Entailment Rules for Predicates:
Representation, Learning, Application
and Evaluation

Idan Szpektor

Department of Computer Science
Ph.D. Thesis

Submitted to the Senate of Bar Ilan University
Ramat Gan, Israel October 2009
This work was carried out under the supervision of Prof. Ido Dagan (Department of Computer Science), Bar-Ilan University.
To my wife Mira and my daughter Noya, with all my love.
Acknowledgements

First, I would like to thank my supervisor, Ido Dagan. I was lucky to pursue my PhD under his supervision. His passion for the field, his insights and his patient guidance make him a role model, and I can only hope that I have learned enough to eventually walk this path as well.

I would like to express my gratitude to several additional individuals who helped with advice and discussion: Oliviero Stock, for providing me the great opportunity to collaborate with FBK-irst and for showing me that interdisciplinary projects could be both fun and challenging; Jacob Goldberger, for sharing his expertise in machine learning; Alfio Massimiliano Gliozzo, for enlightening discussions; Hadar Shemtov, for hosting me at Yahoo! and for encouraging me to think out-of-the-box; and Roy Bar Haim, for his friendship, his good council and for his help along these years.

I would like to thank my colleagues at FBK-irst. It was a great experience to collaborate with them. I especially want to thank Bernardo Magnini, Milen Kouylekov, Lorenza Romano, Alberto Lavelli and Danilo Giampiccolo.

I wish also to thank my colleagues and friends at Bar Ilan. It was a special period, and one of its main reasons are the wonderful people at Bar-Ilan. Especially, I would like to thank Oren Glickman, Shachar Mirkin, Eyal Shmarch, Lili Kotlerman and Jonathan Berant for their collaboration, their ideas and help whenever needed and mostly for their friendship. I would also like to thank Ephi Sachs, Chen Erez, Ruthie Mandel, Malky Rabinowitz, Efrat Brown and Roni Ben Aharon, who helped
my research with various data analysis tasks.

I want to thank my parents, my brothers and the extended family members for their support and interest in my research. Finally, I want to express my love to my daughter Noya, whose smiles light my life, and to my wife Mira, whose belief in me, her endless love, support and encouragement made this work what it is.
Contents

1 Introduction 1
   1.1 The Role of Knowledge in Applied Inference 1
   1.2 Main Contributions 5
   1.3 Thesis Outline 7

2 Background 9
   2.1 Definitions and Terminology 9
   2.2 Rule Representation 12
   2.3 Rule Generation 15
      2.3.1 Template Structure Extraction 15
      2.3.2 Rule Learning 17
   2.4 Validity of Rule Application in Context 24
   2.5 Rule Evaluation 26

3 Contribution of Entailment Rules to Inference 28
   3.1 Introduction 28
   3.2 Definition of Entailment Levels 29
      3.2.1 The Lexical Entailment Level 29
      3.2.2 The Lexical-syntactic Entailment Level 31
   3.3 Empirical Analysis 33
3.3.1 Data and Annotation Procedure ........................................... 33
3.3.2 Evaluating the Different Levels of Entailment ....................... 34
3.3.3 The Contribution of Various Inference Mechanisms ................. 36
3.4 Related Work ........................................................................ 38
3.5 Conclusions ......................................................................... 39

4 Evaluation of Entailment Rules .................................................. 40
  4.1 Introduction ......................................................................... 40
  4.2 Instance-based Evaluation Methodology ...................................... 42
    4.2.1 Example Collection .......................................................... 43
    4.2.2 Example Judgment Questions ............................................. 45
    4.2.3 Evaluation Process ............................................................ 46
    4.2.4 Rule Precision .................................................................. 47
  4.3 Instance-based Evaluation Experiment ...................................... 48
    4.3.1 Experimental Settings ....................................................... 48
    4.3.2 Methodology Evaluation Results ....................................... 50
    4.3.3 DIRT and TEASE Evaluation Results ............................... 51
  4.4 Recap: Human-based vs. Application-based Evaluations .............. 53
  4.5 Application-based Evaluation Methodology .............................. 55
  4.6 Conclusions ......................................................................... 59

5 Canonization of Entailment Rules ................................................. 61
  5.1 Introduction ......................................................................... 61
  5.2 A Modular Approach for Rule Learning ..................................... 62
    5.2.1 Morpho-Syntactic Canonization Module ............................ 63
    5.2.2 Applying the Canonization Module ................................... 65
  5.3 Evaluation ........................................................................... 66
    5.3.1 Human Judgement Evaluation ........................................... 66
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3.3 Nominalization Relations</td>
<td>103</td>
</tr>
<tr>
<td>7.3.4 Verb-Verb Relations</td>
<td>104</td>
</tr>
<tr>
<td>7.3.5 Adjective-Verb Relations</td>
<td>106</td>
</tr>
<tr>
<td>7.3.6 Extending WordNet with Missing Information</td>
<td>107</td>
</tr>
<tr>
<td>7.3.7 Manual Analysis of AmWN Population Quality</td>
<td>109</td>
</tr>
<tr>
<td>7.4 Rule Filtering</td>
<td>113</td>
</tr>
<tr>
<td>7.4.1 Sense Drifting</td>
<td>113</td>
</tr>
<tr>
<td>7.4.2 Rare Senses</td>
<td>114</td>
</tr>
<tr>
<td>7.5 Application-oriented Evaluations</td>
<td>116</td>
</tr>
<tr>
<td>7.5.1 Text Annotation with Functional Roles and Subcategorization frames</td>
<td>117</td>
</tr>
<tr>
<td>7.5.2 The ACE Evaluation</td>
<td>120</td>
</tr>
<tr>
<td>7.5.3 The RTE Evaluation</td>
<td>125</td>
</tr>
<tr>
<td>7.5.4 Additional Analyses</td>
<td>128</td>
</tr>
<tr>
<td>7.6 Related Work</td>
<td>131</td>
</tr>
<tr>
<td>7.6.1 WordNet Extensions</td>
<td>131</td>
</tr>
<tr>
<td>7.6.2 Frame Semantics</td>
<td>132</td>
</tr>
<tr>
<td>7.6.3 Other Related Works</td>
<td>133</td>
</tr>
<tr>
<td>7.7 Conclusions</td>
<td>134</td>
</tr>
<tr>
<td>8 Contextual Preferences</td>
<td>136</td>
</tr>
<tr>
<td>8.1 Introduction</td>
<td>136</td>
</tr>
<tr>
<td>8.2 The Contextual Preferences Framework</td>
<td>138</td>
</tr>
<tr>
<td>8.3 Contextual Preferences Models</td>
<td>139</td>
</tr>
<tr>
<td>8.3.1 Contextual Preferences for Global Context</td>
<td>140</td>
</tr>
<tr>
<td>8.3.2 Contextual Preferences for Variables</td>
<td>141</td>
</tr>
<tr>
<td>8.3.3 Overall Score for a Match</td>
<td>143</td>
</tr>
</tbody>
</table>
List of Tables

2.1 Examples of predicative entailment rules. The first two rules do not contain arguments. The last two rules demonstrate templates whose predicates are nominalizations. .......................................................... 10

2.2 Examples for features for the anchor-set learning approach and single-feature learning approach for two related templates. .......................... 21

3.1 Examples of text-hypothesis pairs, taken from the RTE-1 test-set. Each line includes the example number at the RTE test-set, the text and hypothesis, the task within the test-set, whether entailment holds between the text and hypothesis (Ent.), whether Lexical entailment holds (Lex. Ent.) and whether Lexical-Syntactic entailment holds (Syn. Ent.). ................................................................. 30

3.2 Results per level of entailment. L stands for the lexical level and LS stands for the lexical syntactic level. ................................................. 34

3.3 Correlation between the entailment levels. (a) includes only the positive examples from the RTE dataset sample, and (b) includes only the negative examples. ................................................................. 36

3.4 The frequency (f), contribution to recall (∆R) and percentage (%), within the gold standard positive examples, of the various inference mechanisms at each level, ordered by their significance. .................. 37
4.1 Examples of templates suggested by DIRT and TEASE as having an entailment relation, in some direction, with the input template ‘X change Y’. The entailment direction arrows were judged manually and added for readability. ......................................................... 42

4.2 Rule evaluation examples and their judgment. .......................... 44

4.3 Examples for disagreement between the two judges. .................... 50

4.4 Average Precision (P) and Yield (Y) at the rule and template levels... 51

5.1 Examples of learned rules that differ only in their morpho-syntactic structure. .......................................................... 62

5.2 Some of the syntactic rules used in our implementation, together with usage examples for transforming complex template into the simpler template ‘X subj find obj Y’ (the applications of the second rule and the third rule are also demonstrated in Figure 5.1). .................. 64

5.3 Average Precision and Yield of the output lists. ....................... 68

5.4 Examples for correct templates that TEASE learned only after using canonization rules. .................................................... 68

5.5 Results for the protein interaction setup using TEASE with and without canonization. ....................................................... 71

6.1 Rule type distribution of a sample of 200 rules that extracted incorrect mentions. The corresponding numbers of incorrect mentions extracted by the sampled rules is shown in parentheses. ....................... 84

6.2 Distribution of reasons for false positives (incorrect argument extrac-
tions) by BInc at $K=20$. ...................................................... 85

6.3 Distribution of reasons for false negatives (missed argument mentions) by BInc at $K=20$. ...................................................... 85
7.1 Examples for inference chains obtained using AmWN. Bolded arrows designate inference steps that could be performed correctly only with the additional information within AmWN. Arguments are subscripted with functional roles, e.g. subject (subj) and indirect-object (ind-obj). For brevity, predicate subcategorization frames are omitted. 92

7.2 Some argument mappings for WordNet verb-verb relations discovered by unary-DIRT. 104

7.3 A summary of the relations in WordNet that were identified as representing entailment, and the percentage of these relations for which our population methodologies found at least one mapping edge. 110

7.4 The evaluated accuracy of a random sample of mapping edges created in our current implementation for the entailment information presented in WordNet. Both overall and per type accuracy is presented, together with the number of samples evaluated. 110

7.5 The manually evaluated accuracy of the random samples of the new adjective nodes and copula-construction nodes missing from WordNet created in our current implementation. 112

7.6 Recall (R), Precision (P) and F1 results for the various tested configurations. 121

7.7 Distribution of reasons for false positives (incorrect argument extractions). 123

7.8 Distribution of reasons for false negatives (missed argument mentions). 123

7.9 The Recall (R), Precision (P) and F1 results for ablation tests. 124

7.10 Accuracy(%) results for the different RTE Challenge datasets for the configurations where the inference system uses AmWN as a resource for rules and where it does not. 127

xi
7.11 Examples of AmWN rules missing from other resources, which were applied during inference to the RTE-2 test set. For readability, binary rules are presented when two complementary unary rules were generated (when mapping was needed for two arguments of the same predicate in the hypothesis).

7.12 Recall (R), Precision (P) and F1 results for the ACE dataset for AmWN and AmWN-nf.

7.13 Recall (R), Precision (P) and F1 results for the WordNet-related resources vs. the unary-DIRT resource, tested on the ACE dataset.

7.14 FrameNet’s upper bound for covering WordNet’s entailment relations.

8.1 Recall (R), Precision (P) and Mean Average Precision (MAP) when only matching template hypotheses directly.

8.2 Recall (R), Precision (P) and Mean Average Precision (MAP) when also using rules for matching.

8.3 MAP(%), under the ‘50 rules, All’ setup, when adding component match scores to Precision (P) or prior-only MAP baselines, and when ranking with allCP or allCP+pr methods but ignoring that component scores.
List of Figures

1.1 An example of a positive (entailing) text-hypothesis pair and a negative (non-entailing) pair, taken from the RTE-2 development dataset (Bar-Haim et al., 2006). ................................................................. 2

2.1 Part of the employment Nomlex entry, describing the possible nominal syntactic dependency positions for each functional role of the transitive subcategorization frame of employ. ............................................ 19

5.1 Chaining of canonization rules that transforms the path template between the arguments \( \{ X=Google; Y=Sprinks \} \), which occurs in the sentence “We witnessed the acquisition of Kaltix and Sprinks by another growing company, Google”, into a canonized template form. The first rule applied is a nominalization rule, followed by removal of apposition and removal of conjunction (as described in Table 5.2). As can be seen, applying the rules in any order will result in the same final canonized form. ................................................................. 65

6.1 Average Precision, Recall and F1 at different top K rule cutoff points. ......................... 82
7.1 AmWN node examples. The upper three nodes represent a single term. The lower two nodes represent a parse sub-tree for a predicate expression. Two options for representing kick the bucket are shown, either as a compound verb or as a sub-tree. SID is the attached WordNet synset ID and Frame is the attached subcategorization frame of each node.

7.2 A description of ‘arrest of X ⇒ take X into custody’ as a mapping edge in AmWN. Each template points to the specific node in the predicate parse structure to which the template attaches.

7.3 A description of ‘arrest ⇒ take into custody’ as a substitution edge in AmWN.

7.4 A WordNet sense-drifting traversal, generating the incorrect inference ‘kidnap ⇒ arrest’.

7.5 Algorithm outline for efficiently evaluating a path as valid or not.

8.1 The directional matching relationships between a hypothesis (h), an entailment rule (r) and a text (t) in the Contextual Preferences framework.

8.2 Recall-Precision curves for ranking using: (a) only the prior (baseline); (b) allCP; (c) allCP+pr.
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACE</td>
<td>Automatic Content Extraction</td>
</tr>
<tr>
<td>AmWN</td>
<td>Argument-mapped WordNet</td>
</tr>
<tr>
<td>AP</td>
<td>Average Precision</td>
</tr>
<tr>
<td>BInc</td>
<td>Balanced Inclusion</td>
</tr>
<tr>
<td>CD</td>
<td>Comparable Documents</td>
</tr>
<tr>
<td>CP</td>
<td>Contextual Preferences</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>IE</td>
<td>Information Extraction</td>
</tr>
<tr>
<td>IR</td>
<td>Information Retrieval</td>
</tr>
<tr>
<td>ISP</td>
<td>Inferential Selectional Preferences</td>
</tr>
<tr>
<td>LHS</td>
<td>left hand side</td>
</tr>
<tr>
<td>LSA</td>
<td>Latent Semantic Analysis</td>
</tr>
<tr>
<td>MAP</td>
<td>Mean Average Precision</td>
</tr>
<tr>
<td>MDS</td>
<td>Multi Document Summarization</td>
</tr>
<tr>
<td>MT</td>
<td>Machine Translation</td>
</tr>
<tr>
<td>NER</td>
<td>Named-Entity Recognizer</td>
</tr>
<tr>
<td>NLP</td>
<td>Natural Language Processing</td>
</tr>
<tr>
<td>PMI</td>
<td>point-wise mutual information</td>
</tr>
<tr>
<td>QA</td>
<td>Question Answering</td>
</tr>
<tr>
<td>RE</td>
<td>Relation Extraction</td>
</tr>
<tr>
<td>RHS</td>
<td>right hand side</td>
</tr>
<tr>
<td>RTE</td>
<td>Recognising Textual Entailment</td>
</tr>
<tr>
<td>TE</td>
<td>Textual Entailment</td>
</tr>
<tr>
<td>WN</td>
<td>WordNet</td>
</tr>
</tbody>
</table>
Abstract

A fundamental phenomenon in natural languages is the ability to infer a specific meaning from different textual expressions. Correspondingly, within many NLP applications it is vital to identify the various ways in which a target meaning might be conveyed. For example, NLP systems should be able to infer that “Tom Cruise is married” from the text “Katie Holmes wants to divorce Tom Cruise”. To perform such inference, knowledge about linguistic variability, as well as world-knowledge, is required. To successfully perform inference in the above example, both the knowledge that divorce implies marriage (‘divorce X ⇒ marry X’) and that the person you married is also considered married (‘marry X ⇒ X marry’) should be known. In this thesis we focus on knowledge required for semantic inference between predicates, such as in the above example. We follow the generic Textual Entailment paradigm for textual semantic inference, which models semantic inference as a text-to-text relation. Within this paradigm, a knowledge type called entailment rules, describing inference between two templates (text fragments with variables) seems appropriate to describe inference between predicates, as the above two entailment rules exemplify. Our main goal in this thesis is to construct large-scale knowledge bases of entailment rules that will be useful for semantic inference within various NLP applications. With this target in mind, we investigate different aspects of entailment rules between predicates, including the representation of rules, their acquisition, application and evaluation. This thesis proposes novel representations of entailment rules, including
canonical forms of rules, rules between unary templates and augmentation of rules with functional role and subcategorization frame information; it introduces directional distributional similarity measures as more adequate measures for learning entailment rules from corpora; it presents a general framework for augmenting WordNet with argument mapping for its non-substitutable relations between predicates; it presents a novel generic framework for context-sensitive rule applications; and finally, this thesis introduces novel approaches for both manual evaluation and application-based evaluation of entailment rule-bases.
Mr. Praline: 'E's not pinin'! 'E's passed on! This parrot is no more! He has ceased to be! 'E's expired and gone to meet 'is maker! 'E's a stiff! Bereft of life, 'e rests in peace! If you hadn’t nailed 'im to the perch 'e’d be pushing up the daisies! 'Is metabolic processes are now 'istory! 'E’s off the twig! 'E’s kicked the bucket, 'e’s shuffled off 'is mortal coil, run down the curtain and joined the bleedin’ choir invisibile!! THIS IS AN EX-PARROT!!

— from Monty Python’s Dead Parrot sketch

Chapter 1

Introduction

1.1 The Role of Knowledge in Applied Inference

In many Natural Language Processing applications, such as Question Answering (QA), Information Retrieval (IR), Information Extraction (IE) and Multi Document Summarization (MDS), it is crucial to recognize whether a specific target meaning can be inferred from a given text. For example, a QA system has to deduce that “SCO sued IBM” can be inferred from “SCO won a lawsuit against IBM” to answer “Whom did SCO sue?”

In an IE example, to correctly recognize that the text “Mozart wrote the Jupiter symphony” is a mention of a Composition, “Mozart composed the Jupiter symphony” need to be inferred from the text.
CHAPTER 1. INTRODUCTION

Positive example:
T: The drugs that slow down or halt Alzheimer’s disease work best the earlier you administer them.

H: Alzheimer’s disease is treated using drugs.

Negative example:
T: Arabic, for example, is used densely across North Africa and from the Eastern Mediterranean to the Philippines, as the key language of the Arab world and the primary vehicle of Islam.

H: Arabic is the primary language of the Philippines.

Figure 1.1: An example of a positive (entailing) text-hypothesis pair and a negative (non-entailing) pair, taken from the RTE-2 development dataset (Bar-Haim et al., 2006).

Such applied semantic inference was addressed mainly at the individual application level. That is, research targeting this core problem was performed within specific application frameworks, ignoring the similar reasoning required across various applications. Recently, this type of reasoning has been identified as a generic core semantic inference paradigm by the Textual Entailment (TE) framework (Dagan, Glickman, and Magnini, 2006). The heart of the TE framework is a directional relation between two texts, termed text and hypothesis. The textual entailment relation between a text $t$ and a hypothesis $h$, denoted $t \rightarrow h$, holds if a human reading the text would say that the hypothesis is most likely true. An example for an entailing and a non-entailing text-hypothesis pairs is presented in Figure 1.1.

TE is a text to text mapping paradigm, aiming at “what” is needed to be solved for applied semantic inference. It leaves the “how” as an open question, allowing for different text representations and inference approaches to compete, or complement each other, on correctly identifying textual entailment relations between texts. This was further emphasized by the series of TE benchmarks conducted in the last
several years, the PASCAL Recognizing Textual Entailment (RTE) challenges (Dagan, Glickman, and Magnini, 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Giampiccolo et al., 2008): a large number of TE systems competed in these tasks, showing a variety of approaches from complex logic representation and inference to simple bag of words and word coverage models.

In this thesis, we adopt Textual Entailment as our underlying framework for modeling applied semantic inference. A more detailed inspection of the TE definition reveals an important aspect of the framework: the TE relation follows common human inference decisions, and when humans perform semantic inference they use the linguistic and world knowledge they hold to correctly assess entailment relations. It follows that the TE framework assumes and expects the utilization of linguistic and world knowledge, as possessed by an average human being, as part of the entailment validation process. This is an important aspect of Textual Entailment, as NLP applications indeed regularly need such knowledge as part of answering information requests. Yet, such knowledge is still not widely accessible in the form of available resources. Indeed, as stressed by the participants of the PASCAL RTE challenges (Bar-Haim et al., 2006; Giampiccolo et al., 2007), a major obstacle for further progress in Textual Entailment is the lack of broad-coverage knowledge-bases for semantic variability patterns.

Our overall goal is to help lifting the “lack of knowledge” barrier. Out of the many types of knowledge that are required for applied inference, we target one prominent knowledge type: entailment rules. An entailment rule is a directional relation between two templates, text patterns with variables (see Section 2.1 for a detailed definition). In this thesis, we focus on entailment rules between predicates, where each template represents a predicate and some of its arguments. Examples for such rules are ‘\(X\) win lawsuit against \(Y\) \(\Rightarrow\) \(X\) sue \(Y\)’ and ‘\(X\)’s invasion into \(Y\) \(\Rightarrow\) \(X\) attack \(Y\)’. Entailment rules are rudimentary building blocks in semantic inference. For example, given the rule ‘\(X\) win lawsuit against \(Y\) \(\Rightarrow\) \(X\) sue \(Y\)’, a QA system can identify \(IBM\)
as the answer for “Whom did SCO sue?” in the previous QA example.

Two complementary issues should be taken into consideration when constructing knowledge-bases in general, and specifically for entailment rules. First, for a generic knowledge resource to be useful it should be large-scale, that is, it should cover many of the variability patterns captured within the specific knowledge type. This is not a trivial task and most algorithms for learning entailment rules do not produce such broad-coverage rule-bases. For example, the DIRT learning algorithm (Lin and Pantel, 2001) is almost the only algorithm utilized for the RTE task in practice due to its large-scale output (Bar-Haim et al., 2006; Giampiccolo et al., 2007). In addition to broad coverage, a knowledge resource should also be accurate. A rule-base accuracy is first of all tested by the correctness of its rules. A resource containing many incorrect rules would make it unusable in practice due to the many incorrect inferences it would generate. Generating high quality rule-bases is not a simple problem. Indeed, the quality of current large-scale resources is typically mediocre, as in the case of the DIRT algorithm (de Marneffe, Pado, and Manning, 2009).

Yet, even correct rules should be applied only under specific contexts. The reason is that many predicates have different meanings under different contexts, which corresponds to the well known phenomenon of semantic ambiguity in a language. Thus, applying a rule within inappropriate contexts may result in incorrect inferences. For example, ‘$X$ acquire $Y \Rightarrow X$ buy $Y$’ is a correct rule, but it should not be applied to “Children acquire new languages fast” to prevent making the incorrect inference “children buy new languages”. Thus, when crafting an entailment rule resource, modeling the valid contexts for each rule should be taken into consideration as well.

In this thesis we address the various aspects that are involved in constructing a knowledge base of entailment rules between predicates. Our target is to construct
large-scale rule-bases that are sufficiently accurate to be usable within NLP applications. This thesis introduces advances in the areas of entailment rule representation, learning, application and evaluation, which make significant contributions towards meeting our goal.

1.2 Main Contributions

Following are the main contributions of this thesis:

Entailment Rule Representation

We introduce canonical forms of entailment rules, which compactly represent morphosyntactic rule variations. Using canonical rule forms not only improves inference coverage but also improves rule learning, as more statistics can be gathered to learn a specific rule.

We also propose unary entailment rules, entailment rules between templates with only one variable, as a more appropriate representation of the variability between predicates, compared to the previously widely used binary entailment rules (rules between templates with two variables). We discuss the theoretical benefits of using unary rules instead of binary rules, and show their empirical performance improvement.

Finally, we propose extending the typical syntactic structure of entailment rules, which is based on dependency relation information. We propose to augment each argument in a template with its syntactic functional role description and each predicate in the template with the specification of its subactegorization frame. We present the theoretical benefits of this extension and empirically compare it to the typical representation of rules.
Rule-base Learning

We present two novel approaches for learning rules in an unsupervised way. The first approach is based on directional distributional similarity measures between templates, estimated from large corpora. We introduce a novel directional similarity measure and show that the rule-base learned with that measure substantially outperforms rule-bases learned by symmetric measures.

The second approach takes the WordNet lexicon (Miller, 1995) as a starting point. WordNet’s relations between words are only at the lexical level, ignoring syntactic changes that may be needed for the inference between some predicates to be correct. We introduce a novel framework under which the WordNet lexicon is augmented with argument mapping between predicates, allowing to generate entailment rules between templates with variables. We present a specific implementation of that framework and show that it improves WordNet-based inference compared to standard utilization of WordNet.

The rule-base outputs of both our distributional-similarity learning approach and our augmented WordNet approach will be publicly available soon.

Context-sensitive Rule Application

We present a novel framework for modeling context-sensitive rule application, termed Contextual Preferences. This framework generalizes and extends prior work. Under this framework each textual object is augmented with its contextual preferences, whose purpose is to constrain and disambiguate the object’s meaning. We introduce a specific implementation of this framework and show that the quality of rule application is substantially improved when rules are applied within the Contextual Preferences framework.
1.3. THESIS OUTLINE

Evaluation

We discuss the limitations of the prior common evaluation approaches for entailment rules, in which human judges assess each rule in a rule-base as correct or not. We propose two alternative approaches for evaluating entailment rule resources. The first is a manual approach in which human judges assess not a rule but a sample of its specific applications. We show that this approach is better defined, resulting in higher agreement between annotators. This approach also allows for a more detailed evaluation of the behavior of rules.

Manual evaluation schemes in general have limitations. Thus, as our second evaluation approach we propose an application-based evaluation scheme. Previous application-based evaluations require a full, typically complex system to be tested, which makes it hard to measure rule-base performance in isolation from other system components. Our proposed scheme uses a fairly simple framework, which utilizes an available information extraction dataset. This simple framework allows for relatively easy comparison between the performance of different rule-bases.

1.3 Thesis Outline

The rest of the thesis is organized as follows: Chapter 2 first describes the definition of entailment rules together with the terminology used in the rest of the thesis. It then provides the necessary background and related work on the different aspects of entailment rules.

Chapter 3 presents an analysis of the Textual Entailment task. The performance of two levels for modeling Textual Entailment (lexical vs. lexical-syntactic) is manually evaluated on the RTE-1 benchmark. The evaluation shows that lexical-syntactic modeling is better than lexical modeling for the TE task and that entailment rules are the knowledge type that contributes the most to successful inference.
Chapter 4 presents two novel schemes for entailment rule evaluation. The first scheme is based on manual human evaluation. The second scheme is an application-based evaluation approach. We discuss the benefits of our two schemes over prior state-of-the-art evaluation approaches.

Chapter 5 presents a new compact representation of entailment rules based on morpho-syntactic canonical forms of templates. We discuss the benefits of this representation both at learning time and at inference time and show empirical evidence that supports them.

Chapter 6 introduces unary entailment rules, discussing their benefits over the common binary rules. Our experiments show that unary rules’ performance outperforms that of binary rules. In addition, we discuss the benefits of directional distributional similarity measures for learning entailment rules. We present a novel directional measure that substantially outperforms state-of-the-art symmetric measures.

Chapter 7 presents a framework for augmenting WordNet with argument mappings for its relations between predicates. The framework is based on an extension of the typical entailment rule representation with functional role and subcategorization frame information. A concrete unsupervised implementation of this framework is described and empirical results show substantial improvement over standard WordNet-based inference.

Chapter 8 introduces a framework for context-sensitive rule application. Our novel framework generalizes and extends prior work. An unsupervised implementation for this framework is presented and empirical tests show substantial improvement of inference quality over prior work and strong baselines.

Finally, Chapter 9 concludes the thesis and suggests directions for future work.
Chapter 2

Background

This chapter presents the background material necessary to understand the contributions of this thesis in the area of acquisition and application of entailment rules between predicates. First, some definitions and terminology of entailment rules are presented, with emphasis on rules between predicates. Then, background is presented for each of the various aspects of entailment rules addressed in this thesis. We note that short additional backgrounds are also added in some of the later chapters, addressing their specific aspects.

2.1 Definitions and Terminology

Textual Entailment engines require knowledge resources for identifying entailment relationships between texts. One prominent type of knowledge representation needed for such inference is entailment rules (see analysis in Chapter 3). An entailment rule is a directional relation, ‘entailing-template ⇒ entailed-template’, between an entailing template (a.k.a left-hand-side or LHS) and an entailed template (a.k.a right-hand-side or RHS). Templates are patterns representing textual expressions, possibly with variables, which represent argument slots. A rule example is
Table 2.1: Examples of predicative entailment rules. The first two rules do not contain arguments. The last two rules demonstrate templates whose predicates are nominalizations.

`X prevent Y ⇒ X lower the risk of Y`.

In this thesis, we focus on templates that correspond to semantic predicates. We denote rules between such templates as **predicative entailment rules**. Table 2.1 presents some examples for such rules. We call a template with two argument variables a **binary template**, e.g. `X acquire Y`. A template with only one argument is called a **unary template**, e.g. `X acquire` and `acquire X`. Rules between binary templates are denoted **binary rules** and rules between unary templates are denoted **unary rules**.

A rule can be applied to a text `t` if the LHS is matched in `t`. During the matching process the variables in the matched template are mapped to the corresponding argument instantiations of the template mention in the text. For example, `X prevent Y` is matched in “Aspirin prevents heart attacks”, where the `X` and `Y` variables are mapped to Aspirin and heart attacks respectively. When a matched rule is applied to a text, a new consequent is inferred by generating a mention of the RHS. The mention is generated by replacing the variables in RHS with their mapped LHS argument instantiations in the text. For example, `X prevent Y ⇒ X lower the risk of Y` may be applied to “Aspirin prevents heart attacks” to infer “Aspirin lowers the
The goal of entailment rules is to help applications infer one text expression from another. Entailment rules capture fundamental linguistic and world-knowledge inferences and are used as important building blocks for more complex inference within NLP applications. For example, using ‘$X$ prevent $Y \Rightarrow X$ lower the risk of $Y$’, a QA system can deduce *Aspirin* as the answer for “What lowers the risk of heart attacks?” from the above text. In another example, an IE system can identify the prisoner argument in an *Arrest* event from the text “Elendu was taken into custody on Saturday” using ‘$\text{take } X \text{ into custody } \Rightarrow \text{arrest } X$’.

Within the TE setting, a rule $r$: ‘$LHS \Rightarrow RHS$’ may help proving that a hypothesis $h$ can be inferred from a given text $t$. This is typically done by matching the $LHS$ template in $t$ and the $RHS$ template in $h$. In this thesis, we allow $h$ to be either a text or a template, termed template hypothesis. We say that a text entails a template hypothesis if it entails an instantiation of the template, where argument instantiations are taken from the text. For example, “The stock rose 8%” entails the template hypothesis ‘$X$ gain $Y$’ since it entails its instantiation “The stock gained 8%”, using the rule ‘$X$ rise $Y \Rightarrow X$ gain $Y$’.

We note that the common notion of paraphrase rules (Barzilay and McKeown, 2001; Shinyama et al., 2002) can be viewed as a special case of entailment rules: a paraphrase between two templates, ‘$LHS \Leftrightarrow RHS$’, holds if both templates entail each other (that is, a paraphrase is a bidirectional entailment rule). An example for a paraphrase rule is ‘$X$ buy $Y \Leftrightarrow X$ purchase $Y$’. Following the textual entailment formulation, we observe that many applied inference settings only require that entailment should hold in one direction, ignoring the other direction. Thus, the requirement for symmetric paraphrasing is usually unnecessary and also too limiting. For example, in order to answer the question “Who owns Overture?” from “Yahoo acquired Overture” the directional entailment rule ‘$X$ acquire $Y \Rightarrow X$ own $Y$’ need to
be applied. This rule is clearly not bidirectional. From here on, we do not distinguish between directional entailment rules and paraphrases, and unless explicitly specified, we view both cases as entailment rules.

