Large-Scale Paraphrasing for Text-to-Text Generation

by

Juri Ganitkevitch

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

February, 2018

Abstract

We present our work on the extraction and estimation of syntactic paraphrases using commodity text data and automated linguistic annotation. Our initial approach leverages bilingual parallel data and builds on extraction techniques for synchronous context-free grammars frequently used in statistical machine translation. We extend our estimation methods to include contextual similarity metrics drawn from vast amounts of plain English text. We evaluate the quality of our paraphrases by applying a generalizable adaptation scheme that tunes our paraphraser to arbitrary text-to-text generation tasks. Our system produces results comparable with contemporary systems with only little data and work needed. We further discuss the scaling of our extraction method to large data sizes, and the building of the paraphrase database PPDB, a large-scale collection of paraphrases in 23 languages.

Primary Reader: Chris Callison-Burch

Secondary Reader: Benjamin Van-Durme

Acknowledgments

First of all, I would like to thank my advisor Chris Callison-Burch and my co-advisor Benjamin Van Durme for their long years of patience, collaboration, and help. To have worked with them was a great privilege, and one of the most fortunate things to happen in my life. Thank you for taking a chance on me, and for doing so much not just to help me grow as a researcher, but also to make a community.

Relatedly, I am deeply grateful to have been part of a research lab where diversity was an actively pursued, visibly important matter. It taught me so much, and I hope to bring this spirit with me as I go on. I remain forever impressed with so many of the people I worked alongside of, and I feel honored to have shared in their time.

I also want to thank my thesis committee member Philipp Koehn for his thoughtful comments and advice on this thesis, as well as for his excellent qualities as a fellow German in times of World Cups.

Through the years that *eventually* gave rise to this document, I could not have asked for a better group of peers, and friends than I found here. Omar, Alex, Jonny, Courtney, Katie, Ann(i), and Dirk. Thank you for making a dispersed research

community feel a little bit like a home. Thank you also to the amazing Desirée and Ruth, for the spark and the warmth with which they infused life at CLSP.

I'll always be grateful to my brother, Vladimir, for his unwavering friendship, keeping up our over two-decade-old asinine in-jokes, and the unfathomable strength he showed when I wasn't there to. I am always indebted to my mother Irina for her love, support, and incessant nagging for me to graduate.

Finally, I want to thank Ann, my partner and person. You make me happier than I can express. Let's go to there.

Dedication

To my father.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

The focus of this thesis is the extraction and application of paraphrases at scale. Paraphrases are pairs of natural language expressions that we consider to be identical or very similar in meaning. A collection of paraphrases thus offers a way to capture the different ways in which a particular concept may be expressed in natural language. For practitioners of natural language processing (NLP), this makes paraphrases a valuable resource: if we know more ways in which a concept can be stated in natural language, our NLP system will have better coverage, we will need less data to achieve comparable quality, and the machine-learned models driving our system's decision-making will be able to generalize better. The use of paraphrases in NLP can offer a straightforward avenue to abstract an NLP system's functionality from the surface

form of a natural language statement to the concepts expressed within.

Prior to the work presented in this thesis, state of the art paraphrase extraction methods typically forced the unwieldy choice between quality and volume. More data-driven paraphrase extraction methods could extract high volumes of paraphrases, but be limited to shallow and noise-prone formats lacking linguistic annotation. More linguistically driven approaches were able to produce richly annotated paraphrases, but in severely limited amounts, bounding the paraphrases' usefulness. We present a more detailed overview in Chapter 2.

In this work, we begin by investigating whether linguistically rich paraphrases can be extracted from commodity data sources like bilingual parallel data. The bilingual parallel corpora we use are sentence-aligned data sets in two different languages where the aligned sentences are translations of each other. Large amounts of human-translated text have been produced over time, owing to a social need to transfer and make available information across the many natural languages spoken around the world. This data has been compiled and made available by the academic and industrial efforts driving research in statistical machine translation (SMT). A variety of sources are leveraged to produce sentence-aligned bitext data, ranging from parliamentary proceedings and other government documents produced by multilingual bodies of government such as the European Union or the Canadian government. Bitext data has also been extracted from other domains where translation work is commonplace, like books, newswire, movie subtitles, and technical manuals. Even

larger volumes of bilingual parallel data have been obtained through statistical methods by crawling the Web for potential translations and probabilistically aligning them. After over 20 years of research into statistical machine translation, especially English-to-non-English bitexts are easily and plentifully available.

An equally important prerequisite to our work is the availability of good-quality automatic linguistic annotation. Over two decades of work in statistical parsing and tagging, building on the release of the Penn Treebank corpus have yielded publicly available, efficient, and high-precision toolkits to automatically annotate English text with linguistic structure. We can use these tools to cheaply and quickly produce rich annotations on large volumes of text – leveraging the sheer amount of data to weed out potential outlier mistakes made by the parser.

This thesis investigates whether we can obtain rich, linguistically annotated paraphrases at scale by combining automated linguistic annotation with large amounts of English-sided bitext by pivoting through the non-English language. We use pivoting and probability-estimation approaches that adapt methods used in syntactically informed statistical machine translation to extract probabilistic synchronous context-free grammars (PSCFG) of paraphrases (Chapter 3). We conduct a qualitative analysis of the obtained paraphrases, showing that the method is indeed capable of capturing well-known syntactic rewrites like the possessive rule, the dative shift, or reduced relative clause rewrites, as shown in Table 1.1.

