of the VP which is the predicate of that clause.

"She looks very beautiful" — *acomp*(looks, beautiful)[2]

***agent*: agent**

An agent is the complement of a passive verb which is introduced by the preposition "by" and does the action.

"The man was killed by the police" — *agent*(killed, police)

***appos*: appositional modifier**

An appositional modifier of an NP is an NP immediately to the right of the first NP that serves to define or modify that NP. It includes parenthesized examples.

[1] All definitions are copied from the manual of Stanford dependency parser.

[2] All Stanford dependencies are written as *abbreviated_relation_name*(governor, dependent).

"Sam, my brother" *appos*(Sam, brother)

***attr*: attributive**

An attributive is a WHNP complement of a copular verb such as "to be", "to seem", "to appear".

"What is that?" *attr*(is, What)

***cop*: copula**

A copula is the relation between the complement of a copular verb and the copular verb.

"Bill is big" *cop*(big, is)

***dobj*: direct object**

The direct object of a VP is the noun phrase which is the (accusative) object of the verb; the direct object of a clause is the direct object of the VP which is the predicate of that clause.

"She gave me a raise" *dobj*(gave, raise)

***iobj*: indirect object**

The indirect object of a VP is the noun phrase which is the (dative) object of the verb; the indirect object of a clause is the indirect object of the VP which is the predicate of that clause.

"She gave me a raise" *iobj*(gave, me)

***nsubj*: nominal subject**

A nominal subject is a noun phrase which is the syntactic subject of a clause. The governor of this relation might not always be a verb: when the verb is a copular verb, the root of the clause is the complement of the copular verb.

"Clinton defeated Dole" *nsubj*(defeated, Clinton)

***nsubjpass*: passive nominal subject**

A passive nominal subject is a noun phrase which is the syntactic subject of a passive clause.

"Dole was defeated by Clinton" *nsubjpass*(defeated, Dole)

***pobj*: object of a preposition**

The object of a preposition is the head of a noun phrase following the preposition, or the adverbs "here" and "there". (The preposition in turn may be modifying a noun, verb, etc.) Unlike the Penn Treebank, we here define cases of VBG quasi-prepositions like "including", "concerning", etc. as instances of pobj. In the case of preposition stranding, the object can precede the preposition (e.g., "What does CPR stand for?").

"I sat on the chair" *pobj*(on, chair)

***xcomp*: open clausal complement**

An open clausal complement (*xcomp*) of a VP or an ADJP is a clausal complement without its own subject, whose reference is determined by an external subject. These complements are always non-finite. The name *xcomp* is borrowed from Lexical-Functional Grammar.

"He says that you like to swim" *xcomp*(like, swim)

***xsubj*: controlling subject**

A controlling subject is the relation between the head of a open clausal complement (*xcomp*) and the external subject of that clause.

'Tom likes to eat fish" *xsubj*(eat, Tom)

Appendix C

Complete tables of classification results

d_{ik}	***base***	***average accuracy***
count	token	**90.6**
Boolean	token	83.7
Boolean	sentence	82.2
count	sentence	82.2

Table C.1: Average accuracies (%) of *argument from example*-against-others classification

d_{ik}	***base***	***average accuracy***
Boolean	token	**70.4**
count	token	**70.4**
Boolean	sentence	70.3
count	sentence	70.3

Table C.2: Average accuracies (%) of *argument from cause to effect*-against-others classification

d_{ik}	***base***	***average accuracy***
count	sentence	**90.8**
Boolean	sentence	90.7
Boolean	token	90.7
count	token	90.7

Table C.3: Average accuracies (%) of *practical reasoning*-against-others classification

sentiment	***scaled?***	***base***	***average accuracy***
cp_split	N	sentence	**62.9**
cp_split	N	token	62.5
cp_combined	Y	sentence	61.7
c_only	Y	sentence	60.8
c_only	N	sentence	60.6
c_only	Y	token	60.6
c_only	N	token	60.0
cp_combined	Y	token	60.0
cp_split	Y	sentence	59.9
cp_split	Y	token	59.3
cp_combined	N	sentence	58.8
cp_combined	N	token	58.6

Table C.4: Average accuracies (%) of *argument from consequences*-against-others classification

dependency	***base***	***average accuracy***
cp_combined	token	**63.2**
cp_combined	sentence	62.2
c_only	token	61.3
c_only	sentence	60.7
cp_split	sentence	57.2
cp_split	token	57.2