## 2.2 Rule Representation

One aspect of entailment rules is the representation of the rule templates. While several works learn **lexical rules** between terms or expressions without arguments (Barzilay and McKeown, 2001; Duclaye, cois Yvon, and Collin, 2003; Glickman and Dagan, 2003; Quirk, Brockett, and Dolan, 2004; Bannard and Callison-Burch, 2005), e.g. ‘buy ⇔ purchase’ and ‘run ⇒ move’, most works that learn entailment rules between predicates represent templates as either linear patterns or parse sub-trees and allow argument mapping between the two related templates. In this thesis we focus on the latter approach, being the more general case.

**Linear surface templates** contain a sequence of terms and term variables that should occur sequentially in a text (Ravichandran and Hovy, 2002; Barzilay and Lee, 2003; Pang, Knight, and Marcu, 2003; Sekine, 2005; Pantel and Pennacchiotti, 2006; Bhagat and Ravichandran, 2008), e.g. ‘X acquire Y’ and ‘X call Y indictable’. This representation is typically chosen due to efficiency issues: linear templates are simpler and faster to process as no syntactic parsing is required.

The main limitation of linear surface templates lies in the difficulty to capture well long distance relationships and sometimes even short distance relationships between words. Thus, it is hard to generalize well and learn the correct template form from several template occurrences. Indeed, most work on linear templates focused only on templates with two arguments whose position is at the beginning and at the end of the template text, e.g. ‘X acquire Y’ and ‘X’s acquisition of Y’ (as opposed to ‘X call Y indictable’). As generalization is hard, templates that are too specific are often
learned, such as ‘X completed the $1 billion stock acquisition of Y’. Similar to the problems in rule learning, it is also difficult to identify linear template occurrences when the occurrences contain additional information that is interleaved within the template words, e.g. detecting ‘X pay for Y’ in “People paid him 400 dollars for Sigur Ros tickets”.

The common approach to address the above limitations is to represent templates at the syntactic level as parse sub-trees with arguments, denoted lexical-syntactic templates (Lin and Pantel, 2001; Shinyama et al., 2002; Ibrahim, Katz, and Lin, 2003; Szpektor et al., 2004; Pekar, 2006; Zanzotto, Pennacchiotti, and Pazienza, 2006; Novischi and Moldovan, 2006; Zhao et al., 2008; Dinu and Wang, 2009). Parse trees capture better the syntactic relationships between words, allowing for better generalization when learning rules and for better matching of templates within texts when applying the rules. Typically, dependency parse representation (Hays, 1964) is utilized to describe the relationships between arguments and the predicate (e.g. subject and object relations), for example ‘X \(\leftrightarrow_{\text{subj}}\) acquire \(\rightarrow_{\text{obj}}\) Y’, ‘X \(\leftrightarrow_{\text{subj}}\) sleep’ and ‘X \(\leftrightarrow_{\text{subj}}\) pay \(\rightarrow_{\text{prep-for}}\) Y’. We note that Zhao et al. (2008) present a hybrid approach: at learning time sentences are represented at the syntactic level utilizing parse trees; but candidate templates that are extracted from these parse trees are transformed into their linear representation. The main reason for this choice is the ease of use of linear representation when aligning templates based on parallel corpora, which is the approach taken in (Zhao et al., 2008) for learning rules.

Preferring a more accurate template representation over ease of use, in this thesis we follow the lexical-syntactic representation of templates as parse sub-trees. We take the common approach and use the dependency parse representation for templates. Specifically, in this thesis we utilize the Minipar dependency parser (Lin, 1998b) and follow its dependency parse-tree output form within templates and parsed texts. From here on, unless explicitly stated otherwise, texts are considered dependency-parsed
(with Minipar) and templates are dependency parse sub-trees with arguments. For brevity of presentation, we shall often omit the dependency relation annotation from template examples.

**Syntax-oriented Terminology**

Following our choice of representation at the syntactic level, we utilize some syntax-oriented terminology that will be used later on. Our terminology is adopted from (Macleod et al., 1998) and follows concepts within theorems like Lexical Functional Grammar (Bresnan, 2001).

A **syntactic realization** of an argument is a possible syntactic position for the argument in a sentence with respect to the predicate. Examples for syntactic positions include possessive-determiner, preposition complement, noun-noun modifier and syntactic object and subject. In the dependency parse tree structure, a predicate argument realization is expressed by the dependency relation that connects the argument to the predicate. For example, in ‘\( Y \xleftarrow{\text{obj}} \text{acquire} \xrightarrow{\text{prep-for}} Z \) (‘acquire \( Y \) for \( Z \)’), \( Y \)’s realization is via the \( \text{obj} \) relation and \( Z \)’s realization is via the complement of the preposition ‘for’. In another example, ‘\( X \xleftarrow{\text{gen}} \text{acquisition} \xrightarrow{\text{nn}} Y \) (‘\( X \)’s \( Y \) acquisition’), \( X \)’s realization is via the \( \text{genitive} \) relation and \( Y \)’s realization is via the \( \text{noun-noun-modifier} \) relation.

**Syntactic functional roles** of arguments (subject, object, indirect object etc.) describe the grammatical functions that an argument can take with respect to the predicate. For example, the functional role of \( X \) in ‘\( X \)’s \( Y \) acquisition of \( Y \)’ is the \( \text{subject} \) role and the functional role of \( Y \) is the \( \text{object} \) role. There may be several possible argument realizations (syntactic positions) for the same functional role with respect to the predicate. For example, IBM takes the \( \text{subject} \) role both in “IBM’s acquisition of Cognos”, where its syntactic position is the possessive-determiner, and
in “Cognos’s acquisition by IBM”, where its syntactic position is a preposition complement. In another example, the indirect-object functional role of a verbal argument may be realized as an object position, e.g. “I gave him the book”, or as a preposition complement, e.g. “I gave the book to him”.

A subcategorization frame (frame) of a predicate is a set of functional roles that a predicate may occur with in a sentence. Some example frames are the transitive frame, containing subject and object functional roles \{subject, object\} and the intransitive frame, containing only the subject role \{subject\}. A predicate may have more than one possible related frame. For example, the predicate break can occur either with the transitive frame, e.g. “John\textsubscript{subject} broke the window\textsubscript{object}”, or with the intransitive frame, e.g. “The window\textsubscript{subject} broke”. As another example, the predicate demonstrate can also occur either with the transitive frame, e.g. “The engineers\textsubscript{subject} demonstrated the system\textsubscript{object}”, or with the intransitive frame, e.g. “The workers\textsubscript{subject} demonstrated outside the factory”.

2.3 Rule Generation

The main task regarding entailment rules is how to extract or learn them from various resources. Following a specific rule representation, two issues should be addressed: (a) how to discover the template structure within a resource; (b) how to identify entailment relations between discovered templates. We next present state-of-the-art approaches for these two problems.

2.3.1 Template Structure Extraction

There are two main approaches in the literature for extracting template structures from a given resource, such as a corpus. The first approach is to allow any possible
structure, and then select the appropriate templates according to some test or statistics. Under this scheme, any parse sub-tree may be a candidate lexical-syntactic template. Typically, candidates are extracted from each sentence in the text. For linear templates any sequence of words may be a candidate template. A typical approach to choosing templates out of all candidates is to rank all candidate templates according to their relevancy or to rank candidate rules according to their entailment relation confidence. For example, Shinyama et al. (2002) measure how rule templates are relevant to each other by counting the number of instantiations they share in a corpus. In another example, Zhao et al. (2008) counts the number of aligned sentences shared between two templates in a parallel corpus as a measure of their paraphrasing confidence.

Taking all possible substrings or sub-trees as candidates may yield a very large number of candidates. Specifically, the number of possible sub-trees of a parsed sentence is exponential in the size of a sentence. As the number of candidates may be prohibitive, most works filter out candidates prior to scoring them. The two above works limited the size of the template tree and did not process all possible template candidates. Other works retain only those template that repeat the most in relevant occurrences. Typically, relevant occurrences are chosen as containing variable instantiations that co-occur with an input target template (Ravichandran and Hovy, 2002; Szpektor et al., 2004; Pantel and Pennacchiotti, 2006). For example, if the target is to learn rules that one of their templates is ‘$X$ write $Y$’, relevant sentences might contain instantiations such as \{X=’Jane Austen’, Y=’Pride and Prejudice’\}, \{X=’Jorge Luis Borges’, Y=’Pierre Menard, Author of the Quixote’\} or \{X=’Anton Chekhov’, Y=’Ward No. 6’\}.

The second approach for discovering template structure avoids the complexity of all possible sub-trees. Instead, a pre-defined structure for a template is chosen and
all templates of that form that occur in the resource are extracted. A common pre-defined structure for lexical-syntactic binary templates is a dependency path between the two argument nodes (Lin and Pantel, 2001; Ibrahim, Katz, and Lin, 2003; Dinu and Wang, 2009), e.g. ‘$X \xleftarrow{subj} elect \xrightarrow{obj} Y$’ and ‘$X$’s $\xleftarrow{gen} election \xrightarrow{prep-of} Y$’.

Another pre-defined structure is a verb and one of its direct syntactic dependants (e.g. the subject or the object) (Pekar, 2006; Zanzotto, Pennacchiotti, and Pazienza, 2006; Novischi and Moldovan, 2006). As for linear templates, the typical pre-defined structure is the sequence of words between two argument placeholders (Sekine, 2005; Bhagat and Ravichandran, 2008).

We note that another pre-defined structure is simply to take a whole sentence as a template, replacing some of its words by variables (Barzilay and Lee, 2003; Pang, Knight, and Marcu, 2003). This approach is useful for paraphrasing whole sentences, but it does not scale well to broad-coverage knowledge-bases since it is difficult to combine such rules into more complex variations, and thus every possible sentence variation must be learned explicitly as a new rule.

### 2.3.2 Rule Learning

The main goal in this thesis is to learn a broad-scale knowledge-base of entailment rules. Some works presented semi-supervised algorithms for learning entailment rules for a given input template (Ravichandran and Hovy, 2002; Pantel and Pennacchiotti, 2006). Besides the input template, these approaches require as input several examples of the correct template usage in the form of valid instantiations of its arguments. For example, the input template ‘$X$ born in $Y$’ need to be accompanied by some of its correct instantiations, such as \{$X$=‘Mozart’, $Y$=‘1756’\}, \{$X$=‘Escher’, $Y$=‘1898’\} and \{$X$=‘Einstein’, $Y$=‘1879’\}. Such works typically employ a bootstrapping method, learning relevant templates from the input instantiations and then extracting more correct instantiations from the newly learned templates and so on.
These approaches show reasonable accuracy. However, they do not scale well for large-scale acquisition due to the manual supervised input required for each target template. We thus focus in this background on unsupervised approaches for extracting entailment rules from a given resource. These algorithms can be partitioned based on the type of resource they use: (a) manually constructed resources which capture entailment information; (b) comparable and parallel corpora; (c) general corpora.

**Manually Constructed Resources**

Several manually constructed resources contain intrinsic entailment information as part of their structure, such as lexicons and thesauri. Out of these, the most widely used is **WordNet** (Miller, 1995). It is a manually constructed lexical database organized by meanings (a.k.a synsets). Each synset contains a textual description of the meaning it represents and a list of synonymous words that convey that specific meaning under specific contexts. For example, *write*, *compose*, *pen* and *indite* are synonyms in the synset entry “*produce a literary work*”.

Synonymous words are one type of information in WordNet that can be utilized for generating entailment rules. Additionally, WordNet contains, among other information, relations between synsets, such as the *hypernymy* relation, e.g. ‘*run* ⇒ *move*’ and ‘*divorce* ⇒ *separation*’. Some of these relations capture types of inference between synsets, termed here **inferential relations**. The *hypernymy* relation is one type of such inferential relations. Other examples for WordNet inferential relations include the *cause* relation, e.g. ‘*kill* ⇒ *die*’, and the *derivationally related* relation, e.g. ‘*acquisition* ⇔ *acquire*’.

The prominent approach of utilizing WordNet for semantic inference in NLP applications is by extracting lexical entailment rules from it (de Buenaga Rodríquez, Hidalgo, and Díaz-Agudo, 1997; Mandala, Takenobu, and Hozumi, 1998; Cavaglià, 1999; Moldovan and Mihalcea, 2000; Pasca and Harabagiu, 2001; Silber and McCoy,
2.3. RULE GENERATION

Figure 2.1: Part of the *employment* Nomlex entry, describing the possible nominal syntactic dependency positions for each functional role of the transitive subcategorization frame of *employ*.

2002; Dagan, Glickman, and Magnini, 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007). WordNet rules are typically obtained by traversing WordNet’s inferential relations, a process known in the literature as lexical-chaining (Moldovan and Novischi, 2002). For example, the answer to “From which country was Louisiana acquired?” can be inferred from “The United States bought up Louisiana from France” using the WordNet lexical chains ‘France ⇒ European country ⇒ country’ and ‘buy up ⇒ buy ⇒ acquire’.

Another manually constructed resource is Nomlex (Macleod et al., 1998) and its successor Nomlex-plus (Meyers et al., 2004). Nomlex is a database of English nominalizations, and Nomlex-plus currently covers about 5,000 nominalizations. Nomlex contains syntactic information for mapping a nominalization to its related verb, e.g. *employment* to *employ*. For each nominalization, it describes the possible argument realizations of each functional role in each possible subcategorization frame of the corresponding verb. From this information, entailment rules between nominalizations and their related verbs can be generated. A part of the Nomlex entry for *employment* is presented in Figure 2.1. From this entry, entailment rules such as ‘X’s employment of Y ⇔ X employ Y’ can be extracted. Meyers et al. (1998) demonstrate how such Nomlex rules can be utilized for IE.

In Chapter 7, we show how to combine WordNet and Nomlex, and with the addition of corpus-based statistical information how to generate entailment rules between
predicates from these resources.

**Comparable and Parallel Corpora**

Many algorithms were developed for learning rules from comparable corpora and parallel corpora (Shinyama et al., 2002; Barzilay and Lee, 2003; Pang, Knight, and Marcu, 2003; Ibrahim, Katz, and Lin, 2003; Zhao et al., 2008). Comparable corpora are typically monolingual resources that contain topically similar texts. Such a corpus could be collections of diverse sources of news articles from the same date on the same topic. Parallel corpora are typically multilingual resources containing translations of documents to other languages. An example of such a corpus is Europarl (Koehn, 2005), which is extracted from the proceedings of the European Parliament and includes versions in eleven European languages: English, French, Italian, Spanish, Portuguese, Dutch, German, Danish, Swedish, Greek and Finnish. Another example of such corpora is a corpus containing multiple distinct translations of foreign documents to a target language, e.g. different English translations of Chinese articles.

Most algorithms for learning rules from comparable or parallel corpora are based on sentence alignment. First, sentences conveying the same information are clustered. Then, templates that share argument instantiation tuples (termed here anchor-sets) within the clustered sentences are considered as participating in an entailment relation. For example, from the aligned sentences “Bush says he’ll deliver $20 billion to New York”, “Bush, in New York, affirms $20 billion aid pledge” and “Bush reassures New York of $20 billion”, the paraphrasing templates, ‘X say he’ll deliver Z to Y $\iff$ X, in Y, affirms Z aid pledge $\iff$ X reassures Y of Z’, may be learned. Table 2.2 (right column) presents common anchor-sets for the related templates ‘X compose Y’ and ‘X write Y’. Typically, only few common anchor-sets are identified for each entailment relation.

Comparable and parallel corpora are highly informative for identifying variations
Table 2.2: Examples for features for the anchor-set learning approach and single-feature learning approach for two related templates.

of the same meaning. This is since, typically, when variable instantiations are shared across similar documents or aligned translations the same instantiated predicates are described. However, it is hard to collect broad-scale comparable corpora, as the majority of texts are non-comparable, and the same goes for parallel corpora. Thus, rule coverage is hindered by the limited size of such corpora. In addition, the intrinsic nature of these corpora, containing sentences that convey the same meaning, allows mainly to learn paraphrase rules but not as many directional entailment rules. Indeed, the above cited works all aim at learning paraphrases.

As we aim at large-scale acquisition of both paraphrases and directional entailment rules, in this thesis we focus on acquisition from general corpora, as described next.

### General Corpora

Instead of utilizing highly informative resources, several works aim at learning entailment rules from the abundant existing general (non-comparable) corpora (Lin and Pantel, 2001; Szpektor et al., 2004; Sekine, 2005; Pekar, 2006).

The TEASE algorithm (Szpektor et al., 2004) is an unsupervised algorithm that acquires entailment relations from the Web for a given input template using the anchor-set approach. It starts by automatically extracting, from the Web, anchor-sets that are characteristic for the input template. Typically, a specific event or fact is expressed by the template combined with a specific anchor-set. However,
if an anchor-set is characteristic, then the specific event or fact is also (generally) uniquely identified by the anchor-set alone. For example, the anchor-set \{X='William Shakespeare', Y='154 sonnets'\} is characteristic for ‘X compose Y’. Characteristic anchor-sets are detected using statistical tests, such as conditional probability and frequency in (Szpektor et al., 2004). Next, TEASE retrieves sentences from the Web that contain the extracted anchor-sets. It replaces the anchor-sets in these sentences with their corresponding input template variables and parses the sentences. Finally, TEASE searches for the largest connected parse sub-trees containing the variables that repeat the most within the parsed sentences. Such templates, which are termed maximal-most-general templates, cannot be expanded without reducing the number of their occurrences within the retrieved sentences (the maximal attribute). On the other hand, reducing their size (removing template parts) does not increase their occurrence number (the most general attribute). TEASE extracts these templates as participating in an entailment relation with the input template.

Using a similar anchor-set approach, Sekine (2005) acquires paraphrases from a local corpus. In this algorithm, all templates with a predefined structure (a substring between two variables) are first clustered. The clustering is performed by first automatically identifying the template term that represents the predicate in the template. Then, all templates that share the same predicate term are clustered together. For example, the templates ‘X agreed to buy Y’, ‘X to buy Y’ and ‘X offered to buy Y’ are clustered together after detecting buy as the predicate term in each template. Next, anchor-sets are extracted from a local corpus for all the templates in each cluster, and are considered as the anchor-sets of the cluster. Finally, clusters that share anchor-sets are linked together. All the templates within linked clusters are taken as paraphrases.

In a different approach, Pekar (2006) learns rules only between templates related by local discourse. This algorithm relies on discourse information as a reliable link
2.3. RULE GENERATION

between related templates. However, information from different documents in the corpus is ignored. The algorithm extract pairs of verb templates as candidates for an entailment rule if they share an argument instantiation in occurrences within the same paragraph. For example, from the two consecutive sentences “Mary bought a house” and “The house belongs to Mary”, the candidate entailment relations {‘X buy’, ‘belong to X’} and {‘buy X’, ‘X belong’} are extracted. The direction of each candidate rule is chosen by measuring the contribution of one template to the amount of information the second template contains. This is calculated by the Kullback-Leibler Divergence score (Cover and Thomas, 1991) between the distribution of the second template variable instantiations in the whole corpus and their distribution from mentions that occur near mentions of the first template. The direction with the highest score is chosen as the rule’s direction.

The prominent algorithm for learning rules from general corpora is the DIRT algorithm (Lin and Pantel, 2001). It is based on the Distributional Hypothesis, which states that words that occur in the same contexts tend to have similar meanings (Harris, 1954). DIRT learns binary rules for templates that are paths in a dependency parse-tree between two noun variables X and Y. Instead of anchor-sets, it uses simpler, less informative but more frequent, features for each template: it constructs a feature vector for each variable of each binary template, representing the context words that fill the variable in the different occurrences of the template in the corpus. Table 2.2 shows examples for features of this type. Each instantiation is weighted by its point-wise mutual information (PMI) score with the template (Church and Hanks, 1990). Two templates t and s are identified as semantically related if they have similar vectors, based on the lexical Lin similarity measure (Lin, 1998a) adapted to templates (Equation 2.2). The final similarity score is the geometric average of the
vector similarity scores for the two variables (Equation 2.1):

\[
\text{DIRT}(t, s) = \sqrt{\text{DIRT}^X(t, s) \cdot \text{DIRT}^Y(t, s)} \quad (2.1)
\]

\[
\text{DIRT}^\Lambda(t, s) = \frac{\sum_{f \in F\text{V}_t^\Lambda \cap F\text{V}_s^\Lambda} [w_t^\Lambda(f) + w_s^\Lambda(f)]}{\sum_{f \in F\text{V}_t^\Lambda} w_t^\Lambda(f) + \sum_{f \in F\text{V}_s^\Lambda} w_s^\Lambda(f)} \quad (2.2)
\]

where $F\text{V}_z^\Lambda$ is the feature co-occurrence vector of a template $z$ for variable $\Lambda$ (either $X$ or $Y$) and $w_z^\Lambda(f)$ is the weight of the instantiation $f$ in that template’s vector.

In Chapter 6 we revisit distributional similarity measures, investigating the adequacy of directional measures for entailment rule learning.

### 2.4 Validity of Rule Application in Context

Another important aspect of entailment rules addresses their utilization within NLP applications. As described in Section 2.1, rules are applied to a text $t$ by matching the entailing template to the text. We follow common practice (de Salvo Braz et al., 2005; Romano et al., 2006; Bar-Haim et al., 2007) for the basic inference operation of matching at the syntactic representation level: a template is syntactically matched in a text $t$ if it can be embedded in $t$’s parse tree. The matching induces a mapping between the template’s variables and their instantiation in $t$.

A prominent issue of rule application is that rules should typically be applied only in specific contexts, which we term **relevant contexts**. For example, the rule ‘$X$ acquire $Y$ $\Rightarrow$ $X$ buy $Y$’ can be used in the context of **buying events**. However, it shouldn’t be applied to “Students acquired a new language”. Similarly, the rule ‘$X$ charge $Y$ $\Rightarrow$ $X$ accuse $Y$’ should not be applied to “This store charged my account”.
2.4. VALIDITY OF RULE APPLICATION IN CONTEXT

since the assumed sense of charge in the rule is different than its sense in the text.

Context sensitive inference was mainly investigated in an application-dependent manner. For example, Harabagiu, Maiorano, and Pașca (2003) describe techniques for identifying the question focus and the answer type in QA. Patwardhan and Riloff (2007) propose a supervised approach for IE, in which relevant text regions for a target relation are identified prior to applying extraction rules.

Recently, some generic methods were proposed to handle context-sensitive inference. Sekine (2005) learns entailment rules for typed arguments (Person, Organization etc.). Pantel et al. (2007) propose to learn the preferred instantiations of rule variables, termed Inferential Selectional Preferences (ISP). Their algorithm derives inferential selectional preferences for a rule ‘LHS ⇒ RHS’ by aggregating over the argument instantiations that are shared between the two templates of the rule over a large corpus. In (Pantel et al., 2007) the shared instantiations are aggregated by taking their semantic classes (Resnik, 1996), which may come from a taxonomy such as WordNet or from a word clustering algorithm such as CBC (Pantel and Lin, 2002). For example, an aggregated class of the instantiations Bank of England, clearing house and insurance company might be financial institution, institution, or organization, according to WordNet. To decide whether the context of a new occurrence of the rule’s LHS template is relevant for rule application, the algorithm checks whether the semantic classes chosen for the rule’s arguments include the variable instantiations of the new occurrence as well. A similar approach to ISP is taken in (Pennacchiotti et al., 2007). However, instead of the explicit representation of semantic classes, LSA similarity is used to compare between the preferred variable instantiations for a rule and a new LHS instantiation. Downey, Schoenmackers, and Etzioni (2007) use an HMM-based similarity for the same purpose.

Generic approaches were also proposed in (Dagan et al., 2006; Connor and Roth, 2007) for identifying valid applications of lexical rules by classifying the surrounding
global context of the entailing word as valid or not for that rule application. These approaches extract contextual features for the rule’s entailing and entailed terms from a corpus. Features that are shared between the two terms are assumed to indicate the valid context under which the rule should be applied. Given a new occurrence of the rule LHS, the context surrounding this occurrence is compared against the previously extracted contexts to decide whether the current context is valid for rule application. As examples for a rule’s context, (Dagan et al., 2006) selected lemmas and part-of-speech tags in the current, preceding and following sentence of each occurrence of each term participating in the lexical rule, as well as those at specific positions in a window around the term. (Connor and Roth, 2007) selected their context as specific syntactic positions related to each term (e.g. the subject and object of a verb term) and all noun phrases in the sentences containing the term.

In Chapter 8 we propose a novel generic framework for context-sensitive rule application, which generalizes and extends prior work.

It is interesting to note that though context sensitive application of rules is critical, in many cases NLP applications implicitly incorporate some contextual constraints when applying a rule. For example, when answering the question “Which companies did IBM buy?” a QA system would apply the rule ‘$X$ acquire $Y \Rightarrow X$ buy $Y$’ correctly, since the phrase “IBM acquire $X$” is likely to be found mostly in relevant economic contexts.

### 2.5 Rule Evaluation

Once an algorithm is devised for learning rules, we need ways for measuring the quality of the extracted rules and comparing them to other algorithms and rule resources. The prominent approach for evaluating rules is to directly assess the learned rules through manual human annotation (Lin and Pantel, 2001; Shinyama et al., 2002;
Duclaye, cois Yvon, and Collin, 2003; Barzilay and Lee, 2003; Pang, Knight, and Marcu, 2003; Ibrahim, Katz, and Lin, 2003; Szpektor et al., 2004; Sekine, 2005; Pantel and Pennacchiotti, 2006; Bhagat and Ravichandran, 2008; Zhao et al., 2008). Under this scheme a sample of rules are presented to human judges, which evaluate whether each rule is correct or not.

As the target of learning rules is to utilize them for recognizing semantic variability in applications, few works tested the benefit of the learned rules to specific application, such as QA (Ravichandran and Hovy, 2002; Novischi and Moldovan, 2006) and IE (Sudo, Sekine, and Grishman, 2003; Romano et al., 2006). Under this scheme, an NLP system is additionally provided with the rules and the performance of the two configurations (with and without rules) is compared.

As for evaluating context-sensitive rule application methods, both manual evaluation and application-based evaluation settings were utilized. Pantel et al. (2007) and Pennacchiotti et al. (2007) selected a sample of learned rules and manually evaluated a sample of their applications within a corpus. A context-sensitive approach was then tested whether it could filter out the sampled applications that were manually labeled as incorrect, while retaining those that were labeled as correct. Downey, Schoenmackers, and Etzioni (2007) chose an IE setup, where the quality of the arguments extracted by the learned rules for a set of target IE relations was manually evaluated. The evaluation results with and without context-sensitive methods were compared.

In Chapter 4, we propose two novel schemes for entailment rule evaluation. These schemes are later employed for evaluating rule learning algorithms and approaches for context-sensitive rule application.
Chapter 3

Contribution of Lexical-Syntactic Entailment Rules to Inference

3.1 Introduction

The background for entailment rule research was set in Chapter 2, but before we present our contributions in the area of entailment rule learning and application, this chapter analyzes the role that entailment rules (at the syntactic representation level) take within the task of Textual Entailment.

Identifying entailment is a complex task that incorporates many levels of linguistic knowledge and inference. The complexity of modeling entailment was demonstrated already at the first PASCAL Challenge Workshop on Recognizing Textual Entailment (RTE-1) (Dagan, Glickman, and Magnini, 2006). Systems that participated in the challenge used various combinations of NLP components in order to perform entailment inferences. These components can largely be classified as operating at the lexical, syntactic and semantic levels. However, only little research was done to analyze the contribution of each inference level, and on the contribution of individual

\footnote{Joint work with Roy Bar-Haim}
inference mechanisms within each level.

We suggest that decomposing the complex task of entailment into subtasks and analyzing the contribution of individual NLP components for these subtasks would make a step towards better understanding of the problem. Three goals are set. First, we consider two modeling levels that employ only part of the inference mechanisms, but perform perfectly at each level. The levels we focus on are the lexical and lexical-syntactic, which are represented by bag-of-words and bag-of-parsed-dependency-relations respectively. We explore how well these models approximate the notion of entailment, and analyze the differences between the outcome of the different levels. Second, for each of the presented levels, we evaluate the distribution (and contribution), over the RTE-1 dataset, of each of the inference mechanisms typically associated with that level. Finally, we suggest that the definitions of entailment at different levels of inference, as proposed in this chapter, can serve as guidelines for manual annotation of a “gold standard” for evaluating systems that operate at a particular level. Altogether, we set forth a possible methodology for detailed annotation and analysis of entailment datasets.

3.2 Definition of Entailment Levels

As a first step of our analysis, we present definitions for two entailment models that correspond to the Lexical and Lexical-Syntactic levels. For each level we describe the available inference mechanisms. Table 3.1 presents several examples from the RTE test-set together with annotation of entailment at the different levels.

3.2.1 The Lexical Entailment Level

At the lexical level we assume that the text $T$ and hypothesis $H$ are represented by a bag of (possibly multi-word) terms, ignoring function words. At this level we define
### Table 3.1: Examples of text-hypothesis pairs, taken from the RTE-1 test-set.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>322</td>
<td>Turnout for the historic vote for the first time since the EU took in 10 new members in May has hit a record low of 45.3%.</td>
<td>New members joined the EU.</td>
<td>IR</td>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>1361</td>
<td>A Filipino hostage in Iraq was released.</td>
<td>A Filipino hostage was freed in Iraq.</td>
<td>CD</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>1584</td>
<td>Although a Roscommon man by birth, born in Rooskey in 1932, Albert “The Slasher” Reynolds will forever be a Longford man by association.</td>
<td>Albert Reynolds was born in Co. Roscommon.</td>
<td>QA</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>1911</td>
<td>The SPD got just 21.5% of the vote in the European Parliament elections, while the conservative opposition parties polled 44.5%.</td>
<td>The SPD is defeated by the opposition parties.</td>
<td>IE</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>2127</td>
<td>Coyote shot after biting girl in Vanier Park.</td>
<td>Girl shot in park.</td>
<td>IR</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

that entailment holds between $T$ and $H$ if every term $h$ in $H$ can be matched by a corresponding entailing term $t$ in $T$. $t$ is considered as entailing $h$ if either $h$ and $t$ share the same lemma and part of speech, or $t$ can be matched with $h$ through a sequence of lexical entailment transformations of the types described below.

**Morphological Derivations** This inference mechanism considers two terms as equivalent if one can be obtained from the other by some morphological derivation. Examples include nominalizations (e.g. ‘acquisition ⇔ acquire’), pertainyms (e.g. ‘Afghanistan ⇔ Afghan’) and nominal derivations like ‘terrorist ⇔ terror’. 
3.2. DEFINITION OF ENTAILMENT LEVELS

**Ontological Relations** This inference mechanism refers to ontological relations between terms. A term is entailed from another term if a chain of valid ontological relations between the two terms exists (Andreevskaja, Li, and Bergler, 2005). In our experiment we regarded the following three ontological relations as providing entailment inferences: (a) **synonyms** (e.g., ‘free ⇔ release’ in example 1361, Table 3.1); (b) **hypernym** (e.g., ‘produce ⇒ make’); (c) **meronym-holonym** (e.g., ‘executive ⇒ company’).

**Lexical World Knowledge** This inference mechanism refers to world knowledge reflected at the lexical level, by which the meaning of one term can be inferred from the other. It includes both knowledge about named entities, such as ‘Taliban ⇒ organization’ and ‘Roscommon ⇔ Co. Roscommon’ (example 1584 in Table 3.1), and other lexical relations between words, such as WordNet’s relations **cause** (e.g., ‘kill ⇒ die’) and **entail** (e.g., ‘snore ⇒ sleep’).

### 3.2.2 The Lexical-syntactic Entailment Level

At the lexical-syntactic level we assume that the text and the hypothesis are represented by the set of syntactic dependency relations of their dependency parse. At this level we ignore determiners and auxiliary verbs, but include relations involving other function words. We define that entailment holds between $T$ and $H$ if the relations within $H$ can be “covered” by the relations in $T$. In the trivial case, lexical-syntactic entailment holds if all the relations composing $H$ appear verbatim in $T$ (while additional relations within $T$ are allowed). Otherwise, such coverage can be obtained by a sequence of entailment transformations applied to the relations in $T$, which should yield all the relations in $H$.