To provide a more empirical and quantitative evaluation of the syntactic para-

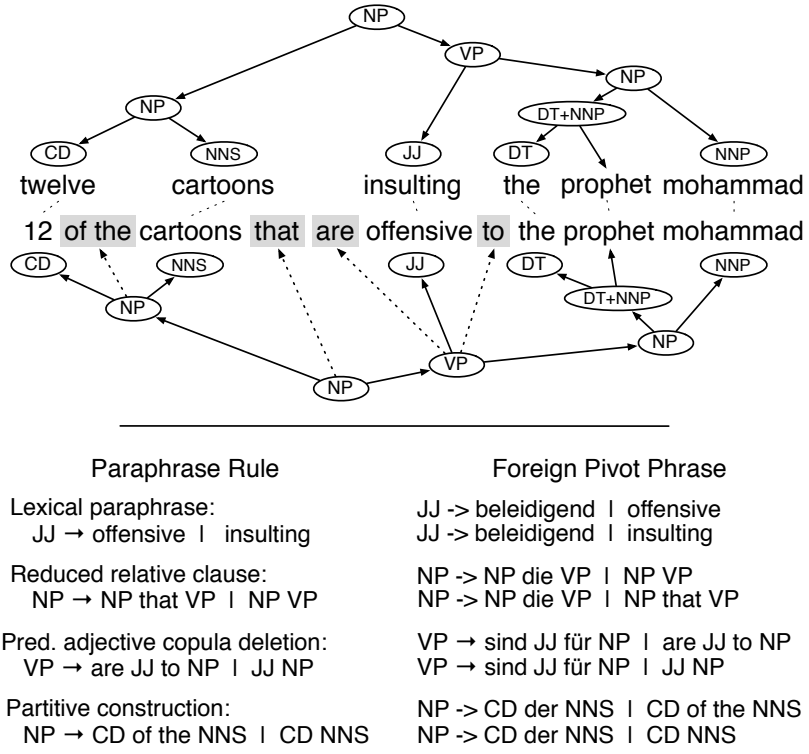

Figure 1.1: An example of a synchronous paraphrastic derivation using syntactically labeled grammars. A few of the rules applied in the parse are show in the left column, with the German-English pivot phrases that gave rise to them on the right.

Possessive rule	$NP \rightarrow$	the NN of the NNP	|	the NNP's NN
	$NP \rightarrow$	the NNS_1 made by NNS_2	|	the NNS_2's NNS_1
Dative shift	$VP \rightarrow$	give NN to NP	|	give NP the NN
	$VP \rightarrow$	provide NP_1 to NP_2	|	give NP_2 NP_1
Adv./adj. phrase move	$S/VP \rightarrow$	$ADVP$ they VBP	|	they VPB $ADVP$
	$S \rightarrow$	it is $ADJP$ VP	|	VP is $ADJP$
Verb particle shift	$VP \rightarrow$	VB NP up	|	VB up NP
Reduced relative clause	$SBAR/S \rightarrow$	although PRP VBP that	|	although PRP VBP
	$ADJP \rightarrow$	very JJ that S	|	JJ S
Partitive constructions	$NP \rightarrow$	CD of the NN	|	CD NN
	$NP \rightarrow$	all $DT\backslash NP$	|	all of the $DT\backslash NP$
Topicalization	$S \rightarrow$	NP, VP .	|	VP, NP .
Passivization	$SBAR \rightarrow$	that NP had VBN	|	which was VBN by NP
Light verbs	$VP \rightarrow$	take action $ADVP$	|	to act $ADVP$
	$VP \rightarrow$	TO take a decision PP	|	TO decide PP

Table 1.1: A selection of meaning-preserving transformations and hand-picked examples of syntactic paraphrases that our system extracts capturing these.

phrases our approach produces, we construct a flexible text-to-text generation framework. Our method casts paraphrasing as an English-to-English translation problem, allowing us to apply syntactic SMT machinery. Figure 1.1 shows an example derivation, the paraphrase rules applied, and examples of German pivot expressions that gave rise to the rules. Our approach uses the paraphrase grammar and a lightweight adaptation scheme to reduce a given text rewriting task to a targeted paraphrase problem.

On the example of text shortening, or compression, we show that a rapidly adapted paraphrase grammar, applied through a readily available SMT decoder, can indeed produce text-to-text generation performance comparable to specialized NLP compres-

Source	he also expected that he would have a role in the future at the level of the islamic movement across the palestinian territories , even if he was not lucky enough to win in the elections .
Reference	he expects to have a future role in the islamic movement in the palestinian territories if he is not successful in the elections .
Paraphrases	he also expected that he would have a role in the future of the islamic movement in the palestinian territories , although he was not lucky enough to win elections .
ILP	he also expected that he would have a role at the level of the islamic movement , even if he was not lucky enough to win in the elections .
Source	in this war which has carried on for the last 12 days , around 700 palestinians , which include a large number of women and children , have died .
Reference	about 700 palestinians , mostly women and children , have been killed in the israeli offensive over the last 12 days .
Paraphrases	in this war has done for the last 12 days , around 700 palestinians , including women and children , died .
ILP	in this war which has carried for the days palestinians , which include a number of women and children died .
Source	hala speaks arabic most of the time with her son , taking into consideration that he can speak english with others .
Reference	hala speaks to her son mostly in arabic , as he can speak english to others .
Paraphrases	hala speaks arabic most of the time with her son , considering that he can speak english with others .
ILP	hala speaks arabic most of the time , taking into consideration that he can speak english with others .

Table 1.2: Examples of text compressions produced by our paraphrase-based text-to-text approach (Paraphrases), a specialized integer linear programming solver-based compression system (ILP), along with the input sentence and a human-generated paraphrase.

sion systems (Chapter 4). In Table 1.2, we show a sample of compressions generated by our adapted text-to-text system.

Another type of easily available, high-volume corpora is plain, monolingual English text. We conduct a series of experiments to investigate the usefulness of signal drawn from this massive data source for our paraphrase extraction work. Prior research has largely relied on distributional signals to extract paraphrases from plain text data. We opt for a more high-precision approach, and choose to augment our extraction framework by integrating monolingual text as an additional source of signal to assess the quality of a paraphrase pair.

We compare simpler, word-count based methods of context representation and context representations that make use of rich, automatically generated syntactic annotations over text. Our work shows that overall distributional signal derived from monolingual text data improves paraphrase quality. It also demonstrates that automatically annotating large volumes of text data yields improvements in paraphrase quality that exceed those derived from purely word count based approaches (Chapter 5).