Table C.5: Average accuracies (%) of *argument from verbal classification*-against-others classification

d_{ik}	***base***	***average accuracy***
Boolean	sentence	**80.6**
count	sentence	80.6
Boolean	token	77.9
count	token	77.9

Table C.6: Average accuracies (%) of *argument from cause to effect* versus *argument from example* classification

d_{ik}	***base***	***average accuracy***
Boolean	token	**93.1**
count	sentence	92.8
Boolean	sentence	92.8
count	token	90.5

Table C.7: Average accuracies (%) of *practical reasoning* versus *argument from example* classification

d_{ik}	***base***	***average accuracy***
count	sentence	**94.2**
Boolean	sentence	93.9
Boolean	token	93.2
count	token	93.2

Table C.8: Average accuracies (%) of *practical reasoning* versus *argument from cause to effect* classification

sentiment	***scaled?***	d_{ik}	***base***	***average accuracy***
cp_split	Y	Boolean	sentence	**86.9**
cp_split	Y	count	sentence	86.9
cp_split	Y	Boolean	token	86.7
cp_split	Y	count	token	86.7
cp_combined	Y	Boolean	token	85.6
cp_combined	Y	count	token	85.6
cp_combined	Y	Boolean	sentence	85.0
cp_combined	Y	count	sentence	85.0
cp_split	N	Boolean	token	84.8
cp_split	N	count	token	84.8
c_only	Y	Boolean	token	84.6
c_only	Y	count	token	84.6
c_only	N	Boolean	token	84.2
c_only	N	count	token	84.2
cp_split	N	Boolean	sentence	84.2
cp_split	N	count	sentence	84.2
c_only	Y	Boolean	sentence	83.5
c_only	Y	count	sentence	83.5
cp_combined	N	Boolean	token	83.5
cp_combined	N	count	token	83.5
c_only	N	Boolean	sentence	83.1
c_only	N	count	sentence	83.1
cp_combined	N	Boolean	sentence	82.9
cp_combined	N	count	sentence	82.9

Table C.9: Average accuracies (%) of *argument from consequences* versus *argument from example* classification

sentiment	***scaled?***	d_{ik}	***base***	***average accuracy***
c_only	Y	Boolean	sentence	**87.7**
c_only	Y	count	sentence	87.7
c_only	Y	Boolean	token	86.3
c_only	Y	count	token	86.3
c_only	N	Boolean	sentence	86.0
c_only	N	count	sentence	86.0
c_only	N	Boolean	token	85.2
c_only	N	count	token	85.2
cp_combined	Y	Boolean	sentence	85.2
cp_combined	Y	Boolean	token	85.2
cp_combined	Y	count	sentence	85.2
cp_combined	Y	count	token	85.2
cp_split	N	Boolean	sentence	84.8
cp_split	N	count	sentence	84.8
cp_combined	N	Boolean	sentence	84.6
cp_combined	N	count	sentence	84.6
cp_split	N	Boolean	token	84.6
cp_split	N	count	token	84.6
cp_split	Y	Boolean	sentence	83.5
cp_split	Y	Boolean	token	83.5
cp_split	Y	count	sentence	83.5
cp_split	Y	count	token	83.5
cp_combined	N	Boolean	token	83.1
cp_combined	N	count	token	83.1

Table C.10: Average accuracies (%) of *argument from consequences* versus *argument from cause to effect* classification

sentiment	***scaled?***	d_{ik}	***base***	***average accuracy***
c_only	N	Boolean	sentence	**97.9**
c_only	N	Boolean	token	97.9
c_only	N	count	sentence	97.9
c_only	N	count	token	97.9
c_only	Y	Boolean	sentence	97.9
c_only	Y	Boolean	token	97.9
c_only	Y	count	sentence	97.9
c_only	Y	count	token	97.9
cp_combined	N	Boolean	sentence	97.9
cp_combined	N	Boolean	token	97.9
cp_combined	N	count	sentence	97.9
cp_combined	N	count	token	97.9
cp_combined	Y	Boolean	sentence	97.9
cp_combined	Y	Boolean	token	97.9
cp_combined	Y	count	sentence	97.9
cp_combined	Y	count	token	97.9
cp_split	N	Boolean	sentence	97.9
cp_split	N	Boolean	token	97.9
cp_split	N	count	sentence	97.9
cp_split	N	count	token	97.9
cp_split	Y	Boolean	sentence	97.9
cp_split	Y	Boolean	token	97.9
cp_split	Y	count	sentence	97.9
cp_split	Y	count	token	97.9