One type of such transformations are the lexical transformations, which replace
corresponding lexical items, as described in Section 3.2.1. When applying morpholog-
cal derivations it is assumed that the syntactic structure is appropriately adjusted.
For example, “Mexico produces oil” can be mapped to “oil production by Mexico”
(the Nomlex resource (Macleod et al., 1998) provides a good example for systematic
specification of such transformations).

Additional types of transformations at this level are specified below.

**Syntactic Transformations** This inference mechanism refers to transformations
between syntactic structures that involve the same lexical elements and preserve the
meaning of the relationships between them (as analyzed in (Vanderwende and Dolan,
2005)). Typical transformations include passive-active and apposition (e.g. “An
Wang, a native of Shanghai ⇔ An Wang is a native of Shanghai”).

**Lexical Syntactic Entailment Rules** This inference mechanism refers to the
focus of this thesis: entailment rules represented at the syntactic level. We focus on
rules that are local in nature, which represent building block transformations that can
be combined into complex inference (as detailed in Chapter 2). Examples include:
‘X is Y man by birth ⇒ X was born in Y’ (example 1584 in Table 3.1), ‘X take in Y
⇔ Y join X’¹ and ‘X is holy book of Y ⇒ Y follow X’².

**Co-reference** Co-references provide equivalence relations between different terms
in the text and thus induce transformations that replace one term in a text with
any of its co-referenced terms. For example, the sentence “Italy and Germany have
each played twice, and they haven’t beaten anybody yet.”³ entails “Neither Italy nor
Germany have won yet”, involving the co-reference transformation ‘they ⇒ Italy and

¹Example no 322 in RTE-1 test-set.
²Example no 1575 in RTE-1 test-set.
³Example no 298 in RTE-1 test-set.
Example 1584 in Table 3.1 demonstrates the need to combine different inference mechanisms to achieve lexical-syntactic entailment, requiring world-knowledge, entailment rules and syntactic transformations.

3.3 Empirical Analysis

Following the definition of our entailment models, we next conduct an experiment that helps analyzing the two entailment levels in terms of relative performance and correlation with the notion of textual entailment.

3.3.1 Data and Annotation Procedure

The RTE-1 test-set\textsuperscript{1} contains 800 Text-Hypothesis pairs (usually single sentences), which are typical to various NLP applications. Each pair is annotated with a boolean value, indicating whether the hypothesis is entailed by the text or not, and the test-set is balanced in terms of positive and negative cases. We shall henceforth refer to this annotation as the gold standard. We sampled 240 pairs from four different tasks in the test-set, which correspond to the main applications that may benefit from entailment: Information Extraction (IE), Information Retrieval (IR), Question Answering (QA), and Comparable Documents (CD). We randomly picked 60 pairs from each task, and in total 118 of the cases were positive and 122 were negative.

In our experiment, two judges annotated, for each of the two levels, whether or not entailment can be established in each of the 240 pairs. The annotators agreed on 89.6\% of the cases at the lexical level, and 88.8\% of the cases at the lexical-syntactic

\textsuperscript{1}The complete RTE dataset can be obtained at http://www.pascal-network.org/Challenges/RTE/Datasets/
CHAPTER 3. CONTRIBUTION OF ENTAILMENT RULES TO INFERENCE

<table>
<thead>
<tr>
<th></th>
<th>L</th>
<th>LS</th>
</tr>
</thead>
<tbody>
<tr>
<td>True positive</td>
<td>52</td>
<td>59</td>
</tr>
<tr>
<td>(118)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>False positive</td>
<td>36</td>
<td>10</td>
</tr>
<tr>
<td>(122)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recall</td>
<td>44%</td>
<td>50%</td>
</tr>
<tr>
<td>Precision</td>
<td>59%</td>
<td>86%</td>
</tr>
<tr>
<td>$F_1$</td>
<td>0.5</td>
<td>0.63</td>
</tr>
<tr>
<td>Accuracy</td>
<td>58%</td>
<td>71%</td>
</tr>
</tbody>
</table>

Table 3.2: Results per level of entailment. L stands for the lexical level and LS stands for the lexical syntactic level.

level, with Kappa statistics of 0.78 and 0.73, respectively, corresponding to ‘substantial agreement’ (Landis and Koch, 1977). This relatively high level of agreement suggests that the notion of lexical and lexical-syntactic entailment we propose are indeed well-defined. Finally, in order to establish statistics from the annotations, the annotators discussed all the examples they disagreed on and produced a final joint decision.

We note that the distribution of the examples in the RTE test-set cannot be considered representative of a real-world distribution (especially because of the controlled balance between positive and negative examples). Thus, our statistics are not an accurate prediction of application performance. Instead, we analyze how well these simplified models of entailment succeed in approximating “real” entailment, and how they compare with each other.

3.3.2 Evaluating the Different Levels of Entailment

Table 3.2 summarizes the results obtained from our annotated dataset for both lexical (L) and lexical-syntactic (LS) levels. Taking a “system”-oriented perspective, the annotations at each level can be viewed as the classifications made by an idealized
3.3. EMPIRICAL ANALYSIS

system that includes a perfect implementation of the inference mechanisms in that level. The first two rows show for each level how the examples that were recognized as positive by this level (i.e. entailment holds) are distributed between true positive (i.e. positive according to the gold standard) and false positive (negative according to the gold standard). The total number of positive and negative pairs in the dataset is reported in parentheses. The rest of the table details Recall, Precision, F1 and Accuracy.

The proportion between true and false positive cases at the lexical level indicates that the correlation between lexical match and entailment is quite low, reflected in the low precision achieved by this level (only 59%). This result can be partly attributed to the idiosyncracies of the RTE-1 test-set: as reported in (Dagan, Glickman, and Magnini, 2006), samples with high lexical match were found to be biased towards the negative side. Interestingly, our measured accuracy correlates well with the performance of systems at the RTE-1 Workshop, where the highest reported accuracy of a lexical system is 58.6% (Dagan, Glickman, and Magnini, 2006).

As one can expect, adding syntax considerably reduces the number of false positives - from 36 to only 10. Surprisingly, at the same time the number of true positive cases grows from 52 to 59, and correspondingly, precision rises to 86%. Interestingly, neither the lexical nor the lexical-syntactic level are able to cover more than half of the positive cases (e.g. example 1911 in Table 3.1).

In order to better understand the differences between the two levels, we next analyze the overlap between them, presented in Table 3.3. Looking at Table 3.3(a), which contains only the positive cases, we see that many examples were recognized only by one of the levels. This interesting phenomenon can be explained on the one hand by lexical matches that could not be validated in the syntactic level, and on the other hand by the use of lexical-syntactic entailment rules, which are introduced only in the lexical-syntactic level. (e.g. example 322 in Table 3.1).
36 CHAPTER 3. CONTRIBUTION OF ENTAILMENT RULES TO INFERENCE

\[
\begin{array}{c|c|c|c}
 & \text{Lexical-Syntactic} & \text{Lexical} \\
 \hline
 \text{H} & \text{T} & \text{F} \\
 \hline
 \text{H} \Rightarrow \text{T} & 38 & 14 \\
 \text{H} \Rightarrow \text{F} & 21 & 45 \\
 \hline
\end{array}
\]

(a) positive examples

\[
\begin{array}{c|c|c|c}
 & \text{Lexical-Syntactic} & \text{Lexical} \\
 \hline
 \text{H} & \text{T} & \text{F} \\
 \hline
 \text{H} \Rightarrow \text{T} & 7 & 29 \\
 \text{H} \Rightarrow \text{F} & 3 & 83 \\
 \hline
\end{array}
\]

(b) negative examples

Table 3.3: Correlation between the entailment levels. (a) includes only the positive examples from the RTE dataset sample, and (b) includes only the negative examples.

This relatively symmetric statistics changes as we move to the negative cases, as shown in Table 3.3(b). By adding syntactic constraints, the lexical-syntactic level was able to fix 29 false positive errors, misclassified at the lexical level (as demonstrated in example 2127, Table 3.1), while introducing only 3 new false-positive errors. This exemplifies the importance of syntactic matching for precision.

3.3.3 The Contribution of Various Inference Mechanisms

In order to get a sense of the contribution of the various components at each level, statistics on the inference mechanisms that contributed to the coverage of the hypothesis by the text (either full or partial) were recorded by one annotator. Only the positive cases in the gold standard were considered.

For each inference mechanism we measured its frequency, its contribution to the recall of the related level and the percentage of cases in which it is required for establishing entailment. The latter also takes into account cases where only partial coverage could be achieved, and thus indicates the significance of each inference mechanism
3.3. **EMPIRICAL ANALYSIS**

<table>
<thead>
<tr>
<th>Inference Mechanism</th>
<th>f</th>
<th>$\Delta R$</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synonym</td>
<td>19</td>
<td>14.4%</td>
<td>16.1%</td>
</tr>
<tr>
<td>Morphological</td>
<td>16</td>
<td>10.1%</td>
<td>13.5%</td>
</tr>
<tr>
<td>Lexical World knowledge</td>
<td>12</td>
<td>8.4%</td>
<td>10.1%</td>
</tr>
<tr>
<td>Hypernym</td>
<td>7</td>
<td>4.2%</td>
<td>5.9%</td>
</tr>
<tr>
<td>Mernoym</td>
<td>1</td>
<td>0.8%</td>
<td>0.8%</td>
</tr>
<tr>
<td><strong>Entailment rules</strong></td>
<td>37</td>
<td>26.2%</td>
<td>31.3%</td>
</tr>
<tr>
<td><strong>Syntactic transformations</strong></td>
<td>22</td>
<td>16.9%</td>
<td>18.6%</td>
</tr>
<tr>
<td><strong>Coreference</strong></td>
<td>10</td>
<td>5.0%</td>
<td>8.4%</td>
</tr>
</tbody>
</table>

Table 3.4: The frequency ($f$), contribution to recall ($\Delta R$) and percentage (%), within the gold standard positive examples, of the various inference mechanisms at each level, ordered by their significance.

for any entailment system, regardless of the models presented in this chapter. The results are summarized in Table 3.4.

The main result from this analysis is the importance of lexical-syntactic entailment rules for textual inference. From the table it stands that entailment rules are the most notable contributors to recall, indicating their importance to the entailment task. This result also raises the need for large-scale collections of such rules, as many of the rules annotated cannot be found in existing collections.

Syntactic transformations are also shown to contribute considerably, indicating the need for collections of syntactic transformations as well. In that perspective, we propose our annotation framework as means for evaluating collections of entailment rules or syntactic transformations in terms of recall.

Finally, we note that the co-reference moderate contribution can be partly attributed to the idiosyncrasies of the RTE-1 test-set: the annotators were guided to replace anaphors with the appropriate reference, as reported in (Dagan, Glickman, and Magnini, 2006).
3.4 Related Work

Our study follows on previous work (Vanderwende and Dolan, 2005), which analyzed the RTE-1 test-set to find the percentage of cases in which syntactic analysis alone (with optional use of thesaurus for the lexical level) suffices to decide whether or not entailment holds. Our study extends this work by considering a broader range of inference levels and inference mechanisms and providing a more detailed view. Another fundamental difference between the two works is that while Vanderwende and Dolan (2005) did not make judgements on cases where additional knowledge was required beyond syntax, our entailment models were evaluated over all of the cases, including those that require higher levels of inference. This allows us to view the entailment model at each level as an idealized system approximating full entailment, and to evaluate its overall success.

Later analyses of the RTE datasets also investigated the contribution of lexical and syntactic information to Textual Entailment task. Blake (2007) showed that taking sentence structure into consideration can substantially improve entailment accuracy. Clark et al. (2007) analyzed the contribution of different lexical information resources (disregarding the different representation levels in which they operate). One of their main observations is that the vast majority of positive entailment pairs require significant world knowledge for successfully inferring the hypothesis from the text. Interestingly, their definition of world-knowledge largely coincides with our definition of entailment rules. Thus, their analysis supports our observation that entailment rules are an important knowledge representation for Textual Entailment.
3.5 Conclusions

In this chapter we presented the definition of two entailment models, Lexical and Lexical-Syntactic, and analyzed their performance manually. Our experiment shows that the lexical-syntactic level outperforms the lexical level in all measured aspects. Furthermore, lexical-syntactic entailment rules emerged as the most important contributor to recall.

The analysis in this chapter suggests that a lexical-syntactic framework is a promising step towards a complete entailment model. It provides an empirical support to the choice of representing predicative entailment rules at the syntactic level. The analysis also supports the focus on entailment rules in general as an important contributor to textual inference, though it is clear that other types of rules and inference mechanisms are required to achieve a complete coverage of linguistic variability and semantic inference.

Our analysis showed that entailment rules at the syntactic representation level are indeed a well worth path to investigate. We next turn to investigate evaluation methodologies of entailment rules and of state-of-the-art learning algorithms. If we want to develop new approaches for acquiring rules, we need a robust evaluation procedure for comparing between various algorithms and different rule-bases. In the following chapter we propose novel entailment rule evaluation schemes that attempt to improve over the limitations of current evaluation methods.
Chapter 4

Evaluation of Entailment Rules

4.1 Introduction

Despite the abundance of methods for automatic acquisition of rules that have been suggested in recent years (see Section 2.3), there is still no commonly accepted framework for their evaluation. Their current typical mediocre performance level only stresses the obvious need for satisfactory evaluation methodologies that would drive future research. Furthermore, the vast majority of these methods learn rules as pairs of templates \( \{L, R\} \) in a symmetric manner, without addressing rule directionality. Accordingly, most previous works (except (Szpektor et al., 2004) and (Pekar, 2006)) evaluated the learned rules under the paraphrasing criterion, expecting bi-directional entailment. Since most application settings require only directional entailment, such evaluations underestimate the practical utility of the learned rules.

One approach which was used for evaluating automatically acquired rules is to measure their contribution to the performance of specific application systems, such as QA or IE (see Section 2.5 for prior work). While measuring the impact of a rule knowledge-base on applications is highly important, current utilization of this approach is lacking. First, developers of acquisition algorithms often do not have access
to the different applications that will later use the learned rules as generic modules. Second, the learned rules may affect individual systems and applications differently, thus making observations that are based on different systems incomparable. Third, within a complex application system it is difficult to assess the exact quality of entailment rules independently of effects of other system components.

Thus, as in many other NLP learning settings, most works turned to a direct evaluation of their acquisition methods. The prominent approach for evaluating rules, termed here the rule-based approach, is to present the learned rules to human judges asking whether each rule is correct or not (see also Section 2.5). An example of manual evaluation is presented in Table 4.1 for DIRT (Lin and Pantel, 2001) and TEASE (Szpektor et al., 2004), two prominent acquisition algorithms whose outputs are publicly available.

Alas, the rule-based methodology turns out to be problematic because the rule correctness criterion is not sufficiently well defined and is hard to apply. By the common view of context relevance for rules (see the definition in Section 2.4), a rule is considered correct if the judge could think of reasonable contexts under which the rule holds. However, while some rules might be easily judged as correct or incorrect (see Table 4.1), judgment is often more difficult due to context relevance. One judge might come up with a certain context that, to her opinion, validates the rule, while another judge might not imagine that context or think that it does not sufficiently support rule correctness. For example, in our experiments one of the judges did not identify the valid religious holidays context for the correct rule ‘X observe Y ⇒ X celebrate Y’ and thus judged the rule as incorrect.

Indeed, the standards for evaluation in this field are lower than in other fields. The criterion for rule correctness is not explicitly described in most previous works. In addition, many papers do not report inter-judge agreement levels at all and those that do, report rather low Kappa agreement levels (Cohen, 1960), such as 0.54 (Barzilay
CHAPTER 4. EVALUATION OF ENTAILMENT RULES

<table>
<thead>
<tr>
<th>Input</th>
<th>Correct</th>
<th>Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>X change Y</td>
<td>(→) X modify Y</td>
<td>X adopt Y</td>
</tr>
<tr>
<td>(DIRT)</td>
<td>(←) X amend Y</td>
<td>X create Y</td>
</tr>
<tr>
<td></td>
<td>(←) X revise Y</td>
<td>X stick to Y</td>
</tr>
<tr>
<td>X change Y</td>
<td>(→) X alter Y</td>
<td>X maintain Y</td>
</tr>
<tr>
<td>(TEASE)</td>
<td>(←) X affect Y</td>
<td>X follow Y</td>
</tr>
<tr>
<td></td>
<td>(←) X extend Y</td>
<td>X use Y</td>
</tr>
</tbody>
</table>

Table 4.1: Examples of templates suggested by DIRT and TEASE as having an entailment relation, in some direction, with the input template ‘X change Y’. The entailment direction arrows were judged manually and added for readability.

and Lee, 2003) and 0.55 - 0.63 (Szpektor et al., 2004). We have also replicated the rule-based methodology but did not manage to reach a 0.6 Kappa agreement level between pairs of judges. Yet, it is crucial to reliably assess rule correctness in order to measure and compare the performance of different algorithms in a replicable manner. Lacking a good evaluation methodology has become a barrier for further advances in the field.

In this chapter, we aim at improving the evaluation methodologies for entailment rules. Two approaches for the evaluation of knowledge-bases of entailment rules are presented. The first is a manual human evaluation that aims to remedy the problems of the current rule-based approach. The second approach is an automatic application-based evaluation, providing a simple replicable framework that is very useful for comparing between different knowledge-bases of rules. These two approaches were developed in a chronological order and are presented in the order of their evolution.

4.2 Instance-based Evaluation Methodology

The goal of entailment rules is to help applications infer one text variant from another. A rule ‘L ⇒ R’ can be applied to a given text only when L can be inferred from it,
4.2. INSTANCE-BASED EVALUATION METHODOLOGY

with appropriate variable instantiation. Then, using the rule, the application deduces that $R$ can also be inferred from the text under the same variable instantiation. Thus, an evaluation methodology for entailment rules should reflect the expected validity of their application within NLP systems.

Following that line, a rule ‘$L \Rightarrow R$’ should be regarded as correct if in all (or at least most) relevant contexts in which the instantiated template $L$ can be inferred from the given text, the instantiated template $R$ can also be inferred from the text. This reasoning corresponds to the common definition of entailment in semantics, which specifies that a text $L$ entails another text $R$ if $R$ is true in every circumstance (possible world) in which $L$ is true (Chierchia and McConnell-Ginet, 2000).

It follows that in order to assess if a rule is correct we should judge whether $R$ is typically entailed from those sentences that entail $L$ (within relevant contexts for the rule). We thus present a new evaluation scheme for entailment rules, termed the instance-based approach. At the heart of this approach, human judges are presented not only with a rule but rather with a set of examples of the rule’s usage. Instead of thinking up valid contexts for the rule the judges need to assess the rule’s validity under the given context in each example. The essence of our proposal is a (apparently non-trivial) protocol of a sequence of questions, which determines rule validity in a given sentence.

We shall next describe how we collect a sample of examples for evaluation and the evaluation process.

4.2.1 Example Collection

Given a rule ‘$L \Rightarrow R$’, our goal is to generate evaluation examples by finding a sample of sentences from which $L$ is entailed. We do that by automatically retrieving, from a given corpus, sentences that match $L$ and are thus likely to entail it, as explained below.
CHAPTER 4. EVALUATION OF ENTAILMENT RULES

<table>
<thead>
<tr>
<th>Rule</th>
<th>Sentence</th>
<th>Judgment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 X clarify Y ⇒ X prepare Y</td>
<td>He didn’t clarify his position on the subject.</td>
<td>Left not entailed</td>
</tr>
<tr>
<td>2 X seek Y ⇒ X disclose Y</td>
<td>If he is arrested, he can immediately seek bail.</td>
<td>Left not entailed</td>
</tr>
<tr>
<td>3 X hit Y ⇒ X approach Y</td>
<td>Other earthquakes have hit Lebanon since ’82.</td>
<td>Irrelevant context</td>
</tr>
<tr>
<td>4 X lose Y ⇒ X surrender Y</td>
<td>Bread has recently lost its subsidy.</td>
<td>Irrelevant context</td>
</tr>
<tr>
<td>5 X regulate Y ⇒ X reform Y</td>
<td>The SRA regulates the sale of sugar.</td>
<td>No entailment</td>
</tr>
<tr>
<td>6 X resign Y ⇒ X share Y</td>
<td>Lopez resigned his post at VW last week.</td>
<td>No entailment</td>
</tr>
<tr>
<td>7 X set Y ⇒ X allow Y</td>
<td>The committee set the following refunds.</td>
<td>Entailment holds</td>
</tr>
<tr>
<td>8 X stress Y ⇒ X state Y</td>
<td>Ben Yahia also stressed the need for action.</td>
<td>Entailment holds</td>
</tr>
</tbody>
</table>

Table 4.2: Rule evaluation examples and their judgment.

For each example sentence, we automatically extract the arguments that instantiate L and generate two phrases, termed left phrase and right phrase, which are constructed by instantiating the left template L and the right template R with the arguments extracted. For example, the left and right phrases generated for example 1 in Table 4.2 are “he seek bail” and “he disclose bail”, respectively.

Finding sentences that match L can be performed at different levels. In this thesis we match lexical-syntactic templates by finding a sub-tree of the sentence parse that is identical to the template structure. Of course, this matching method is not perfect and will sometimes retrieve sentences that do not entail the left phrase for various reasons, such as incorrect sentence analysis or semantic aspects like negation, modality and conditionals. See examples 1-2 in Table 4.2 for sentences that syntactically match L but do not entail the instantiated left phrase. Since we should assess R’s entailment only from sentences that entail L, such sentences should be ignored as an integral part
4.2. INSTANCE-BASED EVALUATION METHODOLOGY

of the evaluation process.

4.2.2 Example Judgment Questions

For each example generated for a rule, the judges are presented with the given sentence and the left and right phrases. They then answer two questions that assess whether entailment holds in this example, following the semantics of entailment rule application as discussed above:

Q_{le}: Is the left phrase entailed from the sentence?
A positive/negative answer corresponds to a ‘Left entailed/not entailed’ judgment.

Q_{re}: Is the right phrase entailed from the sentence?
A positive/negative answer corresponds to an ‘Entailment holds/No entailment’ judgment.

The first question identifies sentences that do not entail the left phrase, and thus should be ignored when evaluating the rule’s correctness. The second question assesses whether the rule application is valid or not for the current example. See examples 5-8 in Table 4.2 for cases where entailment does or does not hold.

By this methodology, the judges focus only on the given sentence in each example, so the task is actually to evaluate whether Textual Entailment holds between the sentence and each of the left and right phrases. Following past experience in textual entailment evaluation (Dagan, Glickman, and Magnini, 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007) we expect a reasonable agreement level between judges.

We may want to ignore examples whose context is irrelevant for the rule, expecting that systems would typically not apply it in such contexts. To optionally capture this distinction the judges are asked another question:
Qrc: Is the right phrase a likely phrase in English?

A positive/negative answer corresponds to a ‘Relevant/Irrelevant context’ evaluation.

If the right phrase is not likely in English then the given context is probably irrelevant for the rule, because it seems inherently incorrect to infer an implausible phrase. Examples 3-4 in Table 4.2 demonstrate cases of irrelevant contexts, which we may choose to ignore when assessing rule correctness.

4.2.3 Evaluation Process

For each example, the judges are presented with the three questions above in the following order: (1) Qle (2) Qrc (3) Qre. If the answer to a certain question is negative then we do not need to present the next questions to the judge: if the left phrase is not entailed then we ignore the sentence altogether; and if the context is irrelevant then the right phrase cannot be entailed from the sentence and so the answer to Qre is already known as negative.

The above entailment judgments assume that we can actually ask whether the left or right phrases are correct given the sentence, that is, we assume that a truth value can be assigned to both phrases. This is the case when the left and right templates correspond, as expected, to semantic relations (e.g. a predicate and its arguments). Yet sometimes the learned templates are (erroneously) not relational, e.g. ‘X, Y, IBM’ (representing a list). We therefore let the judges initially mark rules that include such templates as non-relational, in which case their examples are not evaluated at all.
4.2.4 Rule Precision

We compute the precision of a rule by the percentage of examples for which entailment holds out of all “relevant” examples. We can calculate the precision in two ways, depending on whether we ignore irrelevant contexts or not (obtaining lower precision if we don’t). In real systems, irrelevant contexts are often avoided but not always. Thus, the two options can be viewed as upper and lower bounds for the expected precision of the rule applications in actual systems, as follows:

**upper bound precision:** \[
\frac{\text{\#Entailment holds}}{\text{\#Relevant context}}
\]

**lower bound precision:** \[
\frac{\text{\#Entailment holds}}{\text{\#Left entailed}}
\]

where \# denotes the number of examples with the corresponding judgment.

Finally, we consider a rule to be correct only if its precision is at least 80%, which seems a sensible choice for typical applied settings. This yields two sets of correct and incorrect rules, corresponding to the upper bound and lower bound precision figures. Even though judges may disagree on specific examples for a rule, their judgments may still agree overall on the rule’s correctness. We therefore expect the agreement level on rule correctness to be higher than the agreement on individual examples.
4.3 Instance-based Evaluation Experiment

4.3.1 Experimental Settings

We applied the instance-based methodology to evaluate DIRT and TEASE algorithms. As mentioned, these are two prominent state-of-the-art unsupervised acquisition algorithms and the only ones whose output was publicly available at the time of our experiments. DIRT identifies semantically related templates in a local corpus using distributional similarity over the templates’ variable instantiations. TEASE acquires entailment relations from the Web for a given input template $I$ by identifying characteristic variable instantiations shared by $I$ and other templates. To the best of our knowledge, this is the first comparison between the two state-of-the-art acquisition algorithms.

For the experiment we used the published DIRT and TEASE knowledge-bases, available from the textual entailment resource pool\(^1\). For every given input template $I$, each knowledge-base can be viewed as providing a list of learned output templates \(\{O_j\}_{1}^{n_I}\), where \(n_I\) is the number of output templates learned for $I$. Each output template is suggested as holding an entailment relation with the input template $I$, but the algorithms do not specify the entailment direction(s). Thus, each pair \(\{I, O_j\}\) induces two candidate directional entailment rules: ‘$I \Rightarrow O_j$’ and ‘$O_j \Rightarrow I$’.

Test Set Construction

The test set construction consists of three sampling steps: selecting a set of input templates for the two algorithms, selecting a sample of output rules to be evaluated, and selecting a sample of sentences to be judged for each rule.

First, we randomly selected 30 transitive verbs out of the 1000 most frequent

\(^1\)http://aclweb.org/aclwiki/index.php?title=Textual_Entailment_Resource_Pool
verbs in the Reuters RCV1 corpus\(^1\). For each verb we manually constructed a lexical-syntactic input template by adding subject and object variables. For example, for the verb *seek* we constructed the template ‘X seek Y’.

Next, for each constructed input template $I$ we extracted the learned templates \{${O_j}_1^n$\} from each knowledge-base. Since DIRT has a long tail of templates with a low score and very low precision, DIRT templates whose score are below a threshold of 0.1\(^2\) were filtered out. We then sampled 10\% of the templates in each output list, limiting the sample size to be between 5-20 templates for each list (thus obtaining sufficient evaluation data while controlling judgment load). For each sampled template $O$ we evaluated both directional rules, ‘$I \Rightarrow O$’ and ‘$O \Rightarrow I$’. In total, we sampled 380 templates, inducing 760 directional rules out of which 754 rules were unique.

Last, we randomly extracted a sample of example sentences for each rule ‘$L \Rightarrow R$’ by utilizing a search engine over the first CD of Reuters RCV1. First, we retrieved all sentences containing all lexical terms within $L$. The retrieved sentences were parsed using Minipar, keeping only sentences that syntactically match $L$ (as explained in Section 4.2.1). A sample of 15 matching sentences was selected, or all matching sentences if less than 15 were found. Finally, an example for judgment was generated from each sampled sentence and its left and right phrases (see Section 4.2.1). We did not find sentences in our corpus for 108 rules, and thus we ended up with 646 unique rules that could be evaluated (with 8945 examples to be judged).

Evaluating the Test-Set

Two human judges, fluent English speakers, evaluated the examples. We randomly split the examples between the judges. 100 rules (1287 examples) were cross annotated for agreement measurement. The judges followed the procedure in Section 4.2.3 and

\(^{1}\)http://about.reuters.com/researchandstandards/corpus/

\(^{2}\)Following advice by Patrick Pantel, DIRT’s co-author.


<table>
<thead>
<tr>
<th>Rule</th>
<th>Sentence</th>
<th>Judge 1</th>
<th>Judge 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X \text{ sign } Y \Rightarrow$</td>
<td><em>Iraq and Turkey</em> sign agreement to increase trade cooperation</td>
<td>Entailment holds</td>
<td>Irrelevant context</td>
</tr>
<tr>
<td>$X \text{ set } Y$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$X \text{ worsen } Y \Rightarrow X \text{ slow } Y$</td>
<td><em>News of the strike</em> worsened <em>the situation</em></td>
<td>Irrelevant context</td>
<td>No entailment</td>
</tr>
<tr>
<td>$X \text{ get } Y \Rightarrow X \text{ want } Y$</td>
<td><em>He</em> will get <em>his parade</em> on Tuesday</td>
<td>Entailment holds</td>
<td>No entailment</td>
</tr>
</tbody>
</table>

Table 4.3: Examples for disagreement between the two judges.

The correctness of each rule was assessed based on both its upper and lower bound precision values (Section 4.2.4).

### 4.3.2 Methodology Evaluation Results

We assessed the instance-based methodology by measuring the agreement level between judges. The judges agreed on 75% of the 1287 shared examples, corresponding to a reasonable Kappa value of 0.64. Yet our evaluation target is to assess rules, and the Kappa values for the final correctness judgments of the shared rules were 0.74 and 0.68 for the lower and upper bound evaluations respectively. These Kappa scores are regarded as *substantial agreement* and are substantially higher than published agreement scores and those we managed to obtain using the rule-based approach. As expected, the agreement on rules is higher than on examples, since judges may disagree on a certain example but their judgements would still yield the same rule assessment.

Table 4.3 illustrates some disagreements that were still exhibited within the instance-based evaluation. The primary reason for disagreements was the difficulty to decide whether a context is relevant for a rule or not, resulting in some confusion between *Irrelevant context* and *No entailment*. This may explain the lower agreement for the
upper bound precision, for which examples judged as *Irrelevant context* are ignored, while for the lower bound both judgments are conflated and represent no entailment.

About 43% of all examples were judged as *Left not entailed*, reflecting a syntactic matching precision of 57%. The relatively low precision made us collect more examples than needed, since *Left not entailed* examples are ignored. Better matching capabilities will allow collecting and judging fewer examples, thus improving the efficiency of the evaluation process.

### 4.3.3 DIRT and TEASE Evaluation Results

<table>
<thead>
<tr>
<th></th>
<th>DIRT</th>
<th>TEASE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>Y</td>
</tr>
<tr>
<td>Rules:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upper Bound</td>
<td>30.5%</td>
<td>33.5</td>
</tr>
<tr>
<td>Lower Bound</td>
<td>18.6%</td>
<td>20.4</td>
</tr>
<tr>
<td>Templates:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upper Bound</td>
<td>44%</td>
<td>22.6</td>
</tr>
<tr>
<td>Lower Bound</td>
<td>27.3%</td>
<td>14.1</td>
</tr>
</tbody>
</table>

Table 4.4: Average Precision (P) and Yield (Y) at the rule and template levels.

We evaluated the quality of the entailment rules produced by TEASE and DIRT using two scores: (1) micro average **Precision**, the percentage of correct rules out of all learned rules, and (2) average **Yield**, the average number of correct rules learned for each input template $I$, as extrapolated for the sample. Since DIRT and TEASE do not identify rule directionality, we also measured these scores at the template level, where an output template $O$ is considered correct if at least one of the rules ‘$I \Rightarrow O$’ or ‘$O \Rightarrow I$’ is correct. The results are presented in Table 4.4.