The core thesis of this work is that high-quality paraphrases can be derived from commodity data and automated annotations, and, when made readily available and efficiently usable, can drive progress in NLP research. The main contribution of the research we present in this thesis, is therefore condensed in a large-scale paraphrase resource we extracted, estimated, and made publicly available. This resource, the

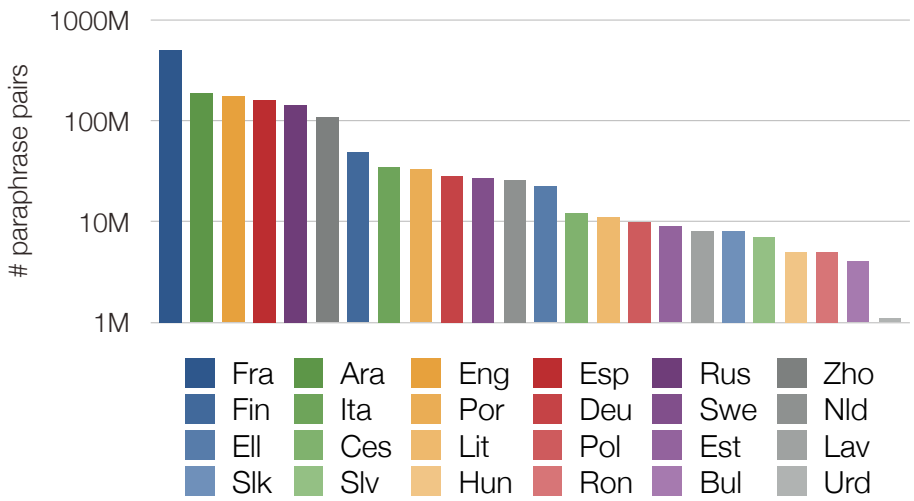

Figure 1.2: A visualization of the resource size for the paraphrase database per language, measured in millions of paraphrase pairs. Languages are in order of resource size.

ParaPhrase DataBase (PPDB), to date remains the largest published paraphrase collection.

The composite parallel corpus PPDB is extracted from has more than 106 million sentence pairs, over 2 billion English words, and spans 23 pivot languages. To scale our extraction machinery to this size of data, and to make the resulting paraphrase collection more widely usable, we explore compact representations for grammar extraction from bitexts, and devise a highly compact representation for probabilistic synchronous CFGs. The resulting system reduces time and space requirements for grammar extraction by a factor of 4, effectively eliminates the loading overhead for large translation and parapharse grammars, and reduces their memory footprint by

over 90%.

Additionally, reversing the pivoting approach developed for English, we made PPDB available in a variety of other languages, in many cases marking the first release of a paraphrase corpus for that language (Chapter 6). Figure 1.2 illustrates the size of the published paraphrase collections per language.

1.2 Related publications

Ganitkevitch et al. [2011] describes the pivot-based extraction of syntactic paraphrases from bilingual parallel corpora, and their application to text-to-text generation.

In Ganitkevitch et al. [2012b] we describe and evaluate the integration of monolingual distributional signal into paraphrase extraction.

Ganitkevitch et al. [2013] and Ganitkevitch and Callison-Burch [2014] mark, respectively, the releases of the Paraphrase Database in English and in its multi-lingual iteration. Ganitkevitch et al. [2012a] reports on the efficiency improvements that enabled the extraction of PPDB.

1.3 Disclaimer

The entirety of the research discussed in this thesis was conducted and published before the use of deep neural networks and word embeddings-based methods became

state-of-the-art in NLP. In fact, the chief contribution of this thesis, the Paraphrase Database, remains widely used in deep learning-based NLP work, frequently cited as a useful resource in improving the quality of word embeddings.

Chapter 2

Background and Related Work

In this thesis, we adapt methods from statistical machine translation towards paraphrase extraction and paraphrase-based text rewriting. Specifically, we adapt techniques from syntactically informed, parsing-based statistical machine translation to extract syntactic paraphrases, rank them, and perform targeted text-to-text generation.

This chapter presents an overview of prior work in data-driven paraphrase extraction, and applications in text rewriting, as well as the statistical machine translation methods we appropriate and adapt for our work. Section 2.1 addresses the nature and utility of paraphrases, followed by an overview of published efforts on paraphrase extraction. In Section 2.2 we give an overview of methods and formalisms used in statistical machine translation.

2.1 Paraphrase Extraction

2.1.1 Paraphrases, How Do They Work?

The object of the majority of the efforts presented in this thesis, paraphrases, are notoriously hard to formally define. Throughout this thesis and associated publications, we use a fairly informal notion of what a paraphrase is. Our work is concerned with what Bhagat and Hovy [2013] call a quasi-paraphrase: loose text rewrites that preserve enough meaning to be still be useful for practical applications. In this we follow a paraphrase definition given in Barzilay [2003]: paraphrases are language units that express approximate conceptual equivalence, and can be substituted for one another in many contexts.

For instance we consider *soldiers* and *combatants* perfectly valid, high-quality paraphrases, even though the terms clearly differ in their legal definitions, more fine-grained implications, and context-dependent associations. To us, the overlapping conceptual notion of "persons participating in an armed conflict" is the salient point, along with the fact that *there exist* a multitude of contexts in which substituting one for the other is largely meaning-preserving.

In this, our work is a precursor to the more in-depth classification efforts of Pavlick et al. [2015a], who presented a method for classifying quasi-paraphrases into finer-grained relations based on the Natural Logic paradigm MacCartney [2009], like hypernyms, hyponyms, and proper equivalence. Pavlick et al. [2015b] applied this improved

categorization of paraphrases at scale, dramatically improving the utility of automatically extracted paraphrase corpora for tasks like recognition of textual entailment, where the precise relationship underlying a quasi-paraphrase matters.

For the remainder of this thesis, we take the term "paraphrases" to mean quasi-paraphrases, as stated above.

2.1.2 Classes and Applications of Paraphrases

Automatically generating and detecting paraphrases is a crucial aspect of many NLP tasks. In multi-document summarization, paraphrase detection is used to collapse redundancies [Barzilay et al., 1999, Barzilay, 2003]. Paraphrase generation can be used for query expansion in information retrieval and question answering systems [McKeown, 1979, Anick and Tipirneni, 1999, Ravichandran and Hovy, 2002, Riezler et al., 2007]. Paraphrases allow for more flexible matching of system output against human references for tasks like machine translation and automatic summarization [Zhou et al., 2006, Kauchak and Barzilay, 2006, Madnani et al., 2007, Snover et al., 2010]. Paraphrases are used to generate additional reference translations for statistical machine translation [Madnani et al., 2007] and for more flexible matching when evaluating the output of machine translation systems or automatically generated summaries [Zhou et al., 2006, Kauchak and Barzilay, 2006, Owczarzak et al., 2006].

In this thesis, we will use the following terminology for different types of para-

phrases: *Lexical paraphrases* denote pairs of synonyms, i.e. words with the same meaning:

ratification | approval

Lexical paraphrases are comparatively straightforward to extract and easy to apply. The single-word format however limits them in both contextualization and expressive power. One way to amend this is to move to *phrasal paraphrases*. These extend the notion of synonymity to contiguous multi-word phrases like:

the proposal's ratification | the approval of the motion.