Table C.11: Average accuracies (%) of *argument from consequences* versus *practical reasoning* classification

dependency	d_{ik}	***base***	***average accuracy***
cp_combined	Boolean	token	**86.0**
cp_combined	count	token	86.0
cp_combined	Boolean	sentence	84.4
cp_combined	count	sentence	84.4
c_only	Boolean	token	82.5
c_only	count	token	82.5
c_only	Boolean	sentence	82.3
c_only	count	sentence	82.3
cp_split	Boolean	token	80.2
cp_split	count	token	80.2
cp_split	Boolean	sentence	80.0
cp_split	count	sentence	80.0

Table C.12: Average accuracies (%) of *argument from verbal classification* versus *argument from example* classification

dependency	d_{ik}	***base***	***average accuracy***
c_only	Boolean	sentence	**85.6**
c_only	count	sentence	85.6
c_only	Boolean	token	85.2
c_only	count	token	85.2
cp_split	Boolean	sentence	83.7
cp_split	count	sentence	83.7
cp_combined	Boolean	sentence	83.1
cp_combined	count	sentence	83.1
cp_split	Boolean	token	83.1
cp_split	count	token	83.1
cp_combined	Boolean	token	82.5
cp_combined	count	token	82.5

Table C.13: Average accuracies (%) of *argument from verbal classification* versus *argument from cause to effect* classification

dependency	d_{ik}	***base***	***average accuracy***
c_only	Boolean	sentence	**98.3**
c_only	Boolean	token	98.3
c_only	count	sentence	98.3
c_only	count	token	98.3
cp_combined	Boolean	sentence	98.3
cp_combined	Boolean	token	98.3
cp_combined	count	sentence	98.3
cp_combined	count	token	98.3
cp_split	Boolean	sentence	98.3
cp_split	Boolean	token	98.3
cp_split	count	sentence	98.3
cp_split	count	token	98.3

Table C.14: Average accuracies (%) of *argument from verbal classification* versus *practical reasoning* classification

Table C.15: Average accuracies (%) of *argument from verbal classification* versus *argument from consequences* classification

dependency	***sentiment***	***scaled?***	***base***	***average accuracy***
cp_split	*c_only*	N	sentence	**64.2**
c_only	*c_only*	Y	token	64.0
cp_split	*c_only*	N	token	64.0
c_only	*c_only*	N	token	63.8
cp_split	*cp_combined*	Y	token	63.1
cp_split	*c_only*	Y	sentence	62.9
cp_combined	*c_only*	N	token	62.7
cp_split	*cp_combined*	N	token	62.7
cp_split	*cp_combined*	Y	sentence	62.7
c_only	*c_only*	N	sentence	62.5
cp_combined	*c_only*	Y	token	62.5
cp_split	*cp_combined*	N	sentence	62.5
c_only	*c_only*	Y	sentence	62.3
cp_combined	*c_only*	N	sentence	61.9
cp_split	*c_only*	Y	token	61.7
cp_combined	*c_only*	Y	sentence	61.5
c_only	*cp_combined*	Y	token	60.4
cp_combined	*cp_combined*	Y	sentence	60.0
cp_split	*cp_split*	Y	sentence	59.8
c_only	*cp_combined*	N	sentence	59.6
cp_combined	*cp_combined*	Y	token	59.6
c_only	*cp_combined*	N	token	59.4
c_only	*cp_combined*	Y	sentence	59.4
cp_split	*cp_split*	N	sentence	59.4
cp_split	*cp_split*	Y	token	59.2

Continued on next page

Table C.15 – continued from previous page

dependency	*sentiment*	*scaled?*	*base*	*average accuracy*
cp_combined	*cp_combined*	N	sentence	58.8
cp_combined	*cp_combined*	N	token	58.8
cp_split	*cp_split*	N	token	58.3
cp_combined	*cp_split*	Y	token	57.1
c_only	*cp_split*	Y	token	55.6
cp_combined	*cp_split*	N	sentence	50.6
cp_combined	*cp_split*	N	token	50.6
cp_combined	*cp_split*	Y	sentence	50.2
c_only	*cp_split*	N	sentence	49.4
c_only	*cp_split*	N	token	49.4
c_only	*cp_split*	Y	sentence	49.2