Since applications typically apply rules in a specific direction, the Precision for rules reflects their expected performance better than the Precision for templates.
Obviously, future improvement in precision is needed for rule learning algorithms. Meanwhile, manual filtering of the learned rules can prove effective within limited domains, where our evaluation approach can be utilized for reliable filtering as well. The substantial yield obtained by these algorithms suggest that they are indeed likely to be valuable for recall increase in semantic applications. While requiring a higher rule precision threshold to assess rule correctness would increase Precision, it would also decrease Yield.

In addition, we measured whether $O$ is a paraphrase of $I$, i.e. whether both $I \implies O$ and $O \implies I$ are correct. Only 20-25% of all correct templates were assessed as paraphrases. This stresses the significance of evaluating directional rules rather than only paraphrases. Furthermore, it shows that to improve precision, acquisition algorithms must identify rule directionality.

About 28% of all Left entailed examples were evaluated as Irrelevant context, yielding the large difference in precision between the upper and lower precision bounds. This result shows that to get closer to the upper bound precision, learning algorithms and applications need to identify the relevant contexts in which a rule should be applied.

We also found that only about 15% of the correct templates were learned by both algorithms, which implies that the two algorithms largely complement each other in terms of coverage. One explanation may be that DIRT is focused on the domain of the local corpus used (news articles for the published DIRT knowledge-base), whereas TEASE learns from the Web, extracting rules from multiple domains. Since Precision is comparable it may be best to use both algorithms in tandem.

Last, we note that the instance-based quality assessment corresponds to the corpus from which the example sentences were retrieved. It is therefore best to evaluate the learned rules using a corpus of the same domain from which they were acquired, or the target application domain for which the rules will be applied.
4.4 Recap: Human-based vs. Application-based Evaluations

At the beginning of this chapter we pointed at two approaches for evaluating entailment rules. The first one promotes human evaluation of rules and the second one promotes application-based evaluation. In Sections 4.2 and 4.3 we addressed the first approach by introducing a novel instance-based evaluation methodology that is more reliable and well-defined than the prior state-of-the-art rule-based methodology.

However, there are two limitations to human-based evaluation. The prominent limitation is that while the overall rule quality is assessed by such evaluations, they cannot measure an important aspect of an entailment rule knowledgebase: its expected utilization within NLP applications. Typically, the resource impact for applications is assessed through two quantifiers: Recall and Precision. Recall measures how well a resource covers the phenomena it was designed to meet. In our case, a rule-base should provide a good coverage to the variability of predicate occurrences together with their possible argument realizations. The recall aspect is not addressed at all within the human-based evaluation approach.

Precision of a rule-base measures the overall accuracy of its rule applications. Human-based evaluations measure one kind of rule quality, the per-rule precision. This measure is different than the overall accuracy of rule applications for a whole rule-base. The difference lies in the fact that under human-based evaluations all rules are treated as equally important, as having the same impact. However, the LHS templates of some rules are more frequent than others, which means that a system will attempt to apply them more often than other rules. Thus, the quality of frequent rules is of more importance to NLP systems than the quality of infrequent rules. For example, ‘\(X \text{ buy } Y \Rightarrow X \text{ acquire } Y\)’ will be applied more than ‘\(X \text{ repurchase } Y \Rightarrow X \text{ acquire } Y\)’, as ‘\(X \text{ buy } Y\)’ is much more frequent than ‘\(X \text{ repurchase } Y\)’. Thus
the impact of the correctness of the first rule on overall system performance is much larger than that of the second rule.

In addition, under human-based evaluation a rule is evaluated as correct if it is correct under some context. However, if that context is infrequent compared to other invalid contexts of the LHS rule, most rule applications in practice might end up being incorrect. In the instance-based methodology we proposed, a lower and upper bounds are drawn by either taking invalid context into account or not, but a final decision is drawn for all of the rule’s application. An application-based evaluation on the other hand, would evaluate each rule application on its own, thus providing a more fine-grained estimation of the rule’s quality, lying between the two bounds.

A second limitation of human-based evaluation is of a more technical nature. It is not trivial, and rather time consuming, to train judges for such evaluation procedures, and it takes time to evaluate a sample of several hundred rules. Furthermore, such annotation needs to be repeated each time the set of rules is changed. Under a development process, in which new methods are designed, the resulting rule set is often changed. Thus, the whole research cycle slows down, hindering entailment rule research quite dramatically. This effect was quite noticeable in entailment rule research in the last several years.

Following the two limitations of human-based evaluation, we are intrigued by the feasibility of the application-based evaluation schemes. As discussed in Section 4.1, the main limitations of such approaches are drawn from relying on full blown systems. Such systems are hard to come by with, their results are often not replicable and the analysis of the rule impact, detached from other system components, is difficult.

To overcome the above limitations, we next propose an application-based evaluation that utilizes a simple application setup, which is easy to implement and is easily replicable. Though the overall inference performance obtained within our proposed experimental setting does not compete with full blown systems, we think it provides
a very useful platform to compare between various entailment rule resources.

4.5 Application-based Evaluation Methodology

We aim to evaluate learned rule-bases by their utility for NLP applications through assessing the validity of inferences that are performed in practice using the rule-base. A testset for evaluating entailment rules requires: (a) a sample of test hypotheses, and (b) a corresponding corpus that contains sentences that entail these hypotheses, where all the entailing texts in the corpus are annotated as such. The task of the rule-base is to help proving the test hypotheses from texts.

Since we evaluate predicative entailment rules, our tested hypotheses should represent predicates. Therefore, as a first step target predicates need to be selected for testing, e.g. marry and arrest. Then, test hypotheses for each target predicate should be constructed. To isolate the affect of the rule-set on entailment inference, the test hypotheses should simple, so that the main contributor to correct and incorrect inferences would be the rule-set. A natural choice for simple hypotheses is to construct them from the predicate and the predicate arguments of interest. Hypotheses may be specific predicate mentions, e.g. “The police arrested a criminal”. However, it is cumbersome to both identify the different entailing texts of a specific predicate in the corpus and to generate corresponding mention hypotheses.

Instead of generating predicate mentions as hypotheses, it is much simpler to utilize template hypotheses, e.g. ‘X arrest Y’ or ‘X marry’. A template hypothesis corresponds to any correct instantiation of that template with variable instantiations taken from an entailing mention (see also the definition in Section 2.1). When using template hypotheses the tested rule-set’s task shifts from proving a hypothesis from texts to extracting correct variable instantiations for a template hypothesis from texts.
Accordingly, the corpus annotation should map between correct argument instantiations and their corresponding hypothesis variable. For example, the annotation of “John married Jane on Tuesday” should contain the mapping of both John and Jane to the $X$ variable in the hypothesis ‘$X$ marry’, marking them as two different correct instantiations of $X$ in this entailing text example.

The evaluation procedure assesses the correctness of all argument extractions, which are obtained by matching in the corpus either the seed templates or templates that entail them according to the tested rule-base, where the latter corresponds to rule-application. The annotated corpus allows immediate comparison of different rule sets and matching methods, without requiring any additional (post-hoc) annotation. We note that in IE terminology, template hypotheses are known as seed templates, and we will use both terminologies interchangeably.

Providing seed templates is simple, and designing matching capabilities for rule application is also not difficult. Annotating a corpus with all entailing mentions, however, is a much harder time consuming task. Luckily, we found that the available event mention annotations in the Automatic Content Extraction Evaluation 2005 event training set (ACE)\footnote{http://projects.ldc.upenn.edu/ace/} provides a useful test set that meets these generic criteria, with the added value of a standard real-world dataset.

The ACE event detection dataset is a standard IE benchmark focusing on event extraction. Its annotation includes 33 types of event predicates, such as Injure, Sue and Divorce, for which all event mentions are annotated in a corpus collected from various sources (newswire articles, blogs, etc.). The annotation of each event mention includes the instantiated arguments for its predicates, which represent the participants in the event, as well as general attributes such as time and place. The ACE guidelines specify for each event type its possible arguments, where all arguments are optional. Each argument is associated with a semantic role and a list of possible
named-entity types. For instance, an Injure event may have the arguments \{Agent, Victim, Instrument, Time, Place\}, where Victim should be of the type person.

To utilize the ACE dataset for evaluating entailment rule applications, each ACE event predicate is manually represented by a set of seed templates. For example, the seed templates for Injure may include ‘A injure V’ and ‘injure V in P’. In addition, each event argument is mapped to a corresponding seed template variable, e.g. Agent to A, Victim to V and Place to P in the above example. As mentioned above, argument mentions are found by matching either the seed templates or templates entailing them in some rules.

A rule application is considered correct if the matched variables are annotated by the corresponding ACE argument. That is, the instantiation of the variables, which are extracted and labeled according to the ACE arguments that are mapped to them, are also annotated as such by the gold standard annotation. As an example, lets look at rule applications of ‘treat V ⇒ injure V’, whose variable V is mapped to the Victim argument of the Injure event. The rule correctly extracted Lynch from “Lynch will be treated for bullet wounds” as the Victim argument, since the ACE gold-standard annotated Lynch as a Victim argument in the sentence. The same rule incorrectly extracted police officers from “Several police officers were treated for the effects of gas after the ensuing clashes”, with respect to the gold-standard. This is because the gold standard annotation did not mark police officers as a Victim argument of Injure in the sentence. As another example, the rule ‘V suffer ⇒ injure V’ incorrectly extracted Millennium Democratic Party as a Victim argument from “The ruling Millennium Democratic Party has suffered declining popularity”. Here, the rule was applied under invalid context, which was obviously not annotated by the ACE gold standard (the meaning of suffer in the sentence is metaphorical and not physical).

We note that in all our experiments under this evaluation scheme, templates
are matched using a syntactic matcher that handles simple syntactic variations such as passive-form and conjunctions (a detailed description of the matcher is given in Chapter 5). For example, ‘wound V in P ⇒ injure V in P’ was matched in “Hagel was wounded in Vietnam”.

We performed two adaptations to the ACE dataset to better fit it to our evaluation needs. First, our evaluation aims at assessing the correctness of inferring a specific target semantic meaning, which is denoted by a specific predicate, using rules. Thus, four events that correspond ambiguously to multiple distinct predicates were ignored. For instance, the Transfer-Money event refers to both donating and lending money, and thus annotations of this event cannot be mapped to a specific seed template representing a single predicate. We also omitted 3 events with less than 10 mentions, and were left with 26 events\(^1\), with 6380 argument mentions.

As a second adaptation, we regard all mentions that entail an event under the textual entailment definition as correct, even if the event is just implicitly implied by the predicate in the text. However, event mentions are annotated as correct in ACE only if they explicitly describe the target event. For instance, a Divorce mention does entail (through implication) a preceding marriage event but it does not explicitly describe it, and thus it is not annotated as a Marry event in ACE. To better utilize the ACE dataset, we considered for a target event the annotations of other events that entail it as being correct as well. For example, we considered the Divorce event mentions as entailing the Marry event. Thus, rule ‘X is divorced ⇒ X marry’ correctly extracted parents from “My parents are also divorced” as the Person argument of a Marry event mention, although the ACE gold-standard annotates this sentence only as a Divorce event and not as a Marry event. We emphasize that each

\(^1\)The remaining test events are: Appeal, Arrest, Attack, Be-Born, Charge-Indict, Convict, Declare-Bankruptcy, Demonstrate, Die, Divorce, Elect, End-Org, End-Position, Execute, Fine, Injure, Marry, Meet, Merge-Org, Nominate, Release-Parole, Sentence, Start-Org, Start-Position, Sue, Trial-Hearing.
event argument was considered separately in these cases. For example, we considered a mention of a divorced person as entailing the marriage of that person, as in the above example, but did not consider the place and time of the divorce act to be entailing those of the marriage event. For example, 1992 in “Nathan divorced in 1992” is not considered a Time argument of a Marry event mention, even though it is annotated as a Time argument of a Divorce event mention. On the other hand, Nathan in the above example is considered a Person argument for both a Divorce and a Marry events.

Finally, we note that the performance of our experimental setting with particular rule-bases as an “IE system” on the ACE task as such is limited. First, most rule-bases do not provide all types of needed rules. To achieve better coverage, several rule-bases need to be utilized in tandem. Second, the experimental system that applies the rules is rather basic, with limited matching capabilities. For example, no discourse analysis is performed. Third, only one rule may be applied for each argument extraction. No rule chaining is performed to increase the coverage of the given rule-set for the ACE events. However, even with these limited capabilities, this setup is very useful for relative comparison of various entailment rule resources, since the relative impact of the rules on real-world data can be rather easily assessed independently from other system components.

4.6 Conclusions

Accurate learning of inference knowledge, such as entailment rules, has become critical for further progress of applied semantic systems. However, evaluation of such knowledge has been problematic, hindering further developments. In this chapter we proposed two approaches that improve current evaluation procedures. The manual instance-based evaluation approach obtained acceptable agreement levels, which are
substantially higher than those obtained for the common rule-based approach. The ACE-based evaluation approach proposes a simple yet effective framework for assessing the impact of rule-bases on real NLP applications. In addition, when using the ACE-based setting, there is no need for post-hoc manual annotation whenever the algorithm that generates the rules is changed.

In this chapter we also conducted the first comparison between two state-of-the-art acquisition algorithms, DIRT and TEASE, using our proposed instance-based methodology. We found that their precision is comparable but they effectively complement each other in terms of rule coverage. In addition, we found that most learned rules are not paraphrases but rather one-directional entailment rules, and that many of the rules are context sensitive. These findings guided us in choosing our research topics, in particular the learning of rule directionality and identifying relevant contexts, issues that were hardly explored till now. Yet, before proposing novel rule acquisition and application approaches, in the next chapter we further investigate a major aspect of the representation of entailment rules.
Chapter 5

Canonization of Entailment Rules

5.1 Introduction

In Section 2.3 we presented state-of-the-art algorithms for automatic acquisition of entailment rules. These algorithms’ strength is in learning relations between lexical-syntactic templates, which capture lexical-based language knowledge and world knowledge. One noticeable phenomenon of lexical-syntactic templates is that they can take on many morpho-syntactic variations, which (largely) represent the same predicate and are semantically equivalent. For example, ‘X compose Y’ can be expressed also by ‘Y is composed by X’ or ‘X’s composition of Y’. This phenomenon is addressed at inference time by recognizing semantically equivalent syntactic variations, such as passive forms and conjunctions (e.g. (Romano et al., 2006; Bar-Haim et al., 2008)); some work was done to systematically recognize morphological variations of predicates (MacLeod et al., 1998; Gurevich et al., 2006).

In contrast, previous rule learning algorithms completely ignore this morpho-syntactic variability at learning time. They treat these variations as semantically different, thus learning rules for each variation separately. This leads to several undesired consequences. First, statistics for a particular semantic predicate are scattered
among different templates. This may result in insufficient statistics for learning a rule in any of its variations. For example, if for each of the rule variations ‘\( X \) acquire \( Y \) \( \Rightarrow \) \( X \) own \( Y \)’, ‘\( Y \) is acquired by \( X \) \( \Rightarrow \) \( X \) own \( Y \)’ and ‘\( X \)’s acquisition of \( Y \) \( \Rightarrow \) \( X \) own \( Y \)’ there are no sufficient statistics then none of them will be learned, even though accumulated statistics for all variations might warrant learning this rule. Second, though rules may be learned in several variations (see Table 5.1), in most cases only a small part of the possible morpho-syntactic variations are learned. Thus, an inference system that uses only these learned rules would miss recognizing a substantial number of variations of the sought predicate.

It therefore makes more sense to design a modular architecture, where a separate entailment module recognizes entailing variations that are based on generic morphological and syntactic regularities (morpho-syntactic entailments). We propose to use such a module first at learning time, by learning only canonical forms of templates and rules. Then, applying the module also at inference time, in conjunction with the learned lexical-based canonical rules, guarantees the coverage of all morpho-syntactic variations of a given canonical rule.

### 5.2 A Modular Approach for Rule Learning

In this section we detail our proposed architecture for addressing morpho-syntactic variability in templates. In our scheme, we use a generic morpho-syntactic entailment
module to transform lexical-syntactic template variations that occur in a text into their canonical form. This form, which we chose to be the active verb form with direct modifiers, is entailed by other template variations. We next describe our implementation of such a module and its application within entailment rule acquisition algorithms.

5.2.1 Morpho-Syntactic Canonization Module

We implemented a morpho-syntactic module based on a set of canonization rules, highly accurate morpho-syntactic entailment rules. Each rule represents one morpho-syntactic regularity that is eliminated when the rule is applied to a given template (see examples in Table 5.2 and Figure 5.1).

Our current canonization rule collection consists of two types of rules: (a) syntactic-based rules; (b) morpho-syntactic nominalization rules. We next describe each rule type. As we use the Minipar parser, all rules are adapted to Minipar’s output format.

**Syntactic-based Rules**  These rules capture entailment patterns associated with common syntactic structures. Their function is to simplify and generalize the syntactic structure of a template.

In the current implementation we manually created the following simplification rules: (a) passive forms into active forms; (b) removal of conjunctions; (c) removal of appositions; (d) removal of abbreviations; (e) removal of set description by the ‘such as’ preposition. Table 5.2 presents some of the rules we created together with examples of their effect.

**Nominalization Rules**  Entailment rules such as ‘acquisition of Y by X ⇒ X acquire Y’ and ‘Y’s acquisition by X ⇒ X acquire Y’ capture the relations between verbs and their nominalizations. We automatically derived these rules from Nomlex
CHAPTER 5. CANONIZATION OF ENTAILMENT RULES

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
<th>Original Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>passive to active</td>
<td>pcomp−n by−subj subj</td>
<td>X by V [→] X V</td>
</tr>
<tr>
<td>conjunction</td>
<td>conj</td>
<td>Z [→] Y [→] Y</td>
</tr>
<tr>
<td>apposition</td>
<td>appo</td>
<td>Z [→] Y [→] Y</td>
</tr>
<tr>
<td>abbreviation</td>
<td>spellout</td>
<td>Z [→] Y [→] Y</td>
</tr>
</tbody>
</table>

Table 5.2: Some of the syntactic rules used in our implementation, together with usage examples for transforming complex template into the simpler template ‘X subj find obj Y’ (the applications of the second rule and the third rule are also demonstrated in Figure 5.1).

(Macleod et al., 1998), as described in (Ron, 2006). These rules transform any nominal template in Nomlex into its related verbal form. These rules preserve the semantics of the original template predicate. We chose the verbal form as the canonical form since for every predicate with specific semantic modifiers there is only one verbal active form in Nomlex, but typically several equivalent nominal forms (as in the acquisition example above).

Chaining of Canonization Rules Each of the syntactic rules decreases the size of a template. In addition, nominalization rules can only be applied once for a given template, since no rule in our rule-set transforms a verbal template into one of its nominal forms. Thus, applying rules until no rule is applicable is a finite process. In addition, each of our rules is independent of the others, operating on a different set of dependency relations. Consequently, applying any sequence of rules until no rule is applicable will result in the same final canonical template form. Figure 5.1 illustrates an example for rule chaining.
5.2. A MODULAR APPROACH FOR RULE LEARNING

Figure 5.1: Chaining of canonization rules that transforms the path template between the arguments \( \{X = \text{Google}; Y = \text{Sprinks}\} \), which occurs in the sentence “We witnessed the acquisition of Kaltix and Sprinks by another growing company, Google”, into a canonized template form. The first rule applied is a nominalization rule, followed by removal of apposition and removal of conjunction (as described in Table 5.2). As can be seen, applying the rules in any order will result in the same final canonized form.

5.2.2 Applying the Canonization Module

When a morpho-syntactic entailment module is utilized at inference time (e.g. (Romano et al., 2006)), it recognizes a closure of morpho-syntactic variations for a lexical-syntactic template. Accordingly, acquisition algorithms may learn just a single morpho-syntactic variation of a template.

With this modular scheme in mind, we propose to solve the learning problems discussed in Section 5.1 by utilizing the morpho-syntactic entailment module at learning time as well. We incorporate the module in learning algorithms (TEASE and DIRT in our experiment) by converting each template variation occurrence in the learning corpus into an occurrence of a canonical template. Thus, the learning algorithms operate only on canonical forms.

When canonization is used, no morpho-syntactically redundant rules are learned, with respect to the variations that are recognized by the module. This makes the output more compact, both for storage and for use. In addition, the statistics of the different morpho-syntactic variations is now accumulated for the canonical form. The
improved statistics may result, for example, in learning more rules that could not be learned before in any single variation.

Methodologically, previous evaluations of learning algorithms reported accuracy relative to the redundant list of rules, which creates a bias for templates with many frequent variations. When this bias is removed and only truly different lexical-syntactic rules are assessed, evaluation is more efficient and accurate.

5.3 Evaluation

We conducted two experiments: (a) a manual evaluation of the contribution of the canonization module to TEASE and DIRT, based on human judgment of the learned rules; (b) a Relation Extraction evaluation setup for a protein interaction data-set.

5.3.1 Human Judgement Evaluation

We selected 20 different verbs and verbal phrases\(^1\) as input templates for the two state-of-the-art algorithms, TEASE and DIRT. We implemented both the baseline versions of the algorithms (without canonization), marked as \(TEASE_b\) and \(DIRT_b\), and the versions with the canonization module, marked as \(TEASE_c\) and \(DIRT_c\). The results of the running these algorithms constitute our test-set rules.

Both TEASE and DIRT do not learn the direction(s) of an entailment relation between an input template \(I\) and a learned output template \(O\). Thus, we evaluated both candidate directional rules, \(I \Rightarrow O\) and \(O \Rightarrow I\), similar to the evaluation in Section 4.3.3. The learned rules from each tested algorithm are evaluated by the instance-based evaluation approach (presented in Section 4.2).

\(^1\)The verbs are: accuse, approve, calculate, change, demand, establish, finish, hit, invent, kill, know, leave, merge with, name as, quote, recover, reflect, tell, worsen, write.
5.3.2 TEASE Evaluation

We separated the templates that were learned by \( TEASE_c \) into two lists: (a) a **baseline-templates** list containing templates also learned by \( TEASE_b \); (b) a **new-templates** list containing templates that were not learned by \( TEASE_b \), but learned by \( TEASE_c \) thanks to the improved statistics. In total, 3871 templates were learned: 3309 in the baseline-templates list and 562 in the new-templates list. Inherently, every output template learned by \( TEASE_b \) is also learned in its canonical form by \( TEASE_c \), since its supporting statistics may only increase.

We randomly sampled 100 templates from each list and evaluated their correctness. 10 example sentences were retrieved for each rule from the first CD of Reuters RCV1. Two judges evaluated the examples. The rules were randomly split between the judges with 100 rules (942 examples) cross annotated for agreement measurement.

**Results**

First, we measured the redundancy in the rules learned by \( TEASE_b \) to be 6.2% per input template on average. We considered only morpho-syntactic phenomena that are addressed in our implementation. This redundancy was eliminated using the canonization module.

Next, we evaluated the quality of each rule sampled using two scores: (1) micro average **Precision**, the percentage of correct templates out of all learned templates, and (2) average **Yield**, the average number of correct templates learned for each input template, as extrapolated for the sample. The results are presented in Table 5.3. The agreement between the judges was measured by the Kappa value (Cohen, 1960), which is 0.67 on the relevant examples (corresponding to substantial agreement).

We expect \( TEASE_c \) to learn new rules using the canonization module. In our experiment, 5.8 more correct templates were learned on average per input template by
Table 5.3: Average Precision and Yield of the output lists.

<table>
<thead>
<tr>
<th>Template List</th>
<th>Avg. Precision</th>
<th>Avg. Yield</th>
</tr>
</thead>
<tbody>
<tr>
<td>$TEASE_b$</td>
<td>30.1%</td>
<td>49.8</td>
</tr>
<tr>
<td>$TEASE_c$</td>
<td>28.7%</td>
<td>55.6</td>
</tr>
<tr>
<td>$DIRT_b$</td>
<td>24.7%</td>
<td>46.9</td>
</tr>
<tr>
<td>$DIRT_c$</td>
<td>24.9%</td>
<td>47.5</td>
</tr>
</tbody>
</table>

Table 5.4: Examples for correct templates that TEASE learned only after using canonization rules.

<table>
<thead>
<tr>
<th>Input Template</th>
<th>Learned Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$ accuse $Y$</td>
<td>$X$ blame $Y$</td>
</tr>
<tr>
<td>$X$ approve $Y$</td>
<td>$X$ take action on $Y$</td>
</tr>
<tr>
<td>$X$ demand $Y$</td>
<td>$X$ call for $Y$ , $X$ in demand for $Y$</td>
</tr>
<tr>
<td>$X$ establish $Y$</td>
<td>$X$ open $Y$</td>
</tr>
<tr>
<td>$X$ hit $Y$</td>
<td>$X$ slap $Y$</td>
</tr>
<tr>
<td>$X$ invent $Y$</td>
<td>grant $X$ patent on $Y$ , $X$ is co-inventor of $Y$</td>
</tr>
<tr>
<td>$X$ kill $Y$</td>
<td>$X$ hang $Y$ , charge $X$ in death of $Y$</td>
</tr>
<tr>
<td>$X$ named as $Y$</td>
<td>hire $X$ as $Y$ , select $X$ as $Y$</td>
</tr>
<tr>
<td>$X$ quote $Y$</td>
<td>$X$ cite $Y$</td>
</tr>
<tr>
<td>$X$ tell $Y$</td>
<td>$X$ persuade $Y$ , $X$ say to $Y$</td>
</tr>
<tr>
<td>$X$ worsen $Y$</td>
<td>$X$ impair $Y$</td>
</tr>
</tbody>
</table>

$TEASE_c$. This corresponds to an increase of 11.6% in average Yield (see Table 5.3). Examples of new correctly learned templates are shown in Table 5.4.

There is a slight decrease in precision when using $TEASE_c$. One possible reason is that the new templates are usually learned from very few occurrences of different variations, accumulated for the canonical templates. Thus, they may have a somewhat lower precision in general. Overall, the significant increase in Yield is much more important, especially if the learned rules are later filtered manually.
5.3. EVALUATION

5.3.3 DIRT Evaluation

Unlike TEASE, DIRT has a very long noisy tail of candidate templates. However, DIRT poses no hard threshold for filtering out this long tail. Instead, we follow (Lin and Pantel, 2001), who evaluated only the top-$N$ templates learned for each input template. (Lin and Pantel, 2001) set $N$ to be 40, but this choice seems quite arbitrary. We set $N$ to be 190 to assess an output list that is similar in size to TEASE’s output. Before selecting the top 190 templates, we removed redundant templates from $DIRT_b$, those that are just morpho-syntactic variations of a template with a higher score. We converted the remaining templates to their canonical forms.

We separated the templates learned for each input template into three lists: (a) a common-templates list containing templates that appear in both $DIRT_b$ and $DIRT_c$ top-190 lists; (b) a new-templates list containing templates that appear only in the $DIRT_c$ list; (c) an old-templates list containing templates that appear only in the $DIRT_b$ list. Out of the 3800 templates selected from each DIRT version output, 3353 were in the common-list and 447 were in each of the new/old lists.

We sampled 100 templates from each list and evaluated their correctness (10 sentences for each rule). One judge evaluated the sample. The evaluation results were affirmed by an additional evaluation by me.

Results

We measured the redundancy in the rules learned by $DIRT_b$ to be 5.6% per input template on average. This redundancy was removed using the canonization module. We found that only about 13% of the learned templates were learned by both TEASE and DIRT. This result is similar to the one achieved in Section 4.3.3, reaffirming that the algorithms do not compete but rather largely complement each other in terms of Yield, since they learn from different resources.
13.3% of the top-190 templates learned by $DIRT_b$ were replaced by other templates in $DIRT_c$, as the change in statistics results in different template ranking. We measured Precision and Yield as in Section 5.3.2. The results are presented in Table 5.3.

As can be seen, the performance of $DIRT_c$ is basically comparable to that of $DIRT_b$. It seems that in acquisition algorithms like TEASE, which use complex and more informative features that are infrequent, adding more statistics results in higher quality learning. On the other hand, DIRT is based on frequent simple features that are less informative. Under this approach, adding some more statistics does not seem to dramatically change the overall score of a rule. Perhaps a more substantial increase in the statistics, such as by adding more canonization rules, will result in a positive change.

Overall, it is useful to incorporate canonization also in DIRT in order to remove the redundancy within the learned rules but also to enable a uniform architecture for applying rules learned by different algorithms.

### 5.3.4 Relation Extraction Evaluation

To illustrate the potential contribution of the increased number of learned rules we conducted a small-scale experiment in a Relation Extraction (RE) setup over a data-set of protein interactions (Bunescu et al., 2005). The task is to identify pairs of proteins that are described in a text as interacting.

We have set a simple partial replication of the RE configuration presented in (Romano et al., 2006). We used ‘$X$ interact with $Y$’ as the only input template for both $TEASE_b$ and $TEASE_c$, which learned entailment rules containing this template from the Web. We then extracted protein pairs using the rules learned. For canonization at inference time, we used only the rules described in Section 5.2.1 (a wider range of matching techniques should be used in order to reach higher recall).
5.3. EVALUATION

Table 5.5: Results for the protein interaction setup using TEASE with and without canonization.

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>$TEASE_b$</td>
<td>9.4%</td>
<td>83%</td>
</tr>
<tr>
<td>$TEASE_c$</td>
<td>10.4%</td>
<td>87.5%</td>
</tr>
</tbody>
</table>

Table 5.5 presents the results of our two TEASE versions for a test set of about 600 mentions of interacting pairs. There is a relative improvement of about 10% in recall, which reflects the yield increase in $TEASE_c$. These results are preliminary and of small scale, but they illustrate the potential benefit of learning with canonization.

We note that TEASE precision in this experiment, which was measured over actual applications of the learned rules in the test set, is much higher than that of Section 5.3.2, where the percentage of correctly learned rules was measured. This shows that many incorrectly learned rules are not applicable in typical contexts and thus rarely deteriorate overall performance.

5.3.5 Analysis

Parser errors are one of the main reasons that variations are sometimes not transformed into their canonical form. These errors result in different parse trees for the same syntactic constructs. Thus, several parser dependent rules may be needed to capture the same phenomenon. Moreover, it is difficult to design canonization rules for some parsing errors, since the resulting parse trees consist of structures that are common to other irrelevant templates. For example, when Minipar chooses the head of the conjunct $Y$ in “The interaction between $X$ and $Y$ will not hold for long” to be $interaction$ and not $X$, the appropriate nominalization rule cannot be applied. These errors affect both the learning phase, where statistics are not accumulated to the appropriate canonical form, and the inference phase, where a variations of a canonical
rule are not recognized.

Finally, we note that the reported results correspond only to the phenomena captured by our currently implemented canonization rules. Adding more rules that cover more morpho-syntactic phenomena is expected to increase the performance obtained by our canonization scheme. For example, there are many nominalizations that are not specified in the Nomlex version we utilized, but can be found in other resources, such as WordNet (Miller, 1995).

5.4 Conclusions

We proposed a modular approach for addressing morpho-syntactic variations of templates when learning entailment rules, based on rule canonization. We then used it for template canonization in two state-of-the-art acquisition algorithms. Our experiments showed that redundancy is removed while new correct rules are learned. We also showed initial improvement in a Relation Extraction setting when using the additional rules learned with the canonization module. Finally, we suggest that the evaluation of rules in a canonical form is more accurate, since the bias for templates with many frequent variations is removed.

Future work may investigate other types of entailment knowledge that can contribute to canonization, such as synonyms, as well as add more syntactic and morpho-syntactic rules, which were not covered yet.