Paraphrastic multiword units offer considerably more expressive power, and can capture complex rewrites like the one above. However semantic similarity is captured purely via memorization; phrasal paraphrases do not generalize well.

Paraphrase patterns that add nonterminal slots into the expression are a step towards greater generalization capabilities:

X_1's ratification | the approval of X_1.

However, since the nonterminal slots in paraphrase patterns are unconstrained they can vastly overgeneralize. For example, the above pattern could be correctly applied to paraphrase the sentence

The press anxiously awaited the proposal's ratification.

into (X_1 matching "the proposal")

The press anxiously awaited the approval of the proposal.

But the X_1 could also be mis-applied to match "anxiously awaited the proposal," leading to the ungrammatical

The press the approval of anxiously awaited the proposal.

This makes it difficult to meaningfully use paraphrase patterns for high-level paraphrastic transforms like the possessive rule. E.g. the pattern

$$X_1\text{'s } X_2 \mid X_2 \text{ of } X_1$$

overgeneralizes too much to be useful.

To limit the paraphrases' generalizations to useful cases, we can add syntactic constraints to the patterns' nonterminals. The resulting *syntactic paraphrases* contain *slots* that are annotated with syntactic constraints:

$$NP_1\text{'s } NP_2 \mid NP_2 \text{ of } NP_1$$

It is evident that syntactic paraphrases have a much higher potential for well-formed generalization, and for capturing interesting paraphrastic transformations.

2.1.3 Sources for Data-Driven Paraphrase Extraction

A variety of different types of corpora and semantic equivalence cues have been used to automatically induce paraphrase collections for English. Madnani and Dorr

[2010] survey a variety of data-driven paraphrasing techniques, categorizing them based on the type of data that they use. These include large monolingual texts [Lin and Pantel, 2001, Szpektor et al., 2004, Bhagat and Ravichandran, 2008], comparable corpora [Barzilay and Lee, 2003, Dolan et al., 2004], monolingual parallel corpora [Barzilay and McKeown, 2001, Pang et al., 2003], and bilingual parallel corpora [Bannard and Callison-Burch, 2005, Madnani et al., 2007, Zhao et al., 2008a]. We focus on the latter type of data.

Perhaps the most natural type of corpus for paraphrase extraction is a monolingual parallel text, which allows sentential paraphrases to be extracted since the sentence pairs in such corpora are perfect paraphrases of each other [Barzilay and McKeown, 2001, Pang et al., 2003]. While rich syntactic paraphrases have been learned from monolingual parallel corpora, they suffer from very limited data availability and thus have poor coverage. Dolan et al. [2004] work around this issue by extracting parallel sentences from the vast amount of freely available comparable English text and apply machine translation techniques to create a paraphrasing system [Quirk et al., 2004]. However, the word-based translation model and monotone decoder they use results in a substantial amount of identity paraphrases or single-word substitutions.

Other methods obtain paraphrases from raw monolingual text by relying on distributional similarity [Lin and Pantel, 2001, Bhagat and Ravichandran, 2008]. While vast amounts of data are readily available for these approaches, the distributional similarity signal they use is noisier than the sentence-level correspondency in parallel

corpora and additionally suffers from problems such as mistaking cousin expressions or antonyms (such as $\{boy, girl\}$ or $\{rise, fall\}$) for paraphrases.

Abundantly available bilingual parallel corpora have been shown to address both these issues, obtaining paraphrases via a pivoting step over foreign language phrases [Bannard and Callison-Burch, 2005]. The coverage of paraphrase lexica extracted from bitexts has been shown to outperform that obtained from other sources [Zhao et al., 2008b]. While there have been efforts pursuing the extraction of more powerful paraphrases [Madnani et al., 2007, Callison-Burch, 2008, Cohn and Lapata, 2008, Zhao et al., 2008a], it is not yet clear to what extent meaningful syntactic paraphrases can be induced from bitexts.

In their paraphrase extraction work using bilingual parallel corpora Bannard and Callison-Burch [2005] used techniques from *phrase-based* statistical machine translation [Koehn et al., 2003]. After extracting a bilingual phrase table, English paraphrases are obtained by pivoting through foreign language phrases. The phrase table contains phrase pairs (e, f) (where the e and f stand for English and foreign phrases, respectively) as well as bi-directional translation probabilities $p(e|f)$ and $p(f|e)$.

Since many paraphrases can be extracted for a phrase, Bannard and Callison-Burch [2005] rank them using a paraphrase probability defined in terms of the trans-

lation model probabilities $p(f|e)$ and $p(e|f)$:

$$p(e_2|e_1) = \sum_f p(e_2, f|e_1) \quad (2.1)$$

$$= \sum_f p(e_2|f, e_1)p(f|e_1) \quad (2.2)$$

$$\approx \sum_f p(e_2|f)p(f|e_1). \quad (2.3)$$

Several subsequent efforts extended the bilingual pivoting technique, many of which introduced elements of more contemporary *syntax-based* approaches to statistical machine translation. Madnani et al. [2007] extended the technique to *hierarchical* phrase-based machine translation [Chiang, 2005], which is formally a synchronous context-free grammar (SCFG) and thus can be thought of as a *paraphrase grammar*. The paraphrase grammar can paraphrase (or "decode") input sentences using an SCFG decoder, like the Hiero, Joshua or cdec MT systems [Chiang, 2007, Li et al., 2009, Dyer et al., 2010]. Like Hiero, Madnani's model uses just one nonterminal X instead of linguistic nonterminals.

Additional efforts incorporated linguistic syntax. Callison-Burch [2008] introduced syntactic constraints by labeling all phrases and paraphrases (even non-constituent phrases) with CCG-inspired slash categories [Steedman and Baldridge, 2011], an approach similar to Zollmann and Venugopal [2006]'s syntax-augmented machine translation (SAMT). Callison-Burch did not formally define a synchronous grammar, nor discuss decoding, since his presentation did not include hierarchical rules. Cohn and Lapata [2008] used the GHKM extraction method [Galley et al., 2004], which is lim-

ited to constituent phrases and thus produces a reasonably small set of syntactic rules. Zhao et al. [2008a] added slots to bilingually extracted paraphrase patterns that were labeled with part-of-speech tags, but not larger syntactic constituents.

More recent work leveraged temporal and structural signals available in online newswire text and on Twitter[1] to extract sentential and predicate-level paraphrases from quasi-parallel text. Zhang and Weld [2013] use a set of temporal heuristics over entities appearing in news articles to model whether relations observed between such entities may be paraphrases of one another. They later extend their method to replace entity candidates generated by an off-the-shelf relation extraction system with an unsupervised approach, showing further improvement in both precision and recall [Zhang et al., 2015].

Xu et al. [2014] present a supervised approach, extracting sentential paraphrases from news headlines on Twitter using a latent-variable model that leverages signals derived from the *trending topics* clustering and pivoting over shared entity mentions.

2.1.3.1 Paraphrases and Word Sense Ambiguity

In this work our focus is on extending the paraphrase extraction methods of Bannard and Callison-Burch [2005]. In their work, and in ours, paraphrase extraction from bilingual corpora relies on the assumption that two English expressions e and e' that share a common high-likelihood translation f are likely paraphrases. It is worth

[1] `twitter.com`

noting that while we are trying to relate two different *types* e and e' by drawing on their occurrences and translations over a dataset, we are glossing over a sense ambiguity problem: for an ambiguous expression e different occurrences of e may in fact have different meanings (e.g. *river bank* and *savings bank*).

Diab [2003] addresses the task of word sense disambiguation (WSD) in a multilingual framework, proposing a projection method to compute estimates for word sense in foreign languages that takes into account the different known senses of a word's English translations.

Addressing word sense ambiguity explicitly as part of a paraphrase extraction approach is an interesting research problem. In this work, we take no special precautions to avoid cross-contamination from different word senses being lumped together. Nonetheless, some extensions we present to prior paraphrase extraction work make the resulting paraphrases more robust against WSD problems: in Chapter 3 we introduce the requirement that an expression's syntactic label has to match that of its paraphrases, avoiding ambiguity that can be resolved with part-of-speech labeling that takes context into account (e.g. *lead paint* and *she lead a team*). Chapter 5 adds a distributional similarity signal that can further help detect sense mismatches.

2.1.4 Paraphrasing and Statistical Machine Translation

Before the shift to statistical natural language processing, paraphrasing was often treated as syntactic transformations or by parsing and then generating from a semantic representation [McKeown, 1979, Muraki, 1982, Meteer and Shaked, 1988, Shemtov, 1996, Yamamoto, 2002]. Indeed, some work generated paraphrases using (non-probabilistic) synchronous grammars[Shieber and Schabes, 1990, Dras, 1997, 1999, Kozlowski et al., 2003].

After the rise of statistical machine translation, many machine translation techniques were repurposed for paraphrasing. These include sentence alignment [Gale and Church, 1993, Barzilay and Lee, 2003], word alignment and noisy channel decoding [Brown et al., 1990, Quirk et al., 2004], phrase-based models [Koehn et al., 2003, Bannard and Callison-Burch, 2005], hierarchical phrase-based models [Chiang, 2005, Madnani et al., 2007], log-linear models, minimum error rate training [Och, 2003, Madnani et al., 2007, Zhao et al., 2008b], and syntax-based machine translation [Wu, 1997, Yamada and Knight, 2001, Melamed, 2004, Quirk et al., 2005].

Relying on small data sets of semantically equivalent translations, Pang et al. [2003] created finite state automata by syntax-aligning parallel sentences, enabling the generation of additional reference translations.

Both Barzilay and McKeown [2001] and Ibrahim et al. [2003] sentence-align exist-

ing noisy parallel monolingual corpora such as translations of the same novels. While Ibrahim et al. [2003] employ a set of heuristics that rely on anchor words identified by textual identity or matching liguistic features such as gender, number or semantic class, Barzilay and McKeown [2001] use a co-training approach that leverages context similarity to identify viable paraphrases.

Semantic parallelism is well-established as a stong basis for the extraction of correspondencies such as paraphrases. However, there are notable efforts that choose to forgo it in favor of clustering approaches based on distributional characteristics. The well-known DIRT method by Lin and Pantel [2001] fully relies on distributional similarity features to extract inference rules. Patterns extracted from paths in dependency graphs are clustered based on the similarity of the observed contents of their slots.

Similarly, Bhagat and Ravichandran [2008] argue that vast amounts of text can be leveraged to make up for the relative weakness of distributional features compared to parallelism. They also forgo complex annotations such as syntactic or dependency parses, relying only on part-of-speech tags to inform their approach. In their work, relations are learned by finding pattern clusters initially seeded by already known patterns. However, this method is not capable of producing syntactic paraphrases.

2.2 Statistical Machine Translation

Work in statistical machine translation (SMT) relies on *bilingual parallel corpora* (or *bitexts*) to learn models for translating between natural languages. Bilingual parallel corpora are collections that pair sentences in one language with their translations in another language.

In statistical machine translation systems, bitexts are used to learn *word alignment* models that compute the probability of individual words in a sentence pair being translations. Based on a word alignment model trained from the bitext, a single best alignment is then computed for each sentence pair. Subsequently, larger translation units, like pairs of contiguous multi-word *phrases* or *context-free grammar rules* that include nonterminal gaps, can be extracted from the sentence pair. Translation probabilities for these elements are estimated by aggregating counts over the entire bitext. The resulting translation model can then be used to translate previously unseen source language sentences by simultaneously finding the best way to partition the input sentence into known units, and picking the best translation for each of these units. A *language model* in the target language is used to ensure the fluency of the output produced.

In our work, we adapt much of this machinery to extract syntactically informed paraphrases from bilingual data. In the following, we give a brief overview of the major components of an SMT system that our work builds on.

2.2.1 Word-Based Machine Translation and Alignments

Brown et al. [1990] first cast translation as a probabilistic process. Given a French sentence $\mathbf{f}$, we can find the best English translation $\hat{\mathbf{e}}$ by choosing the most likely $\mathbf{e}$:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) \tag{2.4}$$

$$= \arg\max_{\mathbf{e}} \frac{p(\mathbf{e})p(\mathbf{f}|\mathbf{e})}{p(\mathbf{f})} \tag{2.5}$$

$$= \arg\max_{\mathbf{e}} p(\mathbf{e})p(\mathbf{f}|\mathbf{e}). \tag{2.6}$$

This formulation decomposes into two sub-models, the *language model* $p(\mathbf{e})$ and the *translation model* $p(\mathbf{f}|\mathbf{e})$. The language model quantifies the fluency of $\mathbf{e}$ (it aims to answer the question "how likely is this sentence to appear in well-formed English?"), while the translation model quantifies how close in meaning $\mathbf{e}$ is to the original French input $\mathbf{f}$. We describe each model in the following.