Handling morpho-syntactic redundancy addresses one current limitation of state-of-the-art rule representation at the syntactic level. We next address further representation and learning aspects of predicative entailment rules: representing each predicate argument separately and learning also the direction of entailment rules.
Chapter 6

Learning Unary Entailment Rules

6.1 Introduction

Most previous unsupervised rule learning algorithms focus on learning binary entailment rules, ignoring rules between unary templates. However, using binary rules for inference is not sufficient. First, a predicate that can take multiple arguments may still occur with only one of its arguments in certain constructions. For example, in “The acquisition of TCA was successful”, TCA is the only argument of acquisition. Second, some predicates are unary by nature. For example, modifiers, such as ‘the elected X’, or intransitive verbs, such as sleep. In addition, it appears more tractable to learn all variations for each argument of a predicate separately than to learn them for combinations of argument pairs.

For these reasons, it seems that unary rule learning should be addressed in addition to binary rule learning. We are further motivated by the fact that some (mostly supervised) works in IE found learning unary templates useful for recognizing relevant named entities (Riloff, 1996; Sudo, Sekine, and Grishman, 2003; Shinyama and Sekine, 2006), though they did not attempt to learn generic knowledge bases of entailment rules. As far as we know, only (Shinyama et al., 2002) and (Pekar, 2006)
learn rules between unary templates. However, the algorithm in (Shinyama et al., 2002) relies on comparable corpora for identifying paraphrases and simply takes any two templates from comparable sentences that share a named entity instantiation to be paraphrases. Such approach is not feasible for non-comparable corpora where statistical measurement is required. Pekar (2006) learns rules only between templates related by local discourse (information from different documents is ignored). In addition, the template structure in (Pekar, 2006) is limited only to verbs and their direct syntactic arguments, which may yield incorrect rules, e.g. for light verb constructions (see Section 6.4.3).

In this chapter we focus on unsupervised learning of unary entailment rules from general corpora. We first present background information on distributional similarity measures, the typical approach to learning rules from general corpora. We then propose two learning approaches. In our main approach, rules are learned by measuring the similarity of the variable instantiations of two templates in a corpus. In addition to adapting state-of-the-art similarity measures for unary rule learning, we propose a new measure, termed Balanced-Inclusion, which balances the notion of directionality in entailment with the common notion of symmetric semantic similarity. In a second approach, unary rules are derived from binary rules learned by state-of-the-art binary rule learning methods.

We also discuss the limitation of binary dependency paths for template structure representation. We present a simple yet more expressive template structure for unary templates in order to overcome this limitation.

6.2 Background

This section reviews relevant distributional similarity measures, both symmetric and directional, which were applied for either lexical similarity or unsupervised entailment
6.2. BACKGROUND

Distributional similarity measures follow the Distributional Hypothesis, which states that words that occur in the same contexts tend to have similar meanings (Harris, 1954). Various measures were proposed in the literature for assessing such similarity between two words, $u$ and $v$. Given a word $q$, its set of features $F_q$ and feature weights $w_q(f)$ for $f \in F_q$, a common symmetric similarity measure is Lin similarity (Lin, 1998a):

$$
Lin(u, v) = \frac{\sum_{f \in F_u \cap F_v} [w_u(f) + w_v(f)]}{\sum_{f \in F_u} w_u(f) + \sum_{f \in F_v} w_v(f)}
$$

where the weight of each feature is the PMI between the word and the feature:

$$
w_q(f) = \log \left[ \frac{Pr(f|q)}{Pr(f)} \right].
$$

Weeds and Weir (2003) proposed to measure the symmetric similarity between two words by averaging two directional (asymmetric) scores: the coverage of each word’s features by the other. The coverage of $u$ by $v$ is measured by:

$$
Cover(u, v) = \frac{\sum_{f \in F_u \cap F_v} w_u(f)}{\sum_{f \in F_u} w_u(f)}
$$

The average can be arithmetic or harmonic:

$$
WeedsA(u, v) = \frac{1}{2} [Cover(u, v) + Cover(v, u)]
$$

$$
WeedsH(u, v) = \frac{2 \cdot Cover(u, v) \cdot Cover(v, u)}{Cover(u, v) + Cover(v, u)}
$$

Weeds et al. also used PMI for feature weights.

Binary rule learning algorithms adapted such lexical similarity approaches for learning rules between templates, where the features of each template are its variable instantiations. One such example is the DIRT algorithm, which learns non-directional
binary rules for templates based on the Lin similarity measure (see also Section 2.3.2 for detailed description). DIRT’s template structure is a path in a dependency parse-tree between two noun variables.

**Directional Measures** Most rule learning methods apply a symmetric similarity measure between two templates, viewing them as paraphrasing each other. However, as already discussed before, entailment is in general a directional relation, e.g. ‘$X$ acquire $Y \Rightarrow X$ own $Y$’ and ‘$countersuit against X \Rightarrow lawsuit against X$’.

Some works address rule directionality. (Weeds and Weir, 2003) propose a directional measure for learning hyponymy between two words, ‘$l \Rightarrow r$’, by giving more weight to the coverage of the features of $l$ by $r$ (with $\alpha > \frac{1}{2}$):

$$WeedsD(l, r) = \alpha \text{Cover}(l, r) + (1 - \alpha) \text{Cover}(r, l)$$

When $\alpha = 1$, this measure degenerates into $\text{Cover}(l, r)$, termed also $\text{Precision}(l, r)$. With $\text{Precision}(l, r)$ we obtain a “soft” version of the inclusion hypothesis presented in (Geffet and Dagan, 2005), which expects $l$ to entail $r$ if the “important” features of $l$ appear also in $r$.

Similarly, the LEDIR algorithm (Bhagat, Pantel, and Hovy, 2007) identifies the entailment direction between two binary templates, $l$ and $r$, which participate in a relation learned by (the symmetric) DIRT, by measuring the proportion of instantiations of $l$ that are covered by the instantiations of $r$.

### 6.3 Learning Unary Entailment Rules

This section investigates acquisition of unary entailment rules from general non-comparable corpora. We first describe the structure of unary templates and then
explore two conceivable approaches for learning unary rules. The first approach directly assesses the relation between two given templates based on the similarity of their instantiations in the corpus. The second approach, which was also mentioned in (Iftene and Balahur-Dobrescu, 2007), derives unary rules from learned binary rules.

### 6.3.1 Unary Template Structure

To learn unary entailment rules we first need to define their structure. As stated before, in this thesis we work at the syntactic representation level: texts are represented by dependency parse trees (using the Minipar parser) and templates by parse sub-trees.

Given a dependency parse tree, any sub-tree can be a candidate template. However, the number of possible templates is exponential in the size of the sentence. In the binary rule learning literature, the main solution for exhaustively learning all rules between any pair of templates in a given corpus is to restrict the structure of templates. Typically, a template is restricted to be a path in a parse tree between two variable nodes (Lin and Pantel, 2001; Ibrahim, Katz, and Lin, 2003) (see also Section 2.3.1).

Even when focusing on unary templates, the number of candidate sub-trees is still exponential. Thus, we follow the main line of work and restrict our unary template structures. We are also interested in as simple structure as possible, aiming at an alternative to the simple tree path used for binary templates. Taking these considerations, we chose the structure of unary templates to be paths as well, where one end of the path is the template’s variable. However, paths with one variable have more expressive power than paths between two variables, since the combination of two unary paths may generate a binary template that is not a path. For example, the combination of ‘X call indictable’ and ‘call Y indictable’ is the template ‘X call Y indictable’, which is not a path between X and Y.
In this work, for every noun node $v$ in a parsed sentence, we generate templates with $v$ as a variable as follows:

1. Traverse the path from $v$ towards the root of the parse tree. Whenever a candidate predicate is encountered (any noun, adjective or verb) the path from that node to $v$ is taken as a template. We stop when the first verb or clause boundary (e.g. a relative clause) is encountered, which typically represent the syntactic boundary of a specific predicate.

2. To enable templates with control verbs and light verbs, e.g. ‘$X$ help preventing’, ‘$X$ make noise’, whenever a verb is encountered we also generate templates that are paths between $v$ and the verb’s modifiers, either objects, prepositional complements or infinite or gerund verb forms (paths ending at stop words, e.g. pronouns, are not generated).

3. To capture noun modifiers that act as predicates, e.g. ‘the losing $X$’, we extract template paths between $v$ and each of its modifiers, nouns or adjectives, that are derived from a verb. We use the Catvar database to identify verb derivations (Habash and Dorr, 2003).

As an example for the procedure, the templates extracted from the sentence “The losing party played it safe” with ‘party’ as the variable are: ‘losing $X$’, ‘$X$ play’ and ‘$X$ play safe’.

6.3.2 Direct Learning of Unary Rules

We applied the lexical similarity measures presented in Section 6.2 for unary rule learning. Each argument instantiation of template $t$ in the corpus is taken as a feature $f$, and the PMI between $t$ and $f$ is used for the feature’s weight. We first adapted DIRT for unary templates, termed here unary-DIRT, applying Lin-similarity to
6.3. LEARNING UNARY ENTAILMENT RULES

the single feature vector. We also adapted DIRT’s output filtering by LEDIR to
unary rules, as well as the various Weeds measures\(^1\): symmetric arithmetic average,
symmetric harmonic average, weighted arithmetic average and Precision.

After initial analysis, we found that given a right hand side template \(r\), symmetric
measures such as Lin (in DIRT) generally tend to prefer (score higher) relations \(\langle l, r \rangle\)
in which \(l\) and \(r\) are related but do not necessarily participate in an entailment or
equivalence relation, e.g. the incorrect rule ‘\(
\text{kill } X \Leftrightarrow \text{injure } X\)’.

On the other hand, directional measures such as Weeds Precision tend to pre-
ffer directional rules in which the entailing template is infrequent. If an infrequent
template has common instantiations with another template, the coverage of its fea-
tures is typically high, whether or not an entailment relation exists between the two
templates. This behavior generates high-score incorrect rules.

Based on this analysis, we propose a new measure that balances the two behaviors,
termed \textbf{Balanced-Inclusion (BInc)}. BInc identifies entailing templates based on
a directional measure but requires as a “prior” that the two templates would have
some semantic relatedness, captures by the symmetric measure:

\[
BInc(l, r) = \sqrt{Lin(l, r) \cdot Precision(l, r)}
\]

In this way, BInc penalizes infrequent templates using a symmetric measure.

6.3.3 Deriving Unary Rules From Binary Rules

An alternative way to learn unary rules is to first learn binary entailment rules and
then derive unary rules from them. We derive unary rules from a given binary rule-
base in two steps. First, for each binary rule, we generate all possible unary rules that

\(^1\)We applied the best performing parameter values presented in (Bhagat, Pantel, and Hovy, 2007)
and (Weeds and Weir, 2003).
are part of that rule (each unary template is extracted following the same procedure
described in Section 6.3.1). For example, from ‘X find solution to Y ⇒ X solve Y’ we
generate the unary rules ‘X find ⇒ X solve’, ‘X find solution ⇒ X solve’, ‘solution
to Y ⇒ solve Y’ and ‘find solution to Y ⇒ solve Y’. The score of each generated
rule is set to be the score of the original binary rule.

The same unary rule can be derived from different binary rules. For example,
‘hire Y ⇒ employ Y’ is derived both from ‘X hire Y ⇒ X employ Y’ and ‘hire Y for
Z ⇒ employ Y for Z’, having a different score from each original binary rule. The
second step of the algorithm aggregates the different scores yielded for each derived
rule to produce the final rule score. Three aggregation functions were tested: sum
(Derived-Sum), average (Derived-Avg) and maximum (Derived-Max).

6.4 Experiments

6.4.1 Experimental Setup

In our experiments we follow the methodology presented in Section 4.5, utilizing the
ACE 2005 event training set. We manually represented each ACE event predicate
by unary seed templates. Example seed templates for the event Injure are ‘A injure’, ‘injure V’ and ‘injure in T’. We mapped each event role annotation to the
corresponding seed template variable, e.g. ‘Agent’ to A and ‘Victim’ to V in the
above example. Templates are matched using a syntactic matcher that handles sim-
ple morpho-syntactic phenomena, as described in Chapter 5.

For testing binary rule-bases, we automatically generated binary seed templates
from any two unary seeds that share the same predicate. For example, for Injure
the binary seeds ‘A injure V’, ‘A injure in T’ and ‘injure V in T’ were automatically
generated from the above unary seeds.
6.4. EXPERIMENTS

We implemented the unary rule learning algorithms described in Section 6.3 and the binary DIRT algorithm. We executed each method over the Reuters RCV1 corpus, learning for each template \( r \) in the corpus the top 100 rules in which \( r \) is entailed by another template \( l \), \( l \Rightarrow r \). All rules were learned in their canonical form (see Chapter 5). The rule-base learned by binary DIRT was taken as the input for deriving unary rules from binary rules.

The performance of each acquired rule-base was measured for each ACE event. We measured the percentage of correct argument mentions extracted out of all correct argument mentions annotated for the event (recall) and out of all argument mentions extracted for the event (precision). We also measured F1, their harmonic average, and report macro average Recall, Precision and F1 over the 26 event types.

No threshold setting mechanism is suggested in the literature for the scores of the different algorithms, especially since rules for different right hand side templates have different score ranges. Thus, we follow common evaluation practice (Lin and Pantel, 2001; Geffet and Dagan, 2005) and test each learned rule-set by taking the top \( K \) rules for each seed template, where \( K \) ranges from 0 to 100. When \( K=0 \), no rules are used and mentions are extracted only by direct matching of seed templates.

As discussed in Section 4.5, our rule application setting provides a rather simplistic IE system (for example, no named entity recognition or approximate template matching). It is thus useful for comparing different rule-bases, though the absolute extraction figures do not reflect the full potential of the rules. In Secion 6.4.3 we analyze the full-system’s errors to isolate the rules’ contribution to overall system performance.

6.4.2 Results

In this section we focus on the best performing variations of each algorithm type: binary DIRT, unary DIRT, unary Weeds Harmonic, BInc and Derived-Avg. We omitted
the results of methods that were clearly inferior to others: (a) WeedsA, WeedsD and Weeds-Precision did not increase Recall over not using rules because rules with very infrequent templates scored highest and arithmetic averaging could not balance well these high scores; (b) out of the methods for deriving unary rules from binary rule-bases, Derived-Avg performed best; (c) filtering with (the directional) LEDIR did not improve the performance of unary DIRT.

Figure 6.1 presents Recall, Precision and F1 of the methods for different cutoff points. First, we observe that even when matching only the seed templates ($K=0$),

Figure 6.1: Average Precision, Recall and F1 at different top K rule cutoff points.
unary seeds outperform the binary seeds in terms of both Precision and Recall. This surprising behavior is consistent through all rule cutoff points: all unary learning algorithms perform better than binary DIRT in all parameters. The inferior behavior of binary DIRT is analyzed in Section 6.4.3.

The graphs show that symmetric unary approaches substantially increase recall, but dramatically decrease precision already at the top 10 rules. As a result, F1 only decreases for these methods. Lin similarity (DIRT) and Weeds-Harmonic show similar behaviors. They consistently outperform Derived-Avg. One reason for this is that incorrect unary rules may be derived even from correct binary rules. For example, from ‘\( X \) gain seat on \( Y \Rightarrow elect X to Y \)’ the incorrect unary rule ‘\( X \) gain \( \Rightarrow elect X \)’ is also generated. This problem is less frequent when unary rules are directly scored based on their corpus statistics.

The directional measure of BInc yields a more accurate rule-base, as can be seen by the much slower precision reduction rate compared to the other algorithms. As a result, it is the only algorithm that improves over the F1 baseline of \( K=0 \), with the best cutoff point at \( K=20 \). BInc’s recall increases moderately compared to other unary learning approaches, but it is still substantially better than not using rules (a relative recall increase of 50% already at \( K=10 \)). We found that many of the correct mentions missed by BInc but identified by other methods are due to occasional extractions of incorrect frequent rules, such as partial templates (see Section 6.4.3). This is reflected in the very low precision of the other methods. On the other hand, some correct rules were only learned by BInc, e.g. ‘\( countersuit against X \Rightarrow X sue \)’ and ‘\( X take wife \Rightarrow X marry \)’.

When only one argument is annotated for a specific event mention (28% of ACE predicate mentions, which account for 15% of all annotated arguments), binary rules either miss that mention, or extract both the correct argument and another incorrect one. To neutralize this bias, we also tested the various methods only on event mentions
CHAPTER 6. LEARNING UNARY ENTAILMENT RULES

<table>
<thead>
<tr>
<th></th>
<th>Binary DIRT</th>
<th>Balanced Inclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct</td>
<td>16 (70)</td>
<td>38 (91)</td>
</tr>
<tr>
<td>Partial Template</td>
<td>27 (2665)</td>
<td>6 (81)</td>
</tr>
<tr>
<td>Incorrect</td>
<td>157 (2584)</td>
<td>156 (787)</td>
</tr>
<tr>
<td>Total</td>
<td>200 (5319)</td>
<td>200 (959)</td>
</tr>
</tbody>
</table>

Table 6.1: Rule type distribution of a sample of 200 rules that extracted incorrect mentions. The corresponding numbers of incorrect mentions extracted by the sampled rules is shown in parentheses.

annotated with two or more arguments and obtained similar results to those presented for all mentions. This further emphasizes the general advantage of using unary rules over binary rules.

6.4.3 Analysis

Binary-DIRT

We analyzed incorrect rules both for binary-DIRT and BInc by randomly sampling, for each algorithm, 200 rules that extracted incorrect mentions. We manually classified each rule ‘l ⇒ r’ as either: (a) Correct - the rule is valid in some contexts of the event but extracted some incorrect mentions; (b) Partial Template - l is only a part of a correct template that entails r. For example, learning ‘X decide ⇒ X meet’ instead of ‘X decide to meet ⇒ X meet’; (e) Incorrect - other incorrect rules, e.g. ‘charge X ⇒ convict X’.

Table 6.1 summarizes the analysis and demonstrates two problems of binary-DIRT. First, relative to BInc, it tends to learn incorrect rules for high frequency templates, and therefore extracted many more incorrect mentions for the same number of incorrect rules. Second, a large percentage of incorrect mentions extracted are due to partial templates at the rule left-hand-side. Such rules are leaned because many binary templates have a more complex structure than paths between arguments. As
6.4. EXPERIMENTS

<table>
<thead>
<tr>
<th>Reason</th>
<th>% mentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect rule learned</td>
<td>39.0</td>
</tr>
<tr>
<td>Context mismatch</td>
<td>27.0</td>
</tr>
<tr>
<td>Match error</td>
<td>19.0</td>
</tr>
<tr>
<td>Annotation problem</td>
<td>15.0</td>
</tr>
</tbody>
</table>

Table 6.2: Distribution of reasons for false positives (incorrect argument extractions) by BInc at $K=20$.

<table>
<thead>
<tr>
<th>Reason</th>
<th>% mentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule not learned</td>
<td>61.5</td>
</tr>
<tr>
<td>Match error</td>
<td>25.0</td>
</tr>
<tr>
<td>Discourse analysis needed</td>
<td>12.0</td>
</tr>
<tr>
<td>Argument is predicative</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Table 6.3: Distribution of reasons for false negatives (missed argument mentions) by BInc at $K=20$.

explained in Section 6.3.1 the unary template structure we use is more expressive, enabling to learn the correct rules. For example, BInc correctly learned ‘take $Y$ into custody $\Rightarrow$ arrest $Y$’. However, the binary template ‘$X$ take $Y$ into custody’ cannot be expressed by a binary path. Thus, binary-DIRT could not learn the correct binary rule and instead learned the incorrect rule ‘$X$ take $Y$ $\Rightarrow$ $X$ arrest $Y$’.

System Level Analysis

We manually analyzed the reasons for false positives (incorrect extractions) and false negatives (missed extractions) of BInc, at its best performing cutoff point ($K=20$), by sampling 200 extractions of each type.

From the false positives analysis (Table 6.2) we see that 39% of the errors are due to incorrect rules. The main reasons for learning such rules are those discussed in Section 6.3.2: (a) related templates that are not entailing; (b) infrequent templates. All learning methods suffer from these issues. As was shown by our results, BInc
provides a first step towards reducing these problems. Yet, these issues require further research.

Apart from incorrectly learned rules, incorrect template matching (e.g. due to parse errors) and context mismatch contribute together 46% of the errors. Context mismatches occur when the entailing template is matched in inappropriate contexts. For example, ‘slam X ⇒ attack X’ should not be applied when X is a ball, only when it is a person. The rule-set net effect on system precision is better estimated by removing these errors and fixing the annotation errors, which yields 72% precision.

Table 6.3 presents the analysis of false negatives. First, we note that 12% of the arguments cannot be extracted by rules alone, due to necessary discourse analysis. Thus, a recall upper bound for entailment rules is 88%. Many missed extractions are due to rules that were not learned (61.5%). However, 25% of the mentions were missed because of incorrect syntactic matching of correctly learned rules. By assuming correct matches in these cases we isolate the recall of the rule-set (along with the seeds), which yields 39% recall.

6.5 Conclusions

In this chapter we focused on unsupervised acquisition of unary entailment rules. We presented two approaches for learning such rules from general (non-comparable) corpora: direct learning via distributional similarity measures and deriving unary rules from binary rules. We showed that a simple unary template structure – dependency path ending at the variable – can correctly represent many predicate-argument relationships that cannot be expressed by the standard path structure for binary templates.

Our results suggest the advantages of learning unary rules: (a) unary rule-bases perform better than binary rules; (b) it is better to directly learn unary rules than
to derive them from binary rule-bases. In addition, our novel Balanced-Inclusion measure outperformed all other tested measures, indicating that directional similarity measures are more suitable for learning entailment rules than symmetric ones. Future work may explore additional unary template structures and directional similarity scores.
Chapter 7

Augmenting WordNet-based Inference with Argument Mapping

7.1 Introduction

The previous chapter showed that unsupervised learning from large collections of documents is a promising direction for acquiring a broad scale entailment rule-base. Yet, there are other resources that could be utilized for the task. Some of these are manually created resources, carefully constructed by linguists, that could be harvested for acquiring accurate entailment rules. This chapter addresses this approach, presenting a general framework for generating lexical-syntactic entailment rules from the prominent manually constructed resource in NLP – WordNet (Miller, 1995). This manually constructed lexical database is probably the most widely used resource for lexical inference in NLP tasks, such as QA, IE, IR and RTE (Moldovan and Mihalcea, 2000; Pasca and Harabagiu, 2001; Bar-Haim et al., 2006; Giampiccolo et al., 2007) (see also Section 2.3.2 for WordNet overview). For example, WordNet was utilized by 24 different systems that competed in the RTE-3 and RTE-4 challenges\(^1\), while the

\(^1\)http://www.aclweb.org/aclwiki/index.php?title=RTE_Knowledge_Resources
second most widely used resource, DIRT, was utilized only by 5 systems.

Inference using WordNet typically involves lexical entailment between words (lexical entailment rules) based on WordNet **inferential relations**, relations that capture some type of entailment information (see also Section 2.3.2). In some applications, such as IR, inferring a hypothesis (e.g. an IR query) from a text typically involves only the need to lexically entail each word in the hypothesis from the text. However, in applications such as QA and IE, correctly entailing the relationship between the words in the hypothesis is also necessary. This is especially true when the hypothesis captures a relationship between a predicate and some of its arguments.

When it is also necessary to entail the relations between words in the hypothesis, only a specific type of lexical rules could be utilized for correct entailment inference. These are lexical rules that describe a **substitutable relation** between their LHS and RHS terms. A substitutable rule is one which is considered correct if a the text containing the LHS term of a rule entails the text generated by replacing the LHS term with the RHS term of the rule. For example, ‘buy ⇒ acquire’ is a substitutable lexical rule while ‘buy ⇒ pay’ is not a substitutable rule. As an example of their difference, the first rule could be applied to “IBM bought Cognos for $5 billion” by substituting buy with acquire, generating “IBM acquired Cognos for $5 billion”. On the other hand, simply replacing buy with pay generates the invalid sentence “IBM paid Cognos for $5 billion”.

WordNet’s inferential relations that are also substitutable relations mainly include the **synonym** and **hypernym** relations. However, there are also inferential relations in WordNet that are not substitutable. Specifically, this thesis we focus on entailment rules between predicates, for which some WordNet relations are **non-substitutable**. To be correctly utilized, such relations require a specification of mappings between the syntactic positions of arguments in one predicate to their positions in the other predicate, on top of lexical substitution. For example, ‘X buy Y for Z ⇒ X pay Z for Y’.
Currently, WordNet’s relations do not include argument mapping, the information needed for mapping argument realizations between related predicates. Therefore, correct WordNet inference chains over predicates can be performed only for substitutable relations (mainly synonyms and hypernyms, e.g. ‘buy ⇒ acquire’), for which argument positions do not change. Other relation types that may be used for inference cannot be utilized when the predicate arguments need to be traced as well. Examples include the WordNet entailment relation (e.g. ‘buy ⇒ pay’) and relations between morphologically derived words (e.g. ‘acquisition ⇔ acquire’). Thus, the common practice of tracing arguments only for substitutable relations from WordNet does not utilize WordNet’s full potential for inference.

Our goal is to improve the utilization of information contained in WordNet for inference by constructing argument mappings for WordNet relations that are often used for inference. In this chapter we address several prominent WordNet inferential relations, including verb-noun and verb-adjective derivations and the verb-verb entailment and cause relations. Under the Textual Entailment paradigm, all these relations can be viewed as expressing entailment. Accordingly, we propose a novel framework, called Argument-mapped WordNet (AmWN), that represents argument mappings for inferential relations as entailment rules. These rules are augmented with subcategorization frames and functional roles, which are proposed as a generally-needed extension for predicative entailment rules.

Following our novel representation scheme, we present a concrete implementation of AmWN for a large number of WordNet’s relations. The mappings for these relations are populated by combining information from manual and corpus-based resources, which provides broader coverage compared to prior work and more accurate mappings. Table 7.1 shows typical inference chains obtained using our implementation.

To further improve WordNet-based inference for NLP applications, we address two phenomena that occur in WordNet: (1) sense drifting; (2) rare WordNet senses.
The phenomenon we term sense drifting occurs when the intended meaning of the entailing or entailed rule word changes substantially during rule generation via chaining of several WordNet relations. This change is typically due to accumulated small changes in word meanings that occur in some WordNet relations along the chain (see an example in Figure 7.4). Rules that are generated when sense drifting occurs are incorrect. As it is difficult to identify sense drifts, we propose to filter out all rules that might have been subject to sense drifting during their generation. In addition, rules generated for rare senses of words might hurt inference accuracy since they are more often applied incorrectly to texts when matched against inappropriate, but more frequent, senses of the rule’s words. Since word sense disambiguation (WSD) solutions are typically not sufficiently robust yet, most applications do not currently apply WSD methods. Hence, we propose to optionally filter out such rules using a novel corpus-based validation algorithm.

The remainder of this chapter is organized as follows: Section 7.2 presents our AmWN framework; Section 7.3 describes a concrete implementation of the AmWN framework, followed by manual evaluation of the quality of this implementation; Section 7.4 presents methods for rule filtering; Section 7.5 describes application-based evaluation of the AmWN implementation; Section 7.6 reviews related work and Section 7.7 concludes this chapter.

7.2 Argument Mapping Framework

In this section we present a novel framework for augmenting WordNet with argument mappings for its inferential relations between predicates. We first describe our generic argument mapping representation. It is based on an extension of the standard definition of entailment rules, which is generally useful for predicative entailment rules of any source. We then present the overall Argument-mapped WordNet (AmWN)
Chapter 7. Augmenting WordNet with Argument Mapping

Rule Chains

| shopping:n of X_{obj} ⇒ buying:n of X_{obj} ⇒ buy:v X_{obj} ⇒ pay:v for X_{mod} |
| vote:v on X_{mod} ⇒ decide:v on X_{mod} ⇒ debate:v X_{obj} |
| X_{obj}’s sentence:n ⇒ sentence:v X_{obj} ⇒ convict:v X_{obj} ⇒ convicted:a X_{mod} |
| X_{ind-obj}’s teacher:n ⇒ teach:v to X_{ind-obj} ⇒ X_{subj} learn:v |

Table 7.1: Examples for inference chains obtained using AmWN. Bolded arrows designate inference steps that could be performed correctly only with the additional information within AmWN. Arguments are subscripted with functional roles, e.g. subject (subj) and indirect-object (ind-obj). For brevity, predicate subcategorization frames are omitted.

7.2.1 Argument Mapping Entailment Rules

In our framework we represent argument mappings for inferential relations between predicates through an extension of the common syntactic representations of entailment rules. An example for this standard representation of a rule is ‘X \x leftarrow^{subj} buy \xrightarrow{obj} Y ⇒ X \xleftarrow{subj} pay \xrightarrow{prep-for} Y’, through which “IBM paid for Cognos” can be inferred from “IBM bought Cognos”.

The syntactic structure of the rule templates specifies the required argument positions for correct argument mapping. However, representing entailment rule structure only by syntactic argument positions is insufficient for predicative rules. Correct argument mapping depends also on the specific syntactic functional roles of the arguments (subject, object etc.) and on the suitable subcategorization frame for the predicate mention (see their definition in Section 2.2). For example, ‘X’s buyout ⇒ buy X’ is incorrectly applied to “IBM’s buyout of Cognos” if functional roles are ignored, since IBM plays the subject role in this sentence while X in the rule needs to to take the object role.
Seeking to address this issue, we were inspired by the Nomlex database (Macleod et al., 1998) (see Section 2.3.2 for Nomlex overview) and explicitly specify the subcategorization frame (frame in short) and functional roles participating in each argument mapping rule. As in Nomlex, we avoid the use of semantic roles and stick to the syntactic level, augmenting the representation of templates with: (1) a syntactic functional role for each argument; (2) the valid predicate subcategorization frame for this template. We note that the choice of the set of functional roles depends on the underlying linguistic theory. The set of functional roles we utilize (subject, object etc.) typically coincide with the syntactic dependency relations of the verbal form (see also Section 7.3). A rule example is ‘\(X_{\text{subj}} \text{break}_{\{\text{intrans}\}} \Rightarrow \text{damage}_{\{\text{trans}\}} X_{\text{obj}}\)^1’, which may be applied to “the window broke” to infer “the window was damaged”. More examples are shown in Table 7.1.

Unlike Nomlex records, our templates can be partial: they may contain only some of the possible predicate arguments, e.g. ‘\(\text{buy}_{\{\text{trans}\}} X_{\text{obj}}\)’, where the subject, included in the frame, is omitted from the template. As already discussed in Chapter 6, partial templates are necessary for matching predicate occurrences that include only some of the possible arguments, as in “Cognos was bought yesterday”. Additionally, some resources, such as automatic rule learning methods (see Section 2.3.2 and Chapter 6), can provide only partial argument information, and we would want to represent such knowledge as well. In this chapter we follow our research in Chapter 6 and use only rules between unary templates. Such templates can describe any argument mapping by decomposing templates with several arguments into unary ones, while preserving the specification of the complete subcategorization frame.

To apply a rule, the entailing template must be first matched in the text, which

---

1Functional roles are denoted here by subscripts of the arguments and frames by subscripts of the predicate. We shorthand \(\text{trans}\) for the transitive frame \(\{\text{subject, object}\}\) and \(\text{intrans}\) for the intransitive frame \(\{\text{subject}\}\). For brevity, we will not show all template information when examples are self explanatory.
requires matching the template’s syntactic dependency structure, as well as its functional roles and frame. Such procedure requires texts to be annotated with these types of information. This can be reasonably performed with existing tools and resources, as described for our own text processing implementation in Section 7.5.1.

For example, the mention of buy in previous example was annotated as “Cognos\textsubscript{obj} was bought\textsubscript{trans} yesterday”, allowing the application of the rule ‘buy\textsubscript{trans} X\textsubscript{obj} ⇒ pay\textsubscript{subj,obj,for} for X\textsubscript{for-comp}.’.

Explicitly matching frames and functional roles in rules avoids incorrect rule applications. For example, ‘X\textsubscript{obj}’s buyout ⇒ buy X\textsubscript{obj}’ would be applied only to “Cognos’s buyout by IBM” following proper role annotation of the text, but not to “IBM’s buyout of Cognos”. As another example, ‘X\textsubscript{subj} break\textsubscript{intrans} ⇒ damage\textsubscript{trans} X\textsubscript{obj}’ would be applied only to the intransitive occurrence of break, e.g. “The vase broke”, but not to “John broke the vase”.

Ambiguous cases may occur during text annotation. For example, the role of John in “John’s invitation was well intended” could be either subject or object. We assume that such recognized ambiguities in the text should be left unannotated, consequently blocking incorrect rule application.

### 7.2.2 The Overall AmWN Framework

Following our extension of entailment rules, we present Argument-mapped WordNet (AmWN), a framework for augmenting WordNet’s inferential relations with argument mapping at the syntactic representation level.

The AmWN structure is a directed graph whose nodes represent specific predicates and edges are entailment relations between predicates. A node contains either a single word or a multi-word expression representing a predicate. A predicate may be split into several nodes if it appears in different WordNet synsets or occurs with
7.2. ARGUMENT MAPPING FRAMEWORK

Figure 7.1: AmWN node examples. The upper three nodes represent a single term. The lower two nodes represent a parse sub-tree for a predicate expression. Two options for representing *kick the bucket* are shown, either as a compound verb or as a sub-tree. *SID* is the attached WordNet synset ID and *Frame* is the attached subcategorization frame of each node.

different subcategorization frames, since under different synsets and frames the predicate participates in different entailment relations. Each node is mapped to a specific synset and a specific frame. An AmWN node contains the following information: (1) the dependency parse sub-tree of the single-word or multi-word expression\(^1\); (2) the ID of the WordNet synset to which the predicate belongs; (3) the subcategorization frame of the predicate. Examples for AmWN nodes are presented in Figure 7.1. A more efficient compact representation of this formal description of AmWN nodes is described in Appendix A.

---

\(^1\)To obtain a unified representation, a single term is also represented as a parse sub-tree with only one node containing that term.
As rich as WordNet is, there are still terms missing from some synsets. In addition, there are also missing specific synsets (meanings) of some predicates. We thus do not confine ourselves to the information within WordNet and allow adding terms to specific synsets or generating new synsets when such information is found. See Section 7.3.6 for a description of specific such cases in our current implementation.

There are two types of graph edges that represent entailment rules between nodes: **mapping edges** and **substitution edges**. Mapping edges specify entailment rules for which explicit argument mapping is required. A mapping edge example is presented in Figure 7.2. The nodes connected by a mapping edge describe the predicates that participate in the rule. The edge contains the rest of the rule structure, that is, it contains the rule parts that correspond to the mapped arguments of each of the predicates. To generate the complete rule, the argument structure information specified by the edge is combined with the predicate information that is specified by the nodes.
Substitution edges connect pairs of predicates for which an entailment relation holds, which preserves argument positions. Consequently, a substitution edge needs only to indicate the entailment relation between the two predicates, without any additional information. A substitution edge example is presented in Figure 7.3. This is analogous to how WordNet is often used for inference via the *synonym* and *hypernym* relations. To be connected by a substitution edge, the two predicates must have the same part-of-speech, that is the part-of-speech of the roots of the two parse sub-trees must be the same.

Unlike WordNet relations, substitution edges in AmWN may connect only nodes that have the same subcategorization frame. This is because a change in the predicate’s subcategorization frame may yield a change in the semantic interpretation of the predicate arguments. For example, the correct rule ‘\(\text{smashing:n}\{\text{trans}\} \; \text{by} \; X \Rightarrow X \; \text{break:v}\{\text{trans}\} \)’ is generated by chaining ‘\(\text{smashing:n}\{\text{trans}\} \; \text{by} \; X \Rightarrow X \; \text{smash:v}\{\text{trans}\} \)’ and ‘\(\text{smash:v}\{\text{trans}\} \Rightarrow \text{break:v}\{\text{trans}\} \)’, while the incorrect rule ‘\(\text{smashing:n}\{\text{trans}\} \; \text{by} \; X \Rightarrow X \; \text{break:v}\{\text{intrans}\} \)’ is generated by chaining ‘\(\text{smashing:n}\{\text{trans}\} \; \text{by} \; X \Rightarrow X \; \text{smash:v}\{\text{trans}\} \)’ and ‘\(\text{smash:v}\{\text{trans}\} \Rightarrow \text{break:v}\{\text{intrans}\} \)’.

### 7.2.3 Rule Generation with AmWN

Typical utilization of a given rule-base in an application is the extraction of all the rules in which a given template occurs either as the entailing or as the entailed template. For example, in our ACE experimental setup (Section 4.5), for each seed template all the rules in which this template appears as the entailed template are extracted from a given rule-base.

In AmWN, rules are generated according to the above typical utilization. The application provides an input unary template, augmented with the predicate subcategorization frame and the argument functional role, together with its desired side in the requested rules (either the entailing or the entailed side). AmWN then generates
rule chains (paths in the graph) for the given input template as follows. First, starting nodes that match the predicate in the input template are selected. Then, rules are generated by traversing either incoming or outgoing graph edges transitively, depending on the desired template side: if the input template should be on the entailing side, outgoing edges are traversed; incoming edges are traversed otherwise. Table 7.1 shows examples of rule chains from AmWN.

During graph traversal, the current template structure is maintained. This template consists of the current predicate and the current argument position for this predicate. At first, the predicate is the input predicate and the argument position is the input argument position. For example, if the input is ‘vote on X’, then the predicate is vote and the argument position is the complement of preposition ‘on’. If a substitution edge is traversed, the argument position does not change, only the predicate. For example, traversing ‘vote ⇒ decide’, the new current template is ‘decide on X’. If a mapping edge is encountered, it can be traversed only if the argument position matches that of the edge side related to the current node. When traversing a mapping edge, both the predicate and the argument position are changed. For example, when traversing ‘decide on X ⇒ debate X’, the new template’s predicate is debate and the new argument position is the object position.

Whenever a new node is reached, the new current template is paired with the input template to generate an entailment rule. In our example of the input ‘vote on X’ at the entailing side and the above two edges, the two rules ‘vote on X ⇒ decide on X’ and ‘vote on X ⇒ debate X’ are generated. The subcategorization frame and synset information of each rule are given as well, as they are easily accessible from the information within the graph.

The specific synset-id of the input predicate, if known, may also be added to the input to constrain the relevant starting nodes that are selected for the input template. Otherwise, several starting nodes that convey different predicate meanings may be
selected. Consequently, together with correct rules, many incorrect rules could be generated. For example, if no specific synset-id is provided for the input template ‘X acquire’ on the entailed side, both the rules ‘X buy ⇒ X acquire’ and ‘X learn ⇒ X acquire’ will be generated.

7.3 Our AmWN Implementation

In this section we present our concrete implementation of the AmWN framework, including population of nodes and edges of the AmWN graph. We describe our current population methods for entailment relations between verbs and nouns, adjectives and other verbs, where mapping edges are created based on three types of resources: (1) manual resources other than WordNet; (2) unsupervised corpus-based statistics; (3) manually constructed schematic rules. As we aim at large scale discovery of argument mappings, we specifically avoid supervised methodologies in this implementation, and hence utilize either unsupervised methods, existing resources or a small number of rules that capture common linguistic phenomena.

Since we focus on entailment between predicates, we include in the AmWN graph only *predicative synsets*: all verb synsets, as well as noun and adjective synsets that are identified as describing predicates. In addition, only WordNet relations that correspond to some type of entailment are considered, as described in the following sub-sections.

With respect to rule structure, in this implementation we adopt the set of functional roles used within Nomlex. Each functional role is mapped to possible argument realizations, where each realization is described as a dependency relation. We remind that, as in previous chapters, in this implementation we utilize the Minipar dependency parser and adapt its dependency-tree output form. Thus, the dependency
relations that describe the argument realizations are taken from Minipar representation. We note that the Nomlex functional role set can typically be unambiguously mapped to specific Minipar verbal argument dependency relations, such as the subject role to the Minipar subj relation and the object role to the Minipar obj relation.

7.3.1 Node Population

Nodes are first generated based on WordNet’s synsets. The major challenge at this step is to recognize which synsets in WordNet are predicative (i.e. describing a predicate). All verb synsets are predicative, but this does not hold for all noun and adjective synsets. First, there are nouns and adjectives that do not correspond to a predicate at all, such as city, tomato or happy. However, even words that do correspond to a predicate in some of their synsets might have synsets that are not predicative. For example, one synset of acquisition that is derivationally related to acquire is not predicative: “an ability that has been acquired by training”. Some hyponyms of this synset include literacy, tradecraft, mastership and mixology.

Nominal and Adjective Predicative Synsets

For the task of recognizing predicative synsets, nominalizations can be partitioned into two types. The first type, termed here predicate-centered nominalizations, corresponds to their related verb itself, e.g. acquisition and employment. The second type, termed here argument-centered nominalizations, corresponds to an argument with a specific functional role with respect to the related verb, e.g. employer (the subject of employ) and employee (the object of employ). We consider noun synsets as predicative for predicate-centered nominalizations if they are direct or indirect hyponyms of the act high-level WordNet synset. Noun synsets are considered predicative for argument-centered nominalizations if they are (transitive) hyponyms
of the *person* high-level WordNet synset\(^1\).

For adjectives, all adjective synsets that are related to a verb synset via the *derivationally related* WordNet relation are considered predicative.

**Node Generation**

In our framework, different subcategorization frames are treated as having different meanings, since different frames may be involved in different entailment rules. Thus, for each term in a predicative synset we collect its list of possible subcategorization frames. We obtain frame specifications for verbs from WordNet\(^2\). Since WordNet does not provide subcategorization frame information for noun predicates, these are taken from Nomlex-plus (Meyers et al., 2004) (see Section 2.3.2). For all adjectives we specify a single frame, containing a single argument role of the adjective’s head.

As rich as Nomlex-plus is, it still does not include all nominalizations. For example, the nominalizations *divorce, endorser* and *striking* are missing. WordNet has a much richer set of nominalizations that we would like to utilize. To make WordNet nominalizations missing from Nomlex usable we need to provide their subcategorization frame information as well. This is done by allowing each such nominalization to inherit its associated subcategorization frames either from one of its synonyms that does appear in Nomlex or, if no such synonym exists, from its closest hypernym that appears in Nomlex. Thus, *divorce* inherits its information from *separation, endorser* inherits from *subscriber* and *striking* from *hit*. A by-product of this process is the automatic extension of Nomlex-plus with 7955 new nominalization entries, on top of its 5000 entries, based on the inherited information.

\(^1\)We chose only this high-level synset because other synset candidates did not contain only predicative synsets within their WordNet hyponymy hierarchy, and thus would have produced incorrect inferences.

\(^2\)We also tried using VerbNet (Kipper, Dang, and Palmer, 2000), without any current performance improvement.
Once predicative synsets are identified and the subcategorization frame information is obtained for their terms, the AmWN nodes can be generated. Given a predicative synset $s$, we generate for each term $t$ in the synset and each possible frame $f$ of $t$ a new node, containing $t$ represented as a parse sub-tree of a single node, $s$ as the synset-Id and $f$ as the node’s frame. If no frame information was obtained for $t$, no node is generated for it.

If $t$ is a multi-word expression, e.g. *kick the bucket*, some parsers might recognize it as such and represent it as a single term, while others might represent it as a parse sub-tree. To overcome this representation problem, we try to generate for each multi-word expression an additional node, on top of its single term node, where $t$ is represented by its parse sub-tree. As parsers find it difficult to parse stand-alone predicative expressions, which are not part of a complete sentence, we manually defined a heuristic rule that generates parse trees for a specific mutli-word form: ‘verb {a|an|the} noun’, such as *kick the bucket* or *hit the hay* (an example for such generated tree is in Figure 7.1). We currently do not generate parse trees to other types of multi-word expressions.

Some predicates are missing from WordNet, or their predicative synsets are missing. We attempt to add additional nodes that represent these predicates, as described in Section 7.3.6.

The next sub-sections describe the various edges that are added to the AmWN graph between the above generated nodes.

### 7.3.2 Substitution Edge Population

We populate the AmWN graph with substitution edges for WordNet’s hyponyms and synonyms, e.g. ‘buy ⇔ purchase’ and ‘buy ⇒ acquire’. A bidirectional substitution edge is drawn between any two synonymous terms. A directional edge is drawn from
each hyponym/troponym to its hypernym. As mentioned earlier, edges are drawn only between nodes that have the same subcategorization frame. In addition, the two terms must share the same part-of-speech of their parse sub-tree roots (e.g. for connecting a single term and a parse sub-tree, as for ‘arrest ⇒ take into custody’).

### 7.3.3 Nominalization Relations

The relation between a verb and its nominalizations, e.g. between *employ* and *employment*, is described in WordNet by the *derivationally related* relation. We first extract all AmWN noun-verb node pairs whose synset-ids are connected by the *derivationally related* relation. We additionally require that the two nodes will also have the same subcategorization frame.

To generate the relevant mapping edges between each pair of nominalization node $n$ and a verb node $v$, we utilize Nomlex-plus. First, the Nomlex entry related to the nominalization described in $n$ is identified. The information it contains for the specific subcategorization frame of $n$ is extracted, such as the entry part in Figure 2.1 on page 19, which describes the transitive frame information for *employment*. From this information mapping edges are generated between $n$ and $v$ by going over all the functional roles within the frame, and for each functional role $r$ generating a mapping edge between each possible nominal argument realization described in the Nomlex entry for $r$ and each possible verbal argument realization of $r$. For example, the mapping edges ‘$X_{obj}$’s *employment* $\Leftrightarrow$ *employ* $X_{obj}$’ and ‘*employment of* $X_{obj}$ $\Leftrightarrow$ *employ* $X_{obj}$’ are generated for the *object* argument role from the information in Figure 2.1.

Additionally, we create self edges for $n$ that map between any two possible argument realizations of each functional role. For example, the mapping ‘$X_{obj}$’s *employment* $\Leftrightarrow$ *employment of* $X_{obj}$’ is derived for the *object* argument role in Figure 2.1.
CHAPTER 7. AUGMENTING WORDNET WITH ARGUMENT MAPPING

<table>
<thead>
<tr>
<th>Lexical Relation</th>
<th>Extracted Mappings</th>
</tr>
</thead>
<tbody>
<tr>
<td>buy ⇒ pay</td>
<td>buy for (X \Rightarrow \text{pay } X)</td>
</tr>
<tr>
<td></td>
<td>(X \text{ buy } \Rightarrow X \text{ pay})</td>
</tr>
<tr>
<td>divorce ⇒ marry</td>
<td>divorce from (X \Rightarrow \text{marry } X)</td>
</tr>
<tr>
<td></td>
<td>divorce from (X \Rightarrow X \text{ marry})</td>
</tr>
<tr>
<td>kill ⇒ die</td>
<td>kill (X \Rightarrow X \text{ die})</td>
</tr>
<tr>
<td></td>
<td>kill among (X \Rightarrow X \text{ die})</td>
</tr>
<tr>
<td>breathe ⇒ inhale</td>
<td>breathe (X \Rightarrow \text{inhale } X)</td>
</tr>
<tr>
<td></td>
<td>breathe in (X \Rightarrow \text{inhale } X)</td>
</tr>
<tr>
<td>remind ⇒ remember</td>
<td>remind (X \Rightarrow X \text{ remember})</td>
</tr>
<tr>
<td></td>
<td>remind of (X \Rightarrow X \text{ remember } X)</td>
</tr>
<tr>
<td>teach ⇒ learn</td>
<td>teach (X \Rightarrow \text{learn } X)</td>
</tr>
<tr>
<td></td>
<td>teach to (X \Rightarrow X \text{ learn})</td>
</tr>
<tr>
<td>give ⇒ have</td>
<td>give (X \Rightarrow \text{have } X)</td>
</tr>
<tr>
<td></td>
<td>give to (X \Rightarrow X \text{ have})</td>
</tr>
</tbody>
</table>

Table 7.2: Some argument mappings for WordNet verb-verb relations discovered by unary-DIRT.

7.3.4 Verb-Verb Relations

There are two inferential relations between verbs in WordNet that do not preserve argument positions (not substitutional): *cause* and *entailment*. Unlike for nominalizations, there is no broad-coverage manual resource of argument mapping for these relations. Hence, we turn to unsupervised approaches that learn entailment rules from corpus statistics, which specify these argument realizations.

Many algorithms were proposed for learning entailment rules between templates from corpora, but typically with mediocre accuracy (see the analysis in Section 4.3.3). However, we only utilize rules between verbs for which WordNet already indicates the existence of an entailment relation. Thus, we are not affected by rules that wrongly relate non-entailing verbs. For this purpose, we acquired a rule-set containing the top 300 rules for every unary template in the Reuters RCV1 corpus by implementing the unary-DIRT algorithm, which was shown to have relatively high recall compared to
other algorithms (see the analysis in Section 6.4.2).

To extract argument mappings, we identify all AmWN verb node pairs whose synsets are related in WordNet by a *cause* or an *entailment* relation. For each pair, we look for unary-DIRT rules between the terms of the two nodes. For example, *buy* entails *pay* in WordNet, so we look for rules that map ‘buy ⇒ pay’. Table 7.2 presents examples for such discovered mappings. While unary-DIRT rules are not annotated with functional roles, the set of functional roles we work with can be derived from the verbal dependency relations available in the rule’s templates, as explained in the beginning of this section. The obtained rules are then added to AmWN as mapping edges.

As the frame of a predicate is currently not recognized in the learned rules (only the argument functional role is identified), the rules are applied to all possible frames of the predicate pair that contain the recognized argument functional roles. This may generate incorrect edges, as demonstrated in the manual analysis (Section 7.3.7, page 112).

We only search for rules that map an argument realization of a specific functional role in the frame of one verb to any argument realization for the other verb. Focusing on frame elements avoids extracting mapping rules learned for adjuncts, which tend to be of low precision, since adjuncts are not part of either verb subcategorization frame.

Finally, WordNet’s policy for its *entailment* relation is that two verbs are connected by *entailment* only if the subject of the entailing verb is also the subject of the entailed verb. Thus, we created a schematic rule that maps between the subjects of any two verbs connected by the *entailment* relation. In addition, corpus-based rules that map the subject to another frame role (e.g. object) are ignored as a precaution measure.
CHAPTER 7. AUGMENTING WORDNET WITH ARGUMENT MAPPING

Mappings Induced from WordNet’s Frame Description

As different frames generate different nodes in AmWN, it would be useful to map between different verb frame variations to increase the AmWN graph connectivity. We currently utilize the semantic role information described within the WordNet frames to identify mappings between ditransitive frame variations with and without the preposition ‘to’.

This mapping is done by examining WordNet’s frame descriptions, which contain some basic semantic selectional preference for each role. For example, some of the frames described for the verb show are: ‘Somebody shows somebody’, ‘Somebody shows somebody something’ and ‘Somebody shows something to somebody’. Specifically, the last two describe variants for the ditransitive frame of show. If the semantic preferences of the direct object and indirect object roles are different, as in the above example (something vs. somebody in ‘Somebody shows somebody something’), then the argument mapping between the two ditransitive frame variants can be automatically extracted unambiguously. Had the frame contained ‘Somebody shows somebody somebody’, we wouldn’t know which of the objects to map to the ‘to’ complement. In unambiguous cases, two mapping edges are generated, one for the direct object position and one for the indirect object position. For example, ‘show $\xrightarrow{\text{obj}_2} X$ $\Leftrightarrow$ show $\xrightarrow{\text{obj}_1} X$’ and ‘show $\xrightarrow{\text{prep}-\text{to}} X$’ were generated for show).

7.3.5 Adjective-Verb Relations

The relation between a verb and a derived adjective is described by the WordNet direvationally related relation. To our advantage, adjectives show some regularities in the manner their arguments are mapped to their related verbs: (1) Gerund adjectives, e.g. ‘the acquiring $X$’, are mapped to the subject verbal role; (2) Adjectives

\footnote{Minipar’s relations for two consecutive object positions are labeled $\text{obj}_1$ and $\text{obj}_2$.}
ending with ‘ive’, e.g. ‘the protective X’, are mapped to the subject verbal role; (3) Past-participle adjectives, e.g. ‘the acquired X’, are mapped to the object role; (4) Adjectives ending with ‘able’ or ‘ible’, e.g. ‘a manageable X’, are mapped to the object role (with the exception of the adjective able itself).

We created schematic rules that capture the above regularities. Whenever an AmWN adjective node is one of the above types, the appropriate mapping edge to the related verbal node is added, e.g. ‘the elected X ⇒ elect X’. For other adjectives, the rules learned by the unary-DIRT algorithm are utilized, as for verb-verb mappings. If a rule is found that maps between an adjective and its related verb in WordNet, the appropriate mapping edge is added, e.g. ‘the poisonous X ⇒ X poison’.

Finally, self edges are added for each adjective node, mapping between two possible argument realizations of adjectives: (1) The adjective is the direct modifier of the argument, e.g. ‘the protective X’; (2) The argument and the adjective are connected via the copula be, e.g. ‘X is protective’.

7.3.6 Extending WordNet with Missing Information

For various adjective-related predicates we could detect lexical information that is currently missing from WordNet. We next describe our approaches to extend WordNet with such information.

Adjectives

WordNet’s current coverage of adjective-verb relations is lacking. For example, married and divorced are not related to their verbal forms. We extend WordNet with this missing information, together with argument mappings between such adjectives and their related verbs.
We construct additional adjective-verb relations by identifying morphological derivations of verbs into adjective as follows. For every AmWN verbal node \( vn \) with verb \( v \), we first generate its morphologically related gerund and past-participle adjectives, e.g. *marrying* and *married* for the verb *marry*\(^1\). Then, for each generated adjective \( a \), if no WordNet synset containing \( a \) exists such that it is also related to at least one of the synsets of the target verb \( v \) in WordNet via the direvationally related relation, we create a new node \( an \) containing \( a \) as its term. Finally, a mapping edge is added between the new node \( an \) and \( vn \) according to the adjective’s type: gerunds are mapped to realizations of the subject role and past-participles to realizations of the object role. Self mapping edges are added as well, as in Section 7.3.5.

**Copula Constructions**

Currently, there are only few verbal phrases in WordNet. We focus here on a specific type of verbal phrases that is based on a single syntactic structure. This type, which we term here *copula constructions*, includes verb phrases constructed by connecting an argument to a past participle adjective using a copula verb. To demonstrate how copula constructions can be added to AmWN, we automatically generated for each verb \( verb \) its related copula construction ‘*get verbed*’, where *get* is the copula verb and *verbed* denotes the past-participle adjective related to \( verb \), e.g. ‘*get elected*’ for *elect*.

For each transitive or ditransitive AmWN verbal node, the related ‘*get-verbed*’ node is constructed with the same synset-id and frame. We map the subject, object and, if applicable, indirect-object between the two nodes. In the case of transitive frames the mappings are: ‘\( X \ get \ verbed \leftrightarrow verb \ X \)’ and ‘*get verbed by \( X \leftrightarrow X \ verb*’ (e.g. “John got elected by Jane” \( \leftrightarrow \) “Jane elected John”). In the case of ditransitive

---

\(^1\)Verbal expressions containing more than one word, e.g. *take in* and *get married*, were ignored in this phase.
frames the mappings are: ‘$X$ get verbed ⇔ verb to $X$’, ‘get verbed $X$ ⇔ verb $X$’ and ‘get verbed by $X$ ⇔ $X$ verb’ (e.g. “John got paid $100 by Jane” ⇔ “Jane paid $100 to John”).

7.3.7 Manual Analysis of AmWN Population Quality

Many mapping edges were added to the AmWN graph in our implementation. Prior to our main application-based evaluations of the populated graph we wanted to manually assess the quality of the various types of edges, which come from different resources. Such an analysis would also give us an indication of the expected overall quality of the resource.

We next describe our manual evaluation of the edges in the AmWN graph. We separated the evaluation into two parts. The first is an evaluation of edges created based on entailment information already existing in WordNet. The second is the evaluation of the nodes and edges added for information missing from WordNet. We provide these two evaluations separately as inherently these are different tasks.

Analysis of Mappings for Existing WordNet Information

Altogether, our implementation identified 9,715 different non-substitutional entailment relations between synsets in WordNet. Table 7.3 presents a summary of the distribution of these relations by the their types: noun-verb, adjective-verb and verb-verb. As can be seen, most WordNet entailment relations are based on morphology while only 6.5% are verb-verb relations that capture some semantic inference. Yet, the substitutable synonym and hypernym relations, which also represent semantic entailments, are abundant in WordNet. Hypernym relations are especially abundant
### Table 7.3: A summary of the relations in WordNet that were identified as representing entailment, and the percentage of these relations for which our population methodologies found at least one mapping edge.

<table>
<thead>
<tr>
<th>Relation Type</th>
<th># of Relations in WordNet</th>
<th># of Relations with Mapping Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noun-Verb (derivationally related)</td>
<td>7689</td>
<td>5682 (74%)</td>
</tr>
<tr>
<td>Adjective-Verb (derivationally related)</td>
<td>1398</td>
<td>1222 (87%)</td>
</tr>
<tr>
<td>Verb-Verb (cause, entailment)</td>
<td>628</td>
<td>585 (70%)</td>
</tr>
<tr>
<td>Total</td>
<td>9715</td>
<td>7345 (76%)</td>
</tr>
</tbody>
</table>

### Table 7.4: The evaluated accuracy of a random sample of mapping edges created in our current implementation for the entailment information presented in WordNet. Both overall and per type accuracy is presented, together with the number of samples evaluated.

<table>
<thead>
<tr>
<th>Edge Type</th>
<th>Sample Size</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noun-Verb</td>
<td>574</td>
<td>99.7</td>
</tr>
<tr>
<td>Adjective-Verb</td>
<td>94</td>
<td>96.8</td>
</tr>
<tr>
<td>Verb-Verb</td>
<td>132</td>
<td>98.5</td>
</tr>
<tr>
<td>Total</td>
<td>800</td>
<td>99.2</td>
</tr>
</tbody>
</table>

between nouns. As there are many noun-verb relations, chaining substitutional relations such as hypernyms with non-substitutional relations such as noun-verb derivations yields many additional semantic entailments, as demonstrated in Table 7.1 on page 92.

Our edge population procedures discovered mapping edges for 76% of the identified relations. Table 7.3 presents the distribution across the different relation types. As can be seen, verb-verb mappings were the most difficult to discover. However, our verb-verb mappings are learned from a corpus, and we expect that with a larger corpus the mapping coverage would increase for both verb-verb and adjective-verb relations.
All together, 34,784 mapping edges were added for 7,345 WordNet non-substitutable relations. As the numbers show, sometimes several arguments of a single WordNet relation were mapped by different mapping edges. Additionally, 37,984 mapping edges were added for mapping between different verbal frame variations (as in Section 7.3.4) and as self edges that map between different nominal and adjective frame realizations (as in Section 7.3.3). These edges describe possible variations in argument realizations for a given predicate. Since they involve only syntactic changes without any lexical change, they do not relate to any WordNet relation.

The author of this thesis manually evaluated for correctness a random sample of 800 mapping edges from all the types discussed above. Table 7.4 presents a summary of this evaluation. The analysis shows that the edges added to AmWN are highly accurate, with an average accuracy of 99%. In this sample, only 2 incorrect edges were found for noun-verb relations, 3 for adjective-verb relations and 2 for verb-verb relations. This shows that our methodologies for populating mapping edges are very reliable, especially when compared to prior work (see Section 7.6).

Further analyzing these incorrect mappings, we found that those between nouns and verbs occurred when a mapping was inherited from a hypernym of an argument-centered nominalization, while the current nominal did not possess such mapping with the related verb. In our sample, *invitee* was incorrectly mapped as the subject function of *invite*. This was due to erroneous inheritance from *invitee*’s hypernym *visitor* (*visitor* is mapped to *visit* as its subject role). In the other incorrect sampled relation, *saint* was incorrectly mapped as the subject role of the verb *sanctify*. The erroneous mapping was taken from the indirect hypernym *leader*, which is mapped as the subject role to the verb *lead*.

In the adjective-verb case, all 3 incorrect mappings are irregular cases of adjectives ending with ‘*ive*’, where the mapping to their related verbs is not as the verbal subject role: ‘*expensive ⇔ expend*’, ‘*nominative ⇔ nominate*’ and ‘*optative ⇔ opt*’. 
Table 7.5: The manually evaluated accuracy of the random samples of the new adjective nodes and copula-construction nodes missing from WordNet created in our current implementation.

Finally, in the case of verb-verb relations, one incorrect mapping is the result of incorrect rule learned from the corpus: ‘buy X ⇒ pay X’. The other incorrect mapping was generated from the correct rule ‘X ignite ⇒ X combust’. However, since the predicate frames are not identified, it was utilized to generate mappings between all frames of combust. While the rule is correct only for the transitive frame of combust, the sampled edge connected between the node containing the intransitive frame of combust.

Analysis of Added Information that was Missing from WordNet

Another evaluation was performed separately for the nodes and mapping edges that were created with the methods described in Section 7.3.6. Table 7.5 summarizes the accuracy of these nodes and their mappings to their related verbal nodes. The high accuracy of both types of rule shows that in general almost all gerund and past-participle forms of a verb can be correctly utilized as related adjectives or as part of a copula construction.

34,877 new adjective nodes, such as married, confronted, hiking and announcing, were added. About two nodes were created for each verb node, one for its related gerund adjective and one for the related past-participle adjective. While the number of distinct adjectives generated is 15,047, more than one node per adjective may be added if the related verb occurs in more than one synset or has more than one frame.
A random sample of 134 new adjective nodes was evaluated by the author of this thesis. Only two generated adjectives were evaluated as incorrect: taking (as in ‘the taking X’) and misspending (as in ‘a misspending X’).

14,646 new nodes containing copula-constructions with the get copula verb, such as get cleaned and get elected were added. These constructions involve 6,918 different verbs, where more than node is created for a verb if that verb occurs with more than one frame or in more than one synset. A random sample of 66 new copula-construction nodes was evaluated. Only two generated expressions were evaluated as incorrect: get meant (related to the verb mean) and get domiciliated.

7.4 Rule Filtering - Improving Precision for Chained Inference

In preliminary analysis we found two phenomena, sense drifting and rare senses, which may reduce the effectiveness of AmWN-based inference even when each graph edge by itself, taken out of context, is valid. To address these phenomena within practical inference we propose the following optional methods for rule filtering.

7.4.1 Sense Drifting

WordNet verbs typically have a more fine-grained set of synsets than their related nominalizations and adjectives. There are cases where several verb synsets are related to the same nominal or adjective synset. Since entailment between a verb and its morphological derivations is bidirectional, all such verb synsets would end up entailing each other via the nominal or adjective node.

Alas, some of these connected verb synsets represent quite different meanings, which results in incorrect inferences. This problem, which we call sense drifting,
### Synset Members

| (verb)  | collar, nail, apprehend, arrest, pick up, nab, cop |
| (noun)  | apprehension, catch, arrest, collar, pinch, taking into custody |
| (verb)  | get, catch, capture |
| (noun)  | capture, seizure |
| (verb)  | seize |
| (verb)  | kidnap, nobble, abduct, snatch |

<table>
<thead>
<tr>
<th>WordNet Gloss</th>
</tr>
</thead>
<tbody>
<tr>
<td>take into custody</td>
</tr>
<tr>
<td>the act of apprehending (especially apprehending a criminal)</td>
</tr>
<tr>
<td>succeed in catching or seizing, especially after a chase</td>
</tr>
<tr>
<td>the act of taking of a person by force</td>
</tr>
<tr>
<td>take or capture by force</td>
</tr>
<tr>
<td>take away to an undisclosed location against their will and usually in order to extract a ransom</td>
</tr>
</tbody>
</table>

Figure 7.4: A WordNet sense-drifting traversal, generating the incorrect inference ‘kidnap ⇒ arrest’.

is demonstrated in Figure 7.4, showing how the incorrect rule ‘kidnap ⇒ arrest’ is generated when traversing the AmWN graph. To address the problem, we add a constraint to rule chaining such that each rule generation chain would include at most one verb-noun edge and at most one verb-adjective edge, which still connects the noun, verb and adjective hierarchies but prevents from sense drifting to occur.