2.2.1.1 Language Model

The language model, borrowing from speech recognition, was an n-gram model. Under an n-gram model, the probability of a sentence was the probability of each

successive word, given the n previous words.

$$
\begin{aligned}
p(\mathbf{e}) &= p(e_1^I) && (2.7)\\
&= p(e_1) \cdot p(e_2|e_1) \cdot p(e_3|e_2, e_1) \cdot p(e_4|e_3 e_2 e_1) \cdots && (2.8)\\
&= \prod_{i=1}^{I} p(e_i|e_{i-1} \ldots e_{i-n}). && (2.9)
\end{aligned}
$$

The language model parameters, i.e. the individual n-gram probabilities are typically estimated using occurrence counts over a large monolingual corpus. A variety of techniques have been developed to smooth the language model probabilities to ensure better performance on unseen data and account for the incompleteness of the training corpus.

While linguistically crude, n-gram models have proven highly successful and are still in use in a wide variety of natural language processing tasks, including machine translation and paraphrasing.

2.2.1.2 Word-Based Translation Model

Owing to dearth of data and the computational limitations of the time, the translation model Brown et al. [1990] introduced is similarly simple in its approach. The translation probability is defined by way of word-to-word translation probabilities, and word alignments between the two sentences:

$$p(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a}} p(\mathbf{f}, \mathbf{a}|\mathbf{e}) \qquad (2.10)$$

Brown et al. [1993] define a series of increasingly complex word-based translation and alignment models, referred to as the IBM models. The simplest of these, Model 1, is based purely on co-occurrence statistics over the bitext. The subsequent models introduce additional parameters to account for translation effects like differences in word order (modeled by the *distortion* probability $d(p_i|i, l, m)$), words that do not have direct counterparts in the other language (modeled by a *spurious word* probability parameter), and single words in one language translating as multiple words in the other (captured by a *fertility* probability parameter).

Since the bitext does not contain given alignments, these parameters cannot be estimated directly. Instead, Brown et al. [1993] treat them as hidden variables, and learn the model parameters using the expectation maximization (EM) algorithm [Dempster et al., 1977]. Especially for complex models, the EM algorithm is liable to converge to sub-optimal local minima. To ameliorate this, Brown et al. [1993] attempt initializing the model parameters to reasonable values. Specifically, each model in the IBM model family is seeded with the parameters learned from training its simpler precursor.

While state-of-art translation modeling in machine translation has long moved past word-based translation, word alignments and the IBM models still are a key component in learning the more complex and linguistically informed models.

2.2.2 SCFGs in Translation

The model we use in our paraphrasing approach is a syntactically informed *synchronous context-free grammar* (SCFG). The SCFG formalism [Aho and Ullman, 1972] was repopularized for statistical machine translation by Chiang [2005]. Formally, a *probabilistic* SCFG $\mathcal{G}$ is defined by specifying

$$\mathcal{G} = \langle \mathcal{N}, \mathcal{T}_S, \mathcal{T}_T, \mathcal{R}, S \rangle,$$

where $\mathcal{N}$ is a set of nonterminal symbols, $\mathcal{T}_S$ and $\mathcal{T}_T$ are the source and target language vocabularies, $\mathcal{R}$ is a set of rules and $S \in \mathcal{N}$ is the root symbol. The rules in $\mathcal{R}$ take the form:

$$C \rightarrow \langle \gamma, \alpha, \sim, w \rangle,$$

where the rule's left-hand side $C \in \mathcal{N}$ is a nonterminal, $\gamma \in (\mathcal{N} \cup \mathcal{T}_S)^*$ and $\alpha \in (\mathcal{N} \cup \mathcal{T}_T)^*$ are strings of terminal and nonterminal symbols with an equal number of nonterminals $c_{NT}(\gamma) = c_{NT}(\alpha)$ and

$$\sim: \{1 \ldots c_{NT}(\gamma)\} \rightarrow \{1 \ldots c_{NT}(\alpha)\}$$

constitutes a one-to-one correspondency function between the nonterminals in γ and α. A non-negative weight $w \geq 0$ is assigned to each rule, reflecting the likelihood of the rule.

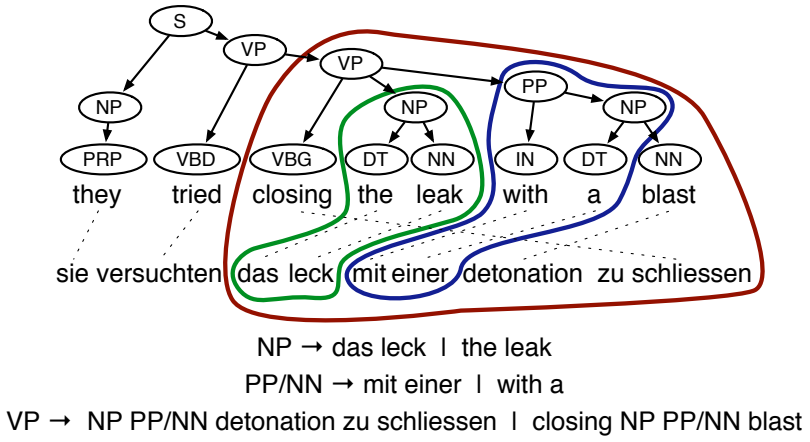

Figure 2.1: Synchronous grammar rules for translation are extracted from sentence pairs in a bitext which have been automatically parsed and word-aligned. Extraction methods vary on whether they extract only minimal rules for phrases dominated by nodes in the parse tree, or more complex rules that include non-constituent phrases.

2.2.3 Rule Extraction

Phrase-based approaches to statistical machine translation (and their successors) extract pairs of (e, f) phrases from automatically word-aligned parallel sentences. **?** described various heuristics for extracting phrase alignments from the Viterbi word-level alignments that are estimated using Brown et al. [1993] word-alignment models.

The phrase extraction heuristics for phrase-based machine translation have been extended so that they extract synchronous grammar rules [Galley et al., 2004, Chiang, 2005, Zollmann and Venugopal, 2006, Liu et al., 2006]. Most of these extraction methods require that one side of the parallel corpus be parsed. This is typically done automatically with a statistical parser.