#### 7.4.2 Rare Senses

Some word senses in WordNet are rare. That is, the probability of encountering a sentence containing a word under a specific rare sense of that word is very low compared to encountering sentences containing that word under any of its other senses. Thus, applying rules that correspond to such senses yields many incorrect inferences, since they are typically matched against other senses of the word. An
7.4. RULE FILTERING

example for such a rule is ‘have $X \Rightarrow X$ is born’, corresponding to a rare sense of have. WSD is a possible solution for this problem. However, most state-of-the-art NLP systems do not rely on WSD methods, which are currently not sufficiently robust (for example, WSD was hardly used in Textual Entailment systems competing in the RTE challenges).

To circumvent the rare sense problem, we attempt to filter out such rules. To that end, each AmWN rule is validated against our unary-DIRT rule-set, which, being corpus-based, contains mostly rules for frequent senses. A rule is considered directly-validated if it is in the corpus-based rule-set, or if it is either a nominal-verb or a adjective-verb rule, which describes a reliable morphological change for a predicate. In our implementation, all the mapping edges that are added to AmWN are of these types and thus considered directly-validated. For each rule, the AmWN graph-path that generated it is automatically examined. A given rule is considered valid if there is a sequence of directly-validated intermediate rules along the path whose transitive chaining generates the rule. Invalid rules are filtered out.

To illustrate this algorithm, suppose the rule ‘$a \Rightarrow d$’ was generated by the chain ‘$a \Rightarrow b \Rightarrow c \Rightarrow d$’ in the graph. This rule is considered valid if there is a transitive rule chain yielding ‘$a \Rightarrow d$’, e.g. ‘$a \Rightarrow b$’, ‘$b \Rightarrow c$’, ‘$c \Rightarrow d$’ or ‘$a \Rightarrow b$’, ‘$b \Rightarrow d$’}, whose rules are all directly-validated.

We note that though all mapping edges added to AmWN are considered directly validated by this approach, a path may contain also substitution edges. Thus, other intermediate rules, either the substitution edges themselves or a combination of several edges, must also be directly validated for the path to be considered valid. For example, ‘shopping of $X \Rightarrow pay for X$’ was extracted by traversing the path ‘shopping of $X \Rightarrow buying of X \Rightarrow buy X \Rightarrow pay for X$’. The rule is considered valid because the two intermediate rules in {‘shopping of $X \Rightarrow buy X$’, ‘buy $X \Rightarrow pay for X$’} were directly validated.
Input: path \(\{p_i\}_{i=1}^{n}\)

Output: true if the path is valid, false otherwise

**VALIDATE-PATH:**

1: for \(l \leftarrow 1...n\) do
2:     for each \(i, j\) s.t. \(j - i = l\) do
3:         if \(\{p_i, p_j\}\) is directly validated then
4:             valid\([i, j]\) \(\leftarrow\) true
5:         else
6:             valid\([i, j]\) \(\leftarrow\) false
7:     for \(k \leftarrow i+1...j-1\) do
8:         if valid\([i, k]\) and valid\([k, j]\) then
9:             valid\([i, j]\) \(\leftarrow\) true
10:            Break
11: return valid\([1, n]\)

Figure 7.5: Algorithm outline for efficiently evaluating a path as valid or not.

Since the number of possible rule chains along a path is exponential in the path length \(2^{n-2}\) for a path of length \(n\), we apply a dynamic programming approach to efficiently evaluate all possibilities, whose pseudo-code is presented in Figure 7.5. All node pairs along the path are evaluated. Each node pair \(\{l, r\}\) is marked as valid if it is either directly validated or it is a combination of two valid sub-paths, \(\{l, k\}\) and \(\{k, r\}\), where \(k\) is a node on the path between \(l\) and \(r\). It takes \(O(n)\) steps to evaluate each node pair and there are \(O(n^2)\) pairs, resulting in a total complexity of \(O(n^3)\).

### 7.5 Application-oriented Evaluations

Manual evaluation of AmWN showed that the mapping edges created by our implementation are very accurate (see Section 7.3.7). However, when applying an entailment rule resource such as AmWN in real NLP applications, their performance may be hindered due to various reasons (Mirkin, Dagan, and Shnarch, 2009). We
addressed the problems of sense drifting and rare word senses in Section 7.4. However, there may be other reasons for inference errors, such as that in real texts the distribution of rule applications may lean towards incorrect rules, while manual evaluation assumes uniform distribution over rule application (as discussed in Section 4.4 regarding evaluation methodologies).

Thus, we would like to evaluate the contribution of AmWN rules to realistic NLP inferences, and to compare it to the common practice of using WordNet without argument mapping. We experimented with two different setups. As a typical and informative setting we chose the ACE setup presented in Section 4.5, under which we could isolate inference operations as applications of single rules. Our second experimental setting utilizes the RTE datasets and setup, which are common benchmarks for entailment inference.

In the following subsection we first describe how we perform the necessary annotation of text that enables rule applications. We then describe our ACE and RTE experiments. We note that unlike for the ACE setup, where each entailment inference corresponds to a single rule application, a hybrid inference is required for RTE testing, typically involving approximate inference and heterogeneous knowledge resources. Thus, under such setup it is less informative to measure the impact of a single knowledge resource like ours on overall system performance, since it gets masked by the impact of other system components. We thus consider the ACE setup as our main evaluation, and provide a thorough analysis of its results.

### 7.5.1 Text Annotation with Functional Roles and Subcategorization frames

As explained in Section 7.2.1, when applying a rule the entailing template must be first matched in the text. This matching requires texts to be annotated with their syntactic
dependency structure, argument functional roles, and predicate subcategorization frames.

To produce such annotation, each target text is dependency-parsed with Minipar (Lin, 1998b) and annotated with functional roles and frames for each predicate mention. To choose the subcategorization frame of a given predicate mention we follow the basic logic: (a) a possible frame for this predicate could be a candidate for this mention only if all of its functional role members are annotated for the arguments of the mention; (b) if two frames are candidates and the set of functional roles of one frame is a subset of the functional role set of the other frame, then the frame with the smaller set cannot be a candidate. Consequently, the largest frame, with respect to its functional role set, is the one chosen for annotation. For example, the two candidate frames for \textit{buy} in “\textit{I subject bought a car object}” are the transitive and the intransitive frames, as both the subject role and the object role are annotated. Since the the roles of the intransitive frame are a subset of those of the transitive frame, the transitive frame is the frame chosen this mention. This procedure chooses the correct frame for annotation unless some frame information is missing for a specific predicate, and thus an incorrect frame might be chosen instead, if it contains a subset of the correct (but missing) frame roles. In addition, ambiguity in functional role annotation would prevent choosing a specific frame for annotation.

We choose candidate frames for a given verb predicate mention from all the frames provided for this verb in WordNet. The functional role for each argument of a verb mention is derived directly from the corresponding dependency tree relation, as explained in Section 7.3 on page 99. Frames and functional roles are annotated for nominalization mentions using our extended Nomlex-plus database (see Section 7.3.3). For each nominalization mention, we find the largest subcategorization frame for this nominalization in the database whose possible syntactic argument positions match those of the mention’s argument positions. The arguments are then annotated with
the specified roles of the chosen frame. For example, the mention of *acquisition* in “Cisco’s acquisition of Topspin was big news” is annotated with the transitive frame, where *Cisco* and *Topspin* are annotated with the subject and object roles respectively. Ambiguous cases, where the same argument position could match multiple roles, are left unannotated, as discussed and exemplified in Section 7.2.1 on page 94.

As discussed in Section 7.2.1, many predicate mentions are partial mentions. That is, the predicate occur only with some of its possible arguments. For example, in a verb passive form, the verb’s subject is missing. To overcome this phenomenon, we allow for missing *subject* roles in cases of passive mentions. For example, in the text “*John was paid $50,000 for his car*”, *John* is annotated as the indirect-object of *pay*, *$50,000* as its direct object and *for car* as a preposition modifier of *pay*. Thus, the chosen frame for *pay* is its ditransitive frame, even though the subject role is missing due to the passive form.

Similar to the case of verb annotation, we allow for missing argument roles in nominalization mentions. Unlike verbs however, due to the nature of nominalizations, we allow for any argument to be missing, including object and indirect-object roles. We denote an occurrence of a nominalization mention where some of its frame arguments are missing a *partial occurrence*. A frame may match even if its occurrence is partial, as long as this partial occurrence is unambiguous with respect to partial occurrences of other frames. For example, in “*the payment to John by Jane*”, *John* is annotated as the indirect-object of *payment*, *Jane* as its subject and the chosen frame is the ditransitive frame, even though the object role is missing.
7.5.2 The ACE Evaluation

Experimental Setting

In our main evaluation, we follow the ACE experimental setup presented in Section 4.5. To utilize the ACE dataset for evaluating rule applications, each ACE event predicate was represented by a set of unary seed templates, one for each event argument (a similar setup is also presented in Section 6.4.1). Example seed templates for *Injure* are ‘A injure’ and ‘injure V’. Each event argument is mapped to the corresponding seed template variable, e.g. *Agent* to *A* and *Victim* to *V* in the above example. We manually annotated each seed template with a subcategorization frame and an argument functional role, e.g. ‘injure\{trans\} V_{obj}’. We also included relevant WordNet synset-ids, so only rules fitting the target meaning of the event will be generated by WordNet and AmWN. In this experiment, we focused only on the core semantic arguments. Adjuncts (time and place) were ignored since they typically don’t require argument mapping, the main target for our assessment. The ACE corpus was automatically annotated according to the procedure described in Section 7.5.1.

Argument mentions for events were found in the annotated corpus by matching either the seed templates or the templates entailing them in some rules. The matching procedure follows the one described in Section 7.2.1. Templates are matched using a syntactic matcher that handles simple syntactic variations such as passive-form and conjunctions. For example, ‘wound\{trans\} V_{obj} ⇒ injure\{trans\} V_{obj}’ was matched in the annotated text “Hagel_{obj} was wounded\{trans\} in Vietnam”.

Results

We tested five different rule-set configurations: (1) only the seed templates, without any rules; (2) rules generated based on WordNet 3.0 without argument mapping,
Table 7.6: Recall (R), Precision (P) and F1 results for the various tested configurations.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>R (%)</th>
<th>P (%)</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) No Rules</td>
<td>13.5</td>
<td>63.0</td>
<td>20.7</td>
</tr>
<tr>
<td>(2) WordNet</td>
<td>18.3</td>
<td>32.2</td>
<td>17.8</td>
</tr>
<tr>
<td>(3) WordNet with roles and frames</td>
<td>17.5</td>
<td>35.3</td>
<td>18.5</td>
</tr>
<tr>
<td>(4) WordNet with roles, frames and rule validation</td>
<td>16.5</td>
<td>46.9</td>
<td>20.4</td>
</tr>
<tr>
<td>(5) AmWN</td>
<td>20.8</td>
<td>43.9</td>
<td>24.2</td>
</tr>
</tbody>
</table>

using only synonym and hypernym relations and ignoring functional role and frame information (this is the typical WordNet usage); (3) rules generated as in (2), but with functional role and frame information; (4) WordNet rules from (3) filtered using our corpus-based validation method for rare senses; (5) rules generated from our complete AmWN implementation.

Each configuration was evaluated for each ACE event. We measured the percentage of correct argument mentions extracted out of all correct argument mentions annotated for the event (recall) and out of all argument mentions extracted (precision), and F1, their harmonic average. We report macro averages over the 26 tested event types.

Table 7.6 summarizes the results for the different configurations. As expected, matching only the seed templates yields the highest precision but lowest recall. Using the standard WordNet configuration actually decreases overall F1 performance: though recall increases relatively by 35%, thanks to WordNet expansions, F1 is penalized by a sharp relative drop in precision (by 59%). The main reasons for the decline in precision are incorrect rule application due to mismatched role and frame annotation, application of rules involving infrequent word senses and other rule applications under invalid contexts (this is further elaborated in Section 7.5.2).

Adding functional role and frame information increases precision by %10 with a smaller drop in recall by %5 (mainly due to parse errors). This shows that the
theoretical benefit of this information to entailment rule representation is reflected in real rule application scenarios (see also the analysis in Section 7.5.4). When our rule validation approach is applied to standard WordNet expansions, a much higher precision is achieved with only a small decline in recall. This shows that our corpus-based filtering method manages to avoid many of the noisy rules for rare senses, while maintaining those that are frequently involved in inference.

Finally, our main result shows that adding argument mapping improves performance substantially. AmWN achieves a much higher recall than WordNet. Recall increases relatively by 26% over validated WordNet, and by 54% over the no-rules baseline. Furthermore, precision drops only slightly, by 6%, compared to validated WordNet. This shows that argument mapping increases WordNet’s graph connectivity, while our rule-validation method maintains almost the same precision for many more generated rules. The improvement in overall F1 performance is statistically significant compared to all other configurations, according to the two-sided Wilcoxon signed rank test at the level of 0.01 (Wilcoxon, 1945).

Error Analysis

We manually analyzed the reasons for false positives (incorrect extractions) and false negatives (missed extractions) of AmWN by sampling 300 extractions of each type.

From the false positives analysis (Table 7.7) we see that practically all generated rules are correct (99.4%), that is, they would be valid in some contexts. This is in line with the high accuracy of edges as measured by our manual analysis of AmWN. Almost all errors come from matching errors (including parse errors) and context mismatches, due to our limited IE implementation. The only two incorrect rules sampled were due to an incorrect Nomlex entry and a WordNet synset that should have been split into two separate senses.
7.5. APPLICATION-ORIENTED EVALUATIONS

<table>
<thead>
<tr>
<th>Reason</th>
<th>% mentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context mismatch</td>
<td>57.2</td>
</tr>
<tr>
<td>Match error</td>
<td>33.6</td>
</tr>
<tr>
<td>Errors in gold-standard annotation</td>
<td>8.6</td>
</tr>
<tr>
<td>Incorrect rule learned</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Table 7.7: Distribution of reasons for false positives (incorrect argument extractions).

<table>
<thead>
<tr>
<th>Reason</th>
<th>% mentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule not learned</td>
<td>67.7</td>
</tr>
<tr>
<td>Match error</td>
<td>18.0</td>
</tr>
<tr>
<td>Discourse analysis needed</td>
<td>12.0</td>
</tr>
<tr>
<td>Argument is predicative</td>
<td>1.3</td>
</tr>
<tr>
<td>Errors in gold-standard annotation</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Table 7.8: Distribution of reasons for false negatives (missed argument mentions).

Context mismatches, which constitute the majority of errors (57.2%), occur when the entailing template of a rule is matched in inappropriate contexts. This occurs typically when the match is against another sense of the predicate, or when an argument is not of the requested type (e.g. “The Enron sentence” vs. “A one month sentence”).

Table 7.8 presents the false negatives analysis. Most missed extractions are due to rules that were not learned (67.7%). From further analysis, we found that 10% of the misses of this type are due to rules that are generated by AmWN but filtered out by one of our filtering methods (Section 7.4). Yet, the vast majority of these misses (90%) are because the information needed for generating these rules does not exist in WordNet in the first place. Such rules mainly involve complex templates (‘file a lawsuit ⇔ sue’) and inference rules that are not synonyms/hypernyms (‘execute ⇒ sentence’), which are not widely annotated in WordNet.

12% of the arguments cannot be extracted by rules alone, due to required discourse
Table 7.9 presents ablations tests that assess the marginal contribution of each AmWN component. Nominal-verb and verb-verb mappings contribute to the graph connectivity, hence the recall reduction when they are removed. However, adjectives hardly added to recall, at least in the ACE experiment. Further analysis pointed some of the blame on parse errors and some on missing rules. At large, however, it seems that adjectives do not play an important role in expressing predicate variations of ACE events.

Complementary to recall components, rule filtering improves precision. When
removing the corpus-based rule-validation, recall increases relatively by 6% but precision drops relatively by 30%, emphasizing again the benefit of this type of rule filtering. We note that AmWN’s F1 without validation is still better than WordNet without validation, showing again the advantage of adding argument mappings.

Allowing sense drifting hurts precision, a relative drop of 22%. Yet, recall increases relatively by 8%, indicating that some verb synsets, connected via a shared nominal, entail each other even though they are not connected directly. For example, ‘found \( X \leftrightarrow create \ X \)’ was generated only via the shared nominal founding. Future work may attempt to apply AmWN to a coarse-grained set of WordNet synsets (Palmer, Trang Dang, and Fellbaum, 2007) as a more thorough candidate solution to sense drifting.

7.5.3 The RTE Evaluation

Experimental Setting

We performed experiments on the task of Recognising Textual Entailment (RTE), using the RTE-2, RTE-3 and RTE-4 datasets (Bar-Haim et al., 2006; Giampiccolo et al., 2007; Giampiccolo et al., 2008), which are standard benchmarks for the task. These datasets consist of \((text, hypothesis)\) pairs, which need to be classified as entailing/non entailing. An example for each pair type is presented in Figure 1.1 on page 2. The RTE-2 and RTE-3 test sets contain 800 test examples each and RTE-4 contains 1000 test examples. The examples in all datasets are evenly distributed between positive (entailing) and negative (non-entailing) examples. RTE-2 and RTE-3 also provide development sets in the same format, containing 800 examples each. RTE-4 does not provide a development set, only a test set.

In this experiment we used the state-of-the-art RTE system presented in (Bar-Haim et al., 2008). This system operates in two primary stages. In the first \textit{Inference}
stage, entailment rules taken from various resources are applied to the text $T$, aiming
to generate a set of entailed sentences (consequences), which might be more similar to
the hypothesis $H$. In the second **Classification** stage, a set of features is extracted
from the generated set of consequences and from $H$ and fed into an SVM classifier,
which determines entailment. The classification setting and its features are quite
typical for the RTE literature. They include lexical and structural measures for the
coverage of $H$ by the set of consequences, where high coverage is assumed to correlate
with entailment, as well as features aiming to detect inconsistencies between the set
of consequences and $H$, such as incompatible arguments for the same predicate or
incompatible verb polarity, which would indicate non-entailment.

In the Inference stage, various resources for entailment rules were utilized: (1)
lexical rules such as ‘*Janis Joplin* ⇒ *singer*’, derived from Wikipedia based on both
its metadata (e.g. links and redirects) and text definitions, using patterns such as ‘$X$
*is a* $Y$’ (Shnarch, Barak, and Dagan, 2009); (2) lexical rules extracted from WordNet
based on synonyms and hypernyms; (3) binary rules, such as ‘$X$ *is fond of* $Y$ ⇒ $X$
*likes* $Y$’, learned by the DIRT algorithm (Lin and Pantel, 2001); (4) generic rules that
capture syntactic variations such as passive forms and conjunctions (Bar-Haim et al.,
2008); (5) rules generated from our implementation of AmWN. In this experiment,
three iterations of applying generic rules were performed, followed by a single iteration
of all other rule types. The system was trained over the RTE-2 development set for
testing the RTE-2 test set and over the RTE-3 development set for testing the RTE-3
and RTE-4 test sets.

**Results**

We tested two different rule-set configurations: (1) all rule resources except AmWN;
(2) all resources including AmWN. Each configuration was evaluated on the RTE-2,
Table 7.10: Accuracy(%) results for the different RTE Challenge datasets for the configurations where the inference system uses AmWN as a resource for rules and where it does not.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>without AmWN</th>
<th>with AmWN</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTE-2</td>
<td>59.375</td>
<td>60.75</td>
<td>2.3</td>
</tr>
<tr>
<td>RTE-3</td>
<td>65.375</td>
<td>66.125</td>
<td>1.1</td>
</tr>
<tr>
<td>RTE-4</td>
<td>60.8</td>
<td>61.3</td>
<td>0.8</td>
</tr>
</tbody>
</table>

RTE-3 and RTE-4 test sets. For each dataset we measured **Accuracy**, the percentage of correctly classified pairs. We applied AmWN in this experiment without rule filtering of rare senses, as sense mismatch between the text and the hypothesis does not arise as a problem in the RTE datasets. This idiosyncrasy is due to the construction procedure of the datasets: each pair was manually selected, and it seems that the annotators tended to choose wordings of the same topic/sense for the text/hypothesis pair.

Table 7.10 presents the results for the different datasets. The results show that adding AmWN as an additional resource for entailment rules constantly improves the performance of the system. Though the improvement is not large, it is in line with the magnitude of reported contribution of other resources for the RTE datasets (Iftene and Balahur-Dobrescu, 2007; Marsi, Krahmer, and Bosma, 2007; Bar-Haim et al., 2008; Clark and Harrison, 2008). This shows that in order to reach a good coverage of the hypotheses in RTE using entailment rules, different resources should be utilized in tandem. On the other hand, AmWN generated useful rules that were missing from the other resources, hence justifying its addition as a resource to RTE systems. Table 7.11 presents some examples for such rules. We remind that the performance of the RTE system without AmWN still includes WordNet as a resource. Thus the improvement in performance when using AmWN is due to the additional rules that can be generated from AmWN but not from standard utilization of WordNet.
Table 7.11: Examples of AmWN rules missing from other resources, which were applied during inference to the RTE-2 test set. For readability, binary rules are presented when two complementary unary rules were generated (when mapping was needed for two arguments of the same predicate in the hypothesis).

### 7.5.4 Additional Analyses

Subcategorization Frame and Functional Role Information in Practice

Though we propose to represent argument mappings as entailment rules augmented with functional roles and subcategorization frames, AmWN may be formulated in the same way without incorporating this additional information (denoted hereafter the role-frame-info). In such a variant, no distinction is made between different roles and different frames of the same predicate under a single synset id. Theoretically, adding this information is needed to avoid incorrect inference, as discussed in Section 7.2.1. Indeed, mapping edges within AmWN incorrectly connect nodes when the role-frame-info is ignored. However, in real applications, the negative affect of ignoring this information might be of a lesser scale. Furthermore, as there is no need to annotate texts with role-frame-info, errors that result from imperfect annotation do not occur.

To test the empirical difference between these two versions we implemented the AmWN framework without role-frame-info, denoted here AmWN-nf, and tested it on the ACE dataset. Table 7.12 presents the results of both implementations.
Table 7.12: Recall (R), Precision (P) and F1 results for the ACE dataset for AmWN and AmWN-nf.

<table>
<thead>
<tr>
<th></th>
<th>R (%)</th>
<th>P (%)</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmWN</td>
<td>20.8</td>
<td>43.9</td>
<td>24.2</td>
</tr>
<tr>
<td>AmWN-nf</td>
<td>22.1</td>
<td>38.4</td>
<td>24.0</td>
</tr>
</tbody>
</table>

The table we see that the impact of the role-frame-info is apparent. AmWN is notably more accurate - 5.5% higher precision - due to the additional role-frame-info. This is an empirical indication for the need to verify frame matching as part of template matching in texts. For example, in the *Demonstrate* event, the seed template ‘X_{subj} demonstrate\_{(intr.)}’ is also matched against transitive occurrences when frames are ignored. This result is in line with the similar evaluation performed in Section 7.5.2, showing that standard WordNet with role-frame-info has higher precision when compared to WordNet, which does not incorporate this additional information.

The downside of incorporating the role-frame-info is the increased complexity of text annotation. Missing subcategorization frame information, as well as parse errors, may result in unannotated or incorrectly annotated frames in texts. In addition, ambiguous cases result in unannotated texts as well. These annotation issues are reflected in the results: AmWN’s recall is lower by 1.3% compared to AmWN-nf. For example, some intransitive occurrences were mistakenly annotated as transitive, due to parse errors. Hence, these occurrences were missed when frame matching was required.

To sum up, we conclude that incorporating role-frame-info as part of entailment rule representation at the syntactic level is indeed a promising direction to improve rule application accuracy. While erroneous frame annotations do occur, we expect them to diminish following improved parsing technology and the inclusion of additional resources for subcategorization frame information.
<table>
<thead>
<tr>
<th>Configuration</th>
<th>R (%)</th>
<th>P (%)</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Rules</td>
<td>13.5</td>
<td>63.0</td>
<td>20.7</td>
</tr>
<tr>
<td>WordNet</td>
<td>18.3</td>
<td>32.2</td>
<td>17.8</td>
</tr>
<tr>
<td>AmWN</td>
<td>20.8</td>
<td>43.9</td>
<td>24.2</td>
</tr>
<tr>
<td>unary-DIRT - top 10 rules</td>
<td>23.9</td>
<td>24.5</td>
<td>19.7</td>
</tr>
<tr>
<td>unary-DIRT - top 50 rules</td>
<td>35.2</td>
<td>8.9</td>
<td>11.5</td>
</tr>
</tbody>
</table>

Table 7.13: Recall (R), Precision (P) and F1 results for the WordNet-related resources vs. the unary-DIRT resource, tested on the ACE dataset.

**Unsupervised Learning vs. WordNet**

In our current implementation of AmWN, we are utilizing an unsupervised approach, unary-DIRT, to identify argument mappings between verbs. To show the differences between utilizing this learning method directly as an entailment rule resource and utilizing it only to obtain argument mappings for known predicative entailments (via mapping edges), we also conducted experiments under the ACE setting in which the tested rule-base was learned by unary-DIRT. We experimented with two rule-bases, which contain the top 10 and the top 50 rules learned for every template occurring in the Reuters RCV1 corpus, respectively.

The results, presented in Table 7.13, show that unary-DIRT learns useful entailment rules that currently cannot be generated by WordNet relations. This is indicated by its recall, which is higher than that of AmWN. However, the rules learned by unary-DIRT are very noisy when used directly (not intersected with WordNet relations), and thus unary-DIRT suffers from low precision. We thus conclude, as we did before, that to further increase system recall these resources should be used in tandem, but some rule filtering should be provided to unary-DIRT first. Future work may enrich the AmWN graph with unary-DIRT rules missing from WordNet to further improve the utilization of the two resources.
7.6 Related Work

7.6.1 WordNet Extensions

Several works attempt to extend WordNet with additional lexical semantic information. Some works added missing words and terms to WordNet, as well as their hyponym relations, which connect these new words to the WordNet graph (Snow, Jurafsky, and Ng, 2006; Suchanek, Kasneci, and Weikum, 2007). These works discover lexical rules, specifically hyponym relations, which are substitutable relations, while our work focused on the non-substitutable relations in WordNet. In other works, the information in WordNet glosses was addressed. Moldovan and Rus (2001) parsed the gloss of each WordNet entry into a logical form, generating an inference rule from the defined term to its parsed gloss. More related to our work, Clark et al. (2008) also generated the same type of inference rules from WordNet glosses. Furthermore, they also labeled each WordNet noun-verb derivational relation with the relation’s semantic type, e.g. agent for employer, recipient for grantee and vehicle for cruiser (this was done by manual verification over an initial automatic guessing). This information, once publicly available, could further help us identify the predicative nominal synsets in WordNet, as well as some of the argument mappings related to them.

The only previous work we are aware of that enriches WordNet with argument mappings is (Novischi and Moldovan, 2006). This work utilizes VerbNet’s subcategorization frames to identify possible verb arguments. Argument mapping is provided only between verbs, ignoring relations between verbs and nouns or adjectives, which cannot be extracted from VerbNet. In (Novischi and Moldovan, 2006), Arguments are mapped based on thematic role names, such as Agent, Theme and Recipient, that are shared between frames of different verbs. However, the semantic interpretation of thematic roles is generally inconsistent across verbs (Lowe, Baker, and Fillmore,
1997; Kaisser and Webber, 2007), unlike syntactic functional roles, which are consistent and immediately derived from the parser output. For example, in VerbNet the subject role within the intransitive frame of *exercise* is labeled with the *Agent* thematic role, while within the intransitive frame of *move*, its label is *Theme*. Thus, the arguments of the WordNet entailment relation ‘*exercise* ⇒ *move*’ cannot be mapped using VerbNet. In another example, the thematic role of the subject functional role within both the transitive frame of *feed* (e.g. “*he* fed the lions”) and the intransitive frame of *eat* (e.g. “*the lions* ate”) is *Agent*. Thus, the incorrect rule ‘\(X_{\text{subj}} \text{feed}_{(\text{trans})}\) \(\Rightarrow X_{\text{subj}} \text{eat}_{(\text{intrans})}\)’ is extracted from VerbNet, while the correct rule ‘\(\text{feed}_{(\text{trans})} X_{\text{obj}} \Rightarrow X_{\text{subj}} \text{eat}_{(\text{intrans})}\)’ is missed. Instead, AmWN discovers these mappings from corpus statistics, offering a more accurate approach (as analyzed in Section 7.3.7 and Section 7.5.2).

### 7.6.2 Frame Semantics

A frame semantics approach for argument mapping between predicates is proposed by the FrameNet project (Baker, Fillmore, and Lowe, 1998). In this representation, arguments are labeled by their frame-semantic name, such as *Buyer* and *Goods*. This approach is similar to the VerbNet-based thematic-role approach presented above, yet it is more accurate due to the nature of frame semantics. As in the case of thematic-roles, argument mapping between two predicates is provided via arguments that have the same frame role label for both predicates.

To provide argument mappings between predicates within the frame-semantics approach, first a resource containing semantic argument description for each predicate is required. The FrameNet project is the only such resource existing to date. However, FrameNet is manually constructed and currently covers much fewer predicates and relations than WordNet. We conducted an upper-bound analysis of FrameNet’s
Table 7.14: FrameNet’s upper bound for covering WordNet’s entailment relations.

<table>
<thead>
<tr>
<th>Relation Type</th>
<th>Coverage Potential (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Verb-Verb</td>
<td>45.5</td>
</tr>
<tr>
<td>Noun-Verb</td>
<td>12.8</td>
</tr>
<tr>
<td>Adjective-Verb</td>
<td>5.0</td>
</tr>
<tr>
<td>Total</td>
<td>13.5</td>
</tr>
</tbody>
</table>

coverage for the WordNet relations recognized as entailing by our AmWN implementation. We marked a relation as covered by FrameNet if at least one shared argument is found between the two related predicates, where the argument list is taken from all the frames in which each predicate occurs. Table 7.14 presents the summary of this analysis. The results show that currently, FrameNet only covers 13.5% of the entailment relations in WordNet. It has the potential to cover only about half of WordNet’s verb-verb relations (45.5%), and its coverage potential for morphologically related predicates is even poorer.

Finally, to be able to apply entailment rules whose arguments are annotated with frame-semantic labels, a frame-semantic parser is required (similar to the syntactic role-frame annotation required in our approach). However, current frame-semantic parsers are less robust than syntactic parsers, presently hindering the utilization of this approach in applications (Burchardt and Pennacchiotti, 2008).

### 7.6.3 Other Related Works

Nomlex argument mapping patterns similar to ours were derived for IE in (Meyers et al., 1998). However, in this work Nomlex was used as a stand-alone resource and was not integrated with any additional information, such as WordNet. Obradovic et al. (2004) proposed to validate WordNet synset members based on corpus statistics, but did not validate relations across synsets.
7.7 Conclusions

In this chapter we presented Argument-mapped WordNet (AmWN), a novel framework for augmenting WordNet with argument mappings at the syntactic representation level. With AmWN, non-substitutable WordNet relations can also be utilized correctly when argument positions need to be traced, increasing the coverage of WordNet-based inference. The standard entailment rule representation is augmented in our work with functional roles and subcategorization frames, shown to be a feasible extension needed for correct rule application in general.

Our implementation of AmWN populates WordNet with mappings based on combining manual and corpus-based resources. It covers a broader range of relations compared to prior work and yields more accurate mappings. To improve the resource’s quality, we further introduced a novel corpus-based validation mechanism, avoiding rules for infrequent senses. Our experiments show that AmWN substantially improves standard WordNet-based inference.

Future work may incorporate resources such as VerbNet for additional subcategorization frame information that is missing from WordNet. In addition, text annotation for frames and functional roles may be enhanced, for example based on noun-compound disambiguation (Lapata and Lascarides, 2003).

The proposed augmentation of lexical-syntactic templates with subcategorization frame and functional role information is a general extension, beyond the rules generated by WordNet. This extended representation of entailment rules can be utilized within other learning algorithms that learn rules at the syntactic level, as long as the sources from which the rules are learned can be annotated with frame and functional role information. For example, algorithms that learn entailment rules from either a comparable or a general corpus, such as DIRT, could first annotate the learning corpus with subcategorization frame information. Then, each template that is extracted
from a specific occurrence in the corpus is augmented with the specific role-frame-info annotated for this occurrence. Thus, rules will be learned between augmented templates, distinguishing between the different frames of a predicate.
8.1 Introduction

Applied semantic inference is typically concerned with inferring a target meaning from a given text. For example, to answer "Who wrote Idomeneo?", Question Answering (QA) systems need to infer the hypothesis "h: Mozart wrote Idomeneo" from a given text "t: Mozart composed Idomeneo". As discussed in Section 2.1, a typical applied inference operation is matching. Sometimes, h can be directly matched in t (in the example above, if the given sentence would be literally "Mozart wrote Idomeneo"). Generally, indirect matching is needed using entailment rules. In our example, "Mozart wrote Idomeneo" can be inferred using the rule ‘X compose Y ⇒ X write Y’.

A common practice is to try matching the structure of h, or of the LHS of a rule r, within t. However, when the matching process considers only the structure of hypotheses, texts and rules it may result in incorrect inference due to contextual mismatches. For example, an IE system may identify mentions of public protests using the template hypothesis h: ‘X demonstrate’. However, h should not be matched in t: “Engineers demonstrated the new system”, due to a mismatch between the intended
8.1. INTRODUCTION

sense of demonstrate in $h$ and its sense in $t$. Similarly, when looking for mentions of physical attack using the hypothesis ‘$X$ attack $Y$’, we should not utilize the rule $r$: ‘$X$ accuse $Y$ ⇒ $X$ attack $Y$’, due to a mismatch between the context of verbal attack in $r$ and the intended physical attack in $h$. Finally, the rule $r$: ‘$X$ produce $Y$ ⇒ $X$ lay $Y$’ (applicable when $X$ refers to poultry and $Y$ to eggs) should not be matched in $t$: “Bugatti produce the fastest cars”, due to a mismatch between the meanings of produce in $r$ and $t$.

Across all our error analyses in this thesis, applying correct rules within invalid contexts keeps showing up as a major contributor to incorrect rule application. Such errors occur when applying rules originating from both unsupervised approaches such as distributional similarity algorithms and from AmWN, even after filtering out rules for rare word senses (see Sections 4.3.3, 6.4.3 and 7.5.2). Overall, such incorrect inferences may be avoided by considering contextual information for $t$, $h$ and $r$ during their matching process.

Context matching at inference time was mostly approached in an application-specific manner (Harabagiu, Maiorano, and Pașca, 2003; Patwardhan and Riloff, 2007). Recently, some generic methods were proposed to handle context-sensitive inference (Dagan et al., 2006; Pantel et al., 2007; Downey, Schoenmackers, and Etzioni, 2007; Connor and Roth, 2007), but these usually treat only a single aspect of context matching (see description of prior work in Section 2.4). In this chapter, we propose a comprehensive framework for handling various contextual considerations, termed Contextual Preferences. It extends and generalizes previous work, defining the needed contextual components and their relationships. We also present and implement concrete representation models and unsupervised matching methods for these components. While our presentation focuses on semantic inference using lexical-syntactic structures, the proposed framework and models seem suitable for other common types of representations as well.
8.2 The Contextual Preferences Framework

We propose the Contextual Preferences (CP) framework for addressing context at inference time. In this framework, the representation of an object $z$, where $z$ may be a text, a template or an entailment rule, is enriched with contextual information denoted $cp(z)$. This information helps constraining or disambiguating the meaning of $z$, and is used to validate proper matching between pairs of objects.

We consider two components within $cp(z)$: (a) a representation for the global ("topical") context in which $z$ typically occurs, denoted $cp_g(z)$; (b) a representation for the preferences and constraints ("hard" preferences) on the possible terms that can instantiate variables within $z$, denoted $cp_v(z)$. For example, to represent physical attacks, $cp_g(’X$ attack $Y’) may specify that the template should occur in sentences related to war or injuries. In another example, $cp_v(’X$ produce $Y \Rightarrow X$ lay $Y’) may specify that $X$’s instantiations should be similar to chicken or duck.

Contextual Preferences are used when entailment is assessed between a text $t$ and a hypothesis $h$, either directly or by utilizing an entailment-rule $r$. On top of structural matching, we now require that the contextual preferences of the participants in the inference will also match. When $h$ is directly matched in $t$, we require that each component in $cp(h)$ will be matched with its counterpart in $cp(t)$. When $r$ is utilized, we additionally require that $cp(r)$ will be matched with both $cp(t)$ and $cp(h)$. 
8.3. CONTEXTUAL PREFERENCES MODELS

Figure 8.1 summarizes the matching relationships between the CP components of \( h \), \( t \) and \( r \).

Like Textual Entailment inference, Contextual Preferences matching is directional. When matching \( h \) with \( t \) we require that the global context preferences specified by \( cp_g(h) \) would subsume those induced by \( cp_g(t) \), and that the instantiations of \( h \)'s variables in \( t \) would adhere to the preferences in \( cp_v(h) \) (since \( t \) should entail \( h \)), but not necessarily vice versa. For example, if the preferred global context of a hypothesis is *sports*, it may match a text that discusses a more specific sports topic, such as *basketball*.

To implement the CP framework, concrete models are needed for each component, specifying its representation, how it is constructed, and an appropriate matching procedure. Section 8.3 describes the specific CP models that were implemented in this thesis.

The CP framework provides a generic view of contextual modeling in applied semantic inference. Mapping from a specific application to the generic framework follows the mappings assumed in the Textual Entailment paradigm. For example, in QA the hypothesis to be proved corresponds to the affirmative template derived from the question (e.g. \( h: \) ‘\( X \) invented the PC’ for “Who invented the PC?”). Thus, \( cp_g(h) \) can be constructed with respect to the question’s focus while \( cp_v(h) \) may be generated from the expected answer type (Moldovan et al., 2000; Harabagiu, Maiorano, and Pașca, 2003). Construction of hypotheses’ CP for IE is demonstrated in Section 8.4.

8.3 Contextual Preferences Models

This section presents the current models that we implemented for the various components of the CP framework. For each component type we describe its representation, how it is constructed, and a corresponding unsupervised match score. Finally, the
different component scores are combined to yield an overall match score, which is used in our experiments to rank inference instances by the likelihood of their validity. Our goal in this work is to cover the entire scope of the CP framework by including specific models that were proposed in previous work, where available, and elsewhere propose initial models to complete the CP scope.

8.3.1 Contextual Preferences for Global Context

To represent the global context of an object \( z \) we utilize Latent Semantic Analysis (LSA) (Deerwester et al., 1990), a well-known method for representing the contextual-usage of words based on corpus statistics. We use LSA analysis of the BNC corpus\(^1\), in which every term is represented by a normalized vector of the top 100 SVD dimensions, as described in (Gliozzo, 2005).

To construct \( cp_g(z) \) we first collect a set of terms that are representative for the preferred general context of \( z \). Then, the (single) vector which is the sum of the LSA vectors of the representative terms becomes the representation of \( cp_g(z) \). This LSA vector captures the “average” typical contexts in which the representative terms occur.

The set of representative terms for a text \( t \) consists of all the nouns and verbs in it, represented by their lemma and part of speech. For a rule \( r: \text{LHS} \Rightarrow \text{RHS} \), the representative terms are the words appearing in \( \text{LHS} \) and in \( \text{RHS} \). For example, the representative terms for ‘\( X \) divorce \( Y \) ⇒ \( X \) marry \( Y \)’ are \{\text{divorce:v, marry:v}\}. As mentioned earlier, construction of hypotheses and their contextual preferences depends on the application at hand. In our experiments these are defined manually, as described in Section 8.4, derived from the manual definitions of target meanings in the IE data.

\(^1\)http://www.natcorp.ox.ac.uk/
The score of matching the \( cp_g \) components of two objects, denoted by \( m_g(\cdot, \cdot) \), is the Cosine similarity of their LSA vectors. Negative values are set to 0.

### 8.3.2 Contextual Preferences for Variables

**Representation**

For comparison with prior work, we follow (Pantel et al., 2007) and represent preferences for variable instantiations using a distributional approach, and in addition incorporate a standard specification of named-entity types. Thus, \( cp_v \) is represented by two lists. The first list, denoted \( cp_{v:e} \), contains examples for valid instantiations of that variable. For example, \( cp_{v:e}(X \; \text{kill} \; Y \rightarrow Y \; \text{die of} \; X) \) may be \([X: \{\text{snakebite, disease}\}, Y: \{\text{man, patient}\}]\). The second list, denoted \( cp_{v:n} \), contains the variable’s preferred named-entity types (if any). For example, \( cp_{v:n}(X \; \text{born in} \; Y) \) may be \([X: \{\text{Person}\}, Y: \{\text{Location}\}]\). We denote \( cp_{v:e}(z)[j] \) and \( cp_{v:n}(z)[j] \) as the lists for a specific variable \( j \) of the object \( z \).

For a text \( t \), in which a template \( p \) is matched, the preference \( cp_{v:e}(t) \) for each template variable is simply its instantiation in \( t \). For example, when ‘\( X \; \text{eat} \; Y \)’ is matched in \( t \): “Many Americans eat fish regularly”, we construct \( cp_{v:e}(t) = [X: \{\text{Many Americans}\}, Y: \{\text{fish}\}] \). Similarly, \( cp_{v:n}(t) \) for each variable is the named-entity type of its instantiation in \( t \) (if it is a named entity). We identify entity types using the default Lingpipe\(^1\) Named-Entity Recognizer (NER), which recognizes the types Location, Person and Organization. In the above example, \( cp_{v:n}(t)[X] \) would be \( \{\text{Person}\} \).

For a rule \( r: LHS \rightarrow RHS \), we automatically add to \( cp_{v:e}(r) \) all the variable instantiations that were found common for both \( LHS \) and \( RHS \) in a corpus (see Section 8.4), as in (Pantel et al., 2007; Pennacchiotti et al., 2007). To construct

---

\(^1\)http://www.alias-i.com/lingpipe/
$cp_{v,n}(r)$, we currently use a simple approach where each individual term in $cp_{v,e}(r)$ is analyzed by the NER system, and its type (if any) is added to $cp_{v,n}(r)$.

For a template hypothesis, we currently represent $cp_v(h)$ only by its list of preferred named-entity types, $cp_{v,n}$. Similarly to $cp_g(h)$, the preferred types for each template variable were adapted from those defined in our IE data (see Section 8.4).

To allow compatible comparisons with previous work (see Sections 8.5 and 2.4), we utilize in this experiment only $cp_{v,e}$ when matching between $cp_v(r)$ and $cp_v(t)$, as only this representation was examined in prior work on context-sensitive rule applications. $cp_{v,n}$ is utilized for context matches involving $cp_v(h)$. We denote the score of matching two $cp_v$ components by $m_v(\cdot, \cdot)$.

**Matching $cp_{v,e}$**

Our primary matching method is based on replicating the best-performing method reported in (Pantel et al., 2007), which utilizes the CBC distributional word clustering algorithm (Pantel and Lin, 2002). In short, this method extends each $cp_{v,e}$ list with CBC clusters that contain at least one term in the list, scoring them according to their “relevancy”. The score of matching two $cp_{v,e}$ lists, denoted here $S_{CBC}(\cdot, \cdot)$, is the score of the highest scoring member that appears in both lists (see also Section 2.4, page 25, for the algorithm description).

We applied the final binary match score presented in (Pantel et al., 2007), denoted here *binaryCBC*: $m_{we}(r, t)$ is 1 if $S_{CBC}(r, t)$ is above a threshold and 0 otherwise. As a more natural ranking method, we also utilize $S_{CBC}$ directly, denoted *rankedCBC*, having $m_{we}(r, t) = S_{CBC}(r, t)$.

In addition, we tried a simpler method that directly compares the terms in two $cp_{v,e}$ lists, utilizing the commonly-used term similarity metric of (Lin, 1998a). This method, denoted *LIN*, uses the same raw distributional data as CBC but computes only pair-wise similarities, without any clustering phase. We calculated the scores of
the 1000 most similar terms for every term in the Reuters RVC1 corpus. Then, a
directional similarity of term \(a\) to term \(b\), \(s(a, b)\), is set to be their similarity score
if \(a\) is in \(b\)'s 1000 most similar terms and 0 otherwise. The final score of matching
\(r\) with \(t\) is determined by a nearest-neighbor approach, as the score of the most
similar pair of terms in the corresponding two lists of the same variable:
\[
m_{\text{v:e}}(r, t) = \max_{j \in \text{vars}(r)} \max_{a \in \text{cp}_{\text{v:e}}(r)[j]} \max_{b \in \text{cp}_{\text{v:e}}(t)[j]} \left[ s(a, b) \right].
\]

**Matching \(cp_{v:n}\)**

We use a simple scoring mechanism for comparing between two named-entity types
\(a\) and \(b\), \(s(a, b)\): 1 for identical types and 0.8 otherwise.

A variable \(j\) has a single preferred entity type in \(cp_{v:n}(t)[j]\), the type of its instan-
tiation in \(t\). However, it can have several preferred types for \(h\). When matching \(h\)
with \(t\), \(j\)'s match score is that of its highest scoring type, and the final score is the
product of all variable scores:
\[
m_{\text{v:n}}(h, t) = \prod_{j \in \text{vars}(h)} \left( \max_{a \in \text{cp}_{v:n}(h)[j]} \left[ s(a, \text{cp}_{v:n}(t)[j]) \right] \right).
\]

Variable \(j\) may also have several types in \(r\), the types of the common arguments in
\(cp_{v:e}(r)\). When matching \(h\) with \(r\), \(s(a, \text{cp}_{v:n}(t)[j])\) is replaced with the average score
for \(a\) and each type in \(\text{cp}_{v:n}(r)[j]\).

### 8.3.3 Overall Score for a Match

A final score for a given match, denoted \(allCP\), is obtained by the product of all six
matching scores of the various CP components (multiplying by 1 if a component score
is missing). The six scores are the results of matching any of the two components of
\(h\), \(t\) and \(r\): \(m_{g}(h, t)\), \(m_{v}(h, t)\), \(m_{g}(h, r)\), \(m_{v}(h, r)\), \(m_{g}(r, t)\) and \(m_{v}(r, t)\) (as specified
above, \(m_{v}(r, t)\) is based on matching \(cp_{v:e}\) while \(m_{v}(h, r)\) and \(m_{v}(h, t)\) are based on
matching \(cp_{v:n}\)). We use \textit{rankedCBC} for calculating \(m_{v}(r, t)\).

Unlike previous work (e.g. (Pantel et al., 2007)), we also utilize the \textit{prior} score
of a rule \( r \), which is provided by the rule-learning algorithm (see next section). We denote by \( \text{allCP} + \text{pr} \) the final match score obtained by the product of the \( \text{allCP} \) score with the prior score of the matched rule.

8.4 Experimental Settings

As in previous chapters, we follow the ACE experimental setup in our experiment (Section 4.5). In this experiment we tested the contribution of the CP framework to binary rule-bases. For each event type we manually created a small set of binary seed templates (template hypotheses) that correspond to the given event predicate. For example, for the \text{Injure} event, the set of hypotheses included ‘\( A \text{ injure} V \)’ and ‘\( \text{injure} V \text{ in } T \)’.

The Contextual Preferences for each template hypothesis \( h \) were constructed manually: the named-entity types for \( \text{cp}_{\text{vn}}(h) \) were set by adapting the entity types given in the guidelines to the types supported by the Lingpipe NER (described in Section 8.3.2). \( \text{cp}_{\text{g}}(h) \) was generated from a short list of nouns and verbs that were extracted from the verbal event definition in the ACE guidelines. For \text{Injure}, this list included \{\text{injure}:v, \text{injury}:n, \text{wound}:v\}. This assumes that when writing down an event definition the user would also specify such representative keywords.

Entailment-rules for a given \( h \) (rules in which \( \text{RHS} \) is equal to \( h \) ) were learned by the binary DIRT algorithm (Lin and Pantel, 2001), which also produces a quality score for each rule. We used a canonized version of DIRT (Chapter 5) on the Reuters corpus parsed by Minipar. Each rule’s arguments for \( \text{cp}_{\text{v}}(r) \) were also collected from this corpus.

We assessed the CP framework by its ability to correctly rank, for each predicate (event), all the candidate entailing mentions that are found for it in the test corpus. Such ranking evaluation is suitable for unsupervised settings, with a perfect ranking
8.5 Results and Analysis

We experimented with three rule setups over the ACE dataset, in order to measure the contribution of the CP framework. In the first setup no rules are used, applying only direct matches of template hypotheses to identify event mentions. In the other two setups we also utilized DIRT’s top 50 or 100 rules for each hypothesis.

A match is considered correct when all matched arguments are extracted correctly according to their annotated event roles. This main measurement is denoted All. As an additional measurement, denoted Any, we consider a match as correct if at least one argument is extracted correctly.

Once event matches are extracted, we first measure for each event its Recall,
the number of correct mentions identified out of all annotated event mentions\(^1\) and \textbf{Precision}, the number of correct matches out of all extracted candidate matches. These figures quantify the baseline performance of the DIRT rule set used. To assess our ranking quality, we measure for each event the commonly used \textbf{Average Precision (AP)} measure (Voorhees and Harmann, 1998), which is the area under the non-interpolated recall-precision curve, while considering for each setup all correct extracted matches as 100\% Recall. Overall, we report \textbf{Mean Average Precision (MAP)}, macro average Precision and macro average Recall over the ACE events. Tables 8.1 and 8.2 summarize the main results of our experiments.

Examining Recall, we see that it increases substantially when rules are applied: by more than 100\% for the top 50 rules, and by about 150\% for the top 100, showing the benefit of entailment-rules to covering language variability. The difference between \textit{All} and \textit{Any} results shows that about 65\% of the rules that correctly match one argument also match correctly both arguments.

We use two baselines for measuring the CP ranking contribution: Precision, which corresponds to the expected MAP of random ranking, and MAP of ranking using the prior rule score provided by DIRT. Without rules, the baseline \textit{All} Precision is 34.1\%, showing that even the manually constructed hypotheses, which correspond directly to the event predicate, extract event mentions with limited accuracy when context is ignored. When rules are applied, Precision is very low. But ranking is considerably improved using only the prior score (from 1.4\% to 22.7\% for 50 rules), showing that the prior is an informative indicator for valid matches.

Our main result is that the \textit{allCP} and \textit{allCP+pr} methods rank matches statistically significantly better than the baselines in all setups (according to the Wilcoxon double-sided signed-ranks test at the level of 0.01 (Wilcoxon, 1945)). In the \textit{All} setup,

\(^1\text{For Recall, we ignored mentions with less than two arguments, as they cannot be correctly matched by binary templates.}\)
8.5. RESULTS AND ANALYSIS

ranking is improved by 70% for direct matching (Table 8.1). When entailment-rules are also utilized, prior-only ranking is improved by about 35% and 50% when using \( \text{allCP} \) and \( \text{allCP+pr} \), respectively (Table 8.2). Figure 8.2 presents the average Recall-Precision curve of the ‘50 Rules, All’ setup for applying \( \text{allCP} \) or \( \text{allCP+pr} \), compared to prior-only ranking baseline (other setups behave similarly). The improvement in ranking is evident: the drop in precision is significantly slower when CP is used. The behavior of CP with and without the prior is largely the same up to 50% Recall, but later on our implemented CP models are noisier and should be combined with the prior rule score.

Templates are incorrectly matched for several reasons. First, there are context mismatches which are not scored sufficiently low by our models. Another main cause is incorrect learned rules in which \( \text{LHS} \) and \( \text{RHS} \) are topically related, e.g. ‘\( X \) convict \( Y \) ⇒ \( X \) arrest \( Y \)’, or rules that are used in the wrong entailment direction, e.g. ‘\( X \) marry \( Y \) ⇒ \( X \) divorce \( Y \)’ (DIRT does not learn rule direction). As such rules
Table 8.3: MAP(%), under the ‘50 rules, All’ setup, when adding component match scores to Precision (P) or prior-only MAP baselines, and when ranking with allCP or allCP+pr methods but ignoring that component scores.

<table>
<thead>
<tr>
<th></th>
<th>Addition To Prior P</th>
<th>Ablation From allCP</th>
<th>Ablation From allCP+pr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>1.4</td>
<td>22.7</td>
<td>30.6</td>
</tr>
<tr>
<td>( cp_g(h, t) )</td>
<td>*10.4</td>
<td>*35.4</td>
<td>32.4</td>
</tr>
<tr>
<td>( cp_v(h, t) )</td>
<td>*11.0</td>
<td>29.9</td>
<td>27.6</td>
</tr>
<tr>
<td>( cp(h, t) )</td>
<td>*8.9</td>
<td>*37.5</td>
<td>28.6</td>
</tr>
<tr>
<td>( cp_g(r, t) )</td>
<td>*4.2</td>
<td>*30.6</td>
<td>32.5</td>
</tr>
<tr>
<td>( cp_v(r, t) )</td>
<td>*21.7</td>
<td>21.9</td>
<td>*12.9</td>
</tr>
<tr>
<td>( cp(r, t) )</td>
<td>*26.0</td>
<td>*29.6</td>
<td>*17.9</td>
</tr>
<tr>
<td>( cp_g(h, r) )</td>
<td>*8.1</td>
<td>22.4</td>
<td>31.9</td>
</tr>
<tr>
<td>( cp_v(h, r) )</td>
<td>*10.7</td>
<td>22.7</td>
<td>*27.9</td>
</tr>
<tr>
<td>( cp(h, r) )</td>
<td>*16.5</td>
<td>22.4</td>
<td>*29.2</td>
</tr>
<tr>
<td>( cp_g(h, r, t) )</td>
<td>*7.7</td>
<td>*30.2</td>
<td>*27.5</td>
</tr>
<tr>
<td>( cp_v(h, r, t) )</td>
<td>*27.5</td>
<td>29.2</td>
<td>*7.7</td>
</tr>
</tbody>
</table>

* Indicates statistically significant changes compared to the baseline, according to the Wilcoxon test at the level of 0.01.

Table 8.3 displays the contribution of different CP components to ranking, when adding only that component’s match score to the baselines, and under ablation tests, when using all CP component scores except the tested component, with or without the prior.

As it turns out, matching \( h \) with \( t \) (i.e. \( cp(h, t) \)), which combines \( cp_g(h, t) \) and
8.5. RESULTS AND ANALYSIS

cp_v(h, t)) is most useful. With our current models, using only cp(h, t) along with
the prior, while ignoring cp(r), achieves the highest score in the table. The strong
impact of matching h and t’s preferences is also evident in Table 8.1, where ranking
based on either cp_g or cp_v substantially improves precision, while their combination
provides the best ranking. These results indicate that the two CP components capture
complementary information and both are needed to assess the correctness of a match.

When ignoring the prior rule score, cp(r, t) is the major contributor over the
baseline Precision. For cp_v(r, t), this is in sync with the result in (Pantel et al.,
2007), which is based on this single model without utilizing prior rule scores. On the
other hand, cp_v(r, t) does not improve the ranking when the prior is used, suggesting
that this contextual model for the rule’s variables is not stronger than the context-
insensitive prior rule score. Furthermore, relative to this cp_v(r, t) model from (Pantel
et al., 2007), our combined allCP model, with or without the prior (first row of
Table 8.2), obtains statistically significantly better ranking (at the level of 0.01).

Comparing between the algorithms for matching cp_v (Section 8.3.2) we found that
while rankedCBC is statistically significantly better than binaryCBC, rankedCBC
and LIN generally achieve the same results. When considering the tradeoffs between
the two, LIN is based on a much simpler learning algorithm while CBC’s output is
more compact and allows faster CP matches.

Currently, some models do not improve the results when the prior is used. Yet,
we would like to further weaken the dependency on the prior score, since it is biased
towards frequent contexts. We aim to properly identify also infrequent contexts (or
meanings) at inference time, which may be achieved by better CP models. More
generally, when used on top of all other components, some of the models slightly
degrade performance, as can be seen by those figures in the ablation tests which are
higher than the corresponding baseline. However, due to their different roles, each
of the matching components might capture some unique preferences. For example,
\(cp(h, r)\) should be useful to filter out rules that do not match the intended meaning of the given \(h\). Overall, this suggests that future research for better models should aim to obtain a marginal improvement by each component.

### 8.6 Conclusions

In this chapter we presented the Contextual Preferences (CP) framework for assessing the validity of inferences in context. CP enriches the representation of textual objects with typical contextual information that constrains or disambiguates their meaning, and provides matching functions that compare the preferences of objects involved in the inference. Experiments with our implemented CP models, over real-world IE data, show significant improvements relative to baselines and some previous work.

In future research we plan to investigate improved models for representing and matching CP, and to extend the experiments to additional applied datasets. We also suggest applying this framework to lexical inference rules, for which it seems directly applicable.
Chapter 9

Conclusions

Entailment rules between predicates are an important type of knowledge required within many NLP applications. Yet, for such knowledge to be useful, knowledge-bases with vast coverage of predicate variability are required. We thus have set the construction of such large-scale knowledge-bases as the main research goal of this thesis.

Analysis of state-of-the-art approaches to entailment rule learning and application revealed several key issues that needed to be addressed. We first looked at the way entailment rule-bases are evaluated, which is critical for understanding where the limitations of current algorithms lie and whether new approaches indeed improve over current state-of-the-art performance. We showed that the main prior evaluation scheme for entailment rules, manually assessing the validity of each rule, is inadequate. As alternatives, we proposed two novel evaluation schemes: an instance-based
approach for human evaluation and an application-based scheme. Using our proposed manual evaluation we analyzed the performance of current learning algorithms, showing that it is mediocre and that ignoring the context under which a rule should be applied significantly hurts performance.

We next introduced a novel entailment rule representation based on morphosyntactic canonical forms of lexical syntactic templates. Using this representation, the coverage of rule application is improved, by applying the proposed morpho-syntactic matching methodology. Furthermore, this representation also improves the coverage of rule learning, as more statistics can be gathered for each learned rule. In addition, the focus of rule learning under this representation is changed from learning and evaluating morpho-syntactic variations, such as passive forms, to learning variations that involve lexical changes.

Our next step was to improve current rule learning approaches. We first focused on unsupervised learning from general corpora. We discussed the limitations of symmetric distributional similarity measures for learning entailment rules, and showed that directional measures are more adequate for the task. Furthermore, we exposed the limitations of the common DIRT-style representation of predicative binary templates as dependency paths. Instead, we proposed to learn rules between unary templates as a better representation. We showed that the representation of unary templates as unary dependency paths correctly describes predicates that cannot be correctly captured by binary paths. Our results show that unary rules outperform binary rules and that directional measures outperform symmetric measures for the entailment rule learning task.

Continuing our effort to generate an accurate rule-base, we next changed our focus from collections of texts to more informative resources. We took the WordNet lexicon as a baseline for manually-created entailment relation information. Yet, many of WordNet’s entailment relations between predicates cannot be utilized to
a full extent since they also require changes in the syntactic positions of predicate arguments, information that WordNet lacks. We introduced a novel framework for augmenting WordNet with argument mapping, termed Argument-mapped WordNet (AmWN), and described a concrete unsupervised implementation of this framework. Additionally, we presented general methods for WordNet rule-filtering, which improve WordNet’s performance. Our results show that our full-blown implementation of AmWN substantially improves WordNet-based inference both in terms of coverage and accuracy.

Finally, we addressed the critical need for taking context into consideration when applying entailment rules by proposing a novel framework, termed Contextual Preferences, which generalizes and extends prior work. We described a concrete unsupervised implementation of the framework, which is based on surface text representation instead of explicit sense tagging. Our results show a substantial improvement in rule application quality over earlier state-of-the-art approaches.

To conclude, this thesis took our understanding of predicative entailment rules a step forward with respect to the research topics that should be investigated in this field as well as to their state-of-the-art approaches. Overall, our proposed algorithms and frameworks provide a more accurate description of the space of predicative entailment rules. We utilized different resources from which unsupervised learning of entailment rules can result in large-scale acquisition of entailment rules, and by improving the limitations of current approaches, in aspects such as entailment rule representation, directional approaches for learning (instead of symmetric “paraphrase learning” approaches) and improved context sensitive rule application, we showed that unsupervised approaches can generate large-scale rule-bases that are accurate enough to be useful in real applications.

The research in this thesis introduced novel frameworks and approaches for entailment rule learning and application whose utilization was demonstrated by concrete
implemented methods. Still, these specific methods are only starting points. Many paths of research, as well as interesting questions, remain open. Further development of better context-sensitive models under the Contextual Preferences framework on the one hand and novel directional measures for rule learning on the other hand are just some of these possible directions. In another direction, the AmWN framework provides a “holistic” approach for combining different entailment rule knowledge-bases into a single global resource. Thus, it would be interesting to combine lexical knowledge that is missing from WordNet all together into the AmWN graph, such as from the unary-DIRT rule-base, or rules learned from additional resources such as comparable corpora. Finally, current unsupervised approaches focus only on positive statistics to affirm the correctness of a rule. It would be interesting to complement this approach by finding types of negative statistics that weaken or contradict the correctness of a rule, combining the two types of statistics for more accurate rule learning.

As a final remark, all our acquisition methods and template matching procedures are parser dependant. Consequently, our current implementations for our proposed algorithms and frameworks and our learned rule-bases may be utilized only within applications that use Minipar as their syntactic parser. As parsing technology keeps evolving, the dependency on specific parser implementations is rather limiting in terms of sharing resources and implementations. It thus seems worthwhile, both to the parsing community and to its users, to develop a standard platform under which parser implementations could be more easily replaced.
Bibliography


Clark, Peter and Phil Harrison. 2008. Recognizing textual entailment with logical inference. In *Proceedings of Text Analysis Conference*, Gaithersburg, Maryland USA.


Bibliography


Palmer, Martha, Hoa Trang Dang, and Christiane Fellbaum. 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. Natural Language Engineering, 13(2):137–163.


Pennacchiotti, Marco, Roberto Basili, Diego De Cao, and Paolo Marocco. 2007. Learning selectional preferences for entailment or paraphrasing rules. In Proceedings of RANLP.


Appendix A

AmWN Compact Representation

This appendix describes a compact representation of the AmWN graph presented in Section 7.2.

Node Grouping

In our framework, formally each node in the AmWN graph consists of a single term together with a specific synset-id a specific subcategorization frame of that term. However, to save both graph space and running time, we collapse some nodes into one aggregate node. To that extent, we follow WordNet, were all members of a synset are described in one synset node.

Unlike WordNet, not all synset members are aggregated into a single node in AmWN, only those members whose nodes share the same frame. For example, sometimes not all frame information in Nomlex is shared between all nominal members of a synset. We group together synset members sharing the same frame into one aggregate synset-frame-node and attach all incoming and outgoing edges of all original nodes to the new node. When a synset-frame-node is reached while traversing the graph for rule generation, all possible rules formed by any of the node members are
generated. In addition, when testing for valid paths during rule filtering, all possible
paths containing different members of synset-frame-nodes are tested for validity.

**Edge Grouping**

Similar to node grouping, there are also edges that could be grouped. whenever a
set of edges map from a list of possible realizations of a specific functional role of
one node to another list of realizations of a functional role of another node, they are
aggregated to a single edge where instead of a single argument position at each side
of the edge, a list of a possible argument positions are described. Whenever such edge
is traversed, all the templates that correspond to the change in argument position
to one of the positions on the list are created. As in node grouping, these different
argument position variations are also taken into consideration during path validation
for rule filtering.