Figure 2.1 shows examples of rules obtained from a sentence pair. To extract a rule, we first choose a source side span f like *das leck*. Then we use phrase extraction techniques to find target spans e that are consistent with the word alignment (in this case *the leak* is consistent with our f). The nonterminal symbol that is the left-hand side of the SCFG rule is then determined by the syntactic constituent that dominates e (in this case *NP*). To introduce nonterminals into the right-hand side of the rule, we can apply rules extracted over sub-phrases of f, synchronously substituting the corresponding nonterminal symbol for the sub-phrases on both sides. The synchronous substitution applied to f and e then yields the correspondency $\sim$.

One significant differentiating factor between the competing ways of extracting SCFG rules is whether the extraction method generates rules only for constituent phrases that are dominated by a node in the parse tree [Galley et al., 2004, Cohn and Lapata, 2008] or whether they include arbitrary phrases, including non-constituent phrases [Zollmann and Venugopal, 2006, Callison-Burch, 2008]. We adopt the extraction for all phrases, including non-constituents, since it allows us to cover a much greater set of phrases, both in translation and paraphrasing.

However, in doing so we can only assign left-hand side labels to a small portion of the possible phrases in a sentence pair. To allow for broader coverage, we rely on the labeling method introduced by Zollmann and Venugopal [2006]: in addition to single constituent nonterminals, we allow for the concatenation of constituents as well as for CCG-style slashed constituents [Steedman, 1999].

2.2.4 Feature Functions

Rather than assigning a single weight w, we define a set of feature functions $\vec{\varphi} = \{\varphi_1 ... \varphi_N\}$ that are combined in a log-linear model:

$$w = -\sum_{i=1}^{N} \lambda_i \log \varphi_i. \tag{2.11}$$

The weights $\vec{\lambda}$ of these feature functions are set to maximize some objective function like BLEU [Papineni et al., 2002] using a procedure called minimum error rate training (MERT), owing to Och [2003]. MERT iteratively adjusts the weights until the decoder produces output that best matches reference translations in a development set, according to the objective function. We will examine appropriate objective functions for text-to-text generation tasks in Section 4.2.2.

Typical features used in statistical machine translation include phrase translation probabilities (calculated using maximum likelihood estimation over all phrase pairs enumerable in the parallel corpus), word-for-word lexical translation probabilities (which help to smooth sparser phrase translation estimates), a "rule application penalty" (which governs whether the system prefers fewer longer phrases or a greater number of shorter phrases), and a language model probability.

- Phrase translation probabilities $\varphi_{phrase}(e|f)$ and $\varphi_{phrase}(f|e)$ which are computed using maximum likelihood estimation over all phrase pairs that are extracted from the parallel corpus.

- Lexical translation probabilities

$$\varphi_{lex} = \langle -\log p_{lex}(e|f), -\log p_{lex}(f|e) \rangle$$

- Count features c_{src} and c_{tgt} indicating the number of words on either side of the rule as well as two difference features, $c_{dcount} = c_{tgt} - c_{src}$ and the analoguosly computed difference in the average word length in characters, c_{davg}.

- Language model probability.

- A length penalty.

- An indicator for when a rule only contains terminal symbols (δ_{lex}) and an indicator for when the source side contains terminals, but the target side does not (δ_{del}).

- Indicators for whether the rule swaps the order of two nonterminals ($\delta_{reorder}$) and whether the two nonterminals are adjacent (δ_{adj}).

- A rarity penalty $\varphi_{rarity} = e^{(1-c_{rule})}$ that quantifies the doubt we may place in a rule based on how often we have encountered it in the corpus[2].

2.2.5 Decoding

Given an SCFG and an input source sentence, the decoder performs a search for the single most probable derivation via the CKY algorithm. In principle the best

[2]Since we do not have an immediate rule count for a paraphrase rule $N \rightarrow e_1|e_2$, we instead estimate its rarity penalty as $\varphi_{rarity}(N \rightarrow e_1|e_2) = \max_f \min\{\varphi_{rarity}(N \rightarrow e_1|f), \varphi_{rarity}(N \rightarrow f|e_2)\}$

translation should be the English sentence e that is the most probable after summing over all $d \in D$ derivations, since many derivations yield the same e. In practice, we use a Viterbi approximation and return the translation that is the yield of the single best derivation:

$$\begin{aligned} \hat{e} &= \arg \max_{e \in Trans(f)} \sum_{d \in D(e,f)} p(d, e|f) \\ &\approx yield(\arg \max_{d \in D(e,f)} p(d, e|f)). \end{aligned} \tag{2.12}$$

Derivations are simply successive applications of the SCFG rules such as those given in Figure 2.2.

$NP/NN \rightarrow$ dem rest des | the rest of the
$VB + JJ \rightarrow$ gefährlich werden | be dangerous
$NP \rightarrow NP/NN$ dorfes | NP/NN village
$VP/PP \rightarrow$ nicht $VB + JJ$ können | can not $VB + JJ$
$S \rightarrow$ sie NP VP/PP | they VP/PP to NP

A variety of different types of corpora (and semantic equivalence cues) have been used to automatically induce paraphrase collections for English [Madnani and Dorr, 2010]. Perhaps the most natural type of corpus for this task is a monolingual parallel text, which allows sentential paraphrases to be extracted since the sentence pairs in such corpora are perfect paraphrases of each other [Barzilay and McKeown, 2001, Pang et al., 2003]. While rich syntactic paraphrases have been learned from monolingual parallel corpora, they suffer from very limited data availability and thus have poor coverage.

Other methods obtain paraphrases from raw monolingual text by relying on distributional similarity [Lin and Pantel, 2001, Bhagat and Ravichandran, 2008]. While

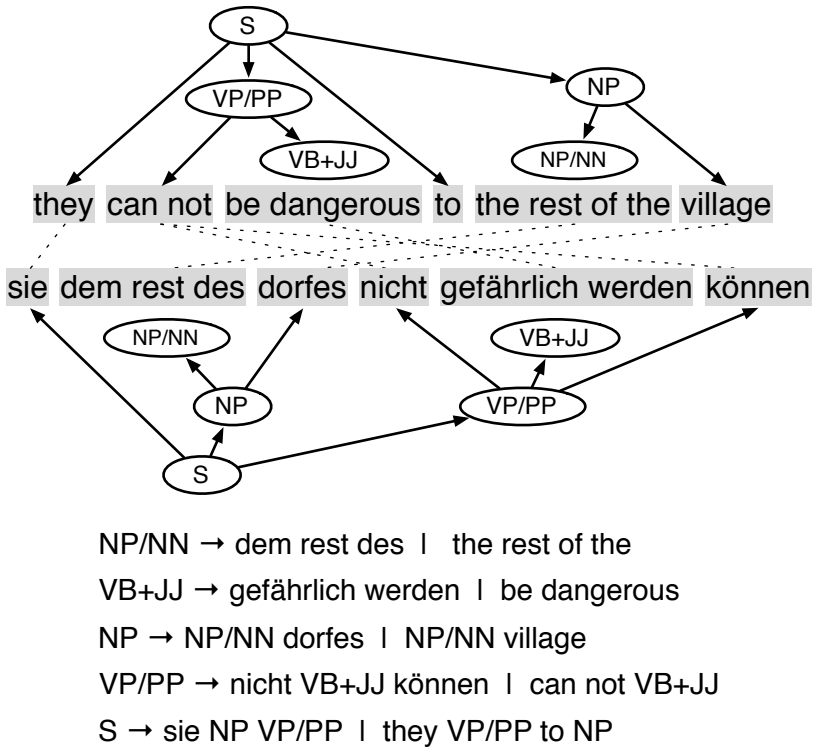

Figure 2.2: An example derivation produced by a syntactic machine translation system. Although the synchronous trees are unlike the derivations found in the Penn Treebank, their yield is a good translation of the German.

vast amounts of data are readily available for these approaches, the distributional similarity signal they use is noisier than the sentence-level correspondency in parallel corpora and additionally suffers from problems such as mistaking cousin expressions or antonyms (such as $\{boy, girl\}$ or $\{rise, fall\}$) for paraphrases.

Abundantly available bilingual parallel corpora have been shown to address both these issues, obtaining paraphrases via a pivoting step over foreign language phrases [Bannard and Callison-Burch, 2005]. The coverage of paraphrase lexica extracted

from bitexts has been shown to outperform that obtained from other sources [Zhao et al., 2008a]. While there have been efforts pursuing the extraction of more powerful paraphrases [Madnani et al., 2007, Callison-Burch, 2008, Cohn and Lapata, 2008, Zhao et al., 2008b], it is not yet clear to what extent meaningful syntactic paraphrases can be induced from bitexts. In this thesis we propose that high-coverage and -quality collections of syntactic paraphrases can indeed be extracted from bilingual parallel data.

Chapter 3

Paraphrasing as Monolingual Syntactic Machine Translation

In this work we approach paraphrasing as monolingual syntactic machine translation. Our method is an extension of two prominent lines of work on paraphrasing, which cast it as phrase-based machine translation [Bannard and Callison-Burch, 2005] and hierarchical machine translation [Madnani et al., 2007]. Like this past work, we extract paraphrases from bilingual parallel corpora. Our work expands on these methods to extract syntactic paraphrases.

Paraphrase extraction using bilingual parallel corpora was proposed by Bannard and Callison-Burch [2005] who induced paraphrases using techniques from *phrase-based* statistical machine translation [Koehn et al., 2003]. After extracting a bilingual phrase table, English paraphrases are obtained by pivoting through foreign language

phrases.

Since many paraphrases can be extracted for a phrase, Bannard and Callison-Burch [2005] rank them using a paraphrase probability defined in terms of the translation model probabilities $p(f|e)$ and $p(e|f)$:

$$p(e_2|e_1) = \sum_f p(e_2, f|e_1) \tag{3.1}$$

$$= \sum_f p(e_2|f, e_1)p(f|e_1) \tag{3.2}$$

$$\approx \sum_f p(e_2|f)p(f|e_1). \tag{3.3}$$

Madnani et al. [2007] extended the technique to *hierarchical* phrase-based machine translation [Chiang, 2005], which is formally a synchronous context-free grammar (SCFG) and thus can be thought of as a *paraphrase grammar*. Like Hiero, Madnani's model uses just one nonterminal X instead of linguistic nonterminals.

In this chapter we extend the bilingual pivoting approach to paraphrase induction to produce rich syntactic paraphrases.[1] We expect syntactic paraphrases to be able to generally capture meaning-preserving English transformations without overgenerating. We describe the formalisms and techniques used in syntactically informed statistical machine translation, and extend them to apply to our paraphrases. Furthermore, we perform a thorough analysis of the types of paraphrases we obtain and discuss the paraphrastic transformations we are capable of capturing.

[1] Chapters 3 and 4 extend the exposition and analysis presented in Ganitkevitch et al. [2011]. The experimental results extend the published work using identical techniques.

3.1 Syntax-Based Machine Translation

A variety of different approaches to including syntactic information in machine translation have been proposed. Here, we use synchronous context free grammars (SCFGs). We extend their use to paraphrasing.

3.1.1 SCFGs in Translation

The model we use in our paraphrasing approach is a syntactically informed *synchronous context-free grammar* (SCFG). The SCFG formalism [Aho and Ullman, 1972] was repopularized for statistical machine translation by Chiang [2005]. Formally, a *probabilistic* SCFG $\mathcal{G}$ is defined by specifying

$$\mathcal{G} = \langle \mathcal{N}, \mathcal{T}_S, \mathcal{T}_T, \mathcal{R}, S \rangle,$$

where $\mathcal{N}$ is a set of nonterminal symbols, $\mathcal{T}_S$ and $\mathcal{T}_T$ are the source and target language vocabularies, $\mathcal{R}$ is a set of rules and $S \in \mathcal{N}$ is the root symbol. The rules in $\mathcal{R}$ take the form:

$$C \rightarrow \langle \gamma, \alpha, \sim, w \rangle,$$

where the rule's left-hand side $C \in \mathcal{N}$ is a nonterminal, $\gamma \in (\mathcal{N} \cup \mathcal{T}_S)^*$ and $\alpha \in (\mathcal{N} \cup \mathcal{T}_T)^*$ are strings of terminal and nonterminal symbols with an equal number of nonterminals $c_{NT}(\gamma) = c_{NT}(\alpha)$ and

$$\sim: \{1 \ldots c_{NT}(\gamma)\} \rightarrow \{1 \ldots c_{NT}(\alpha)\}$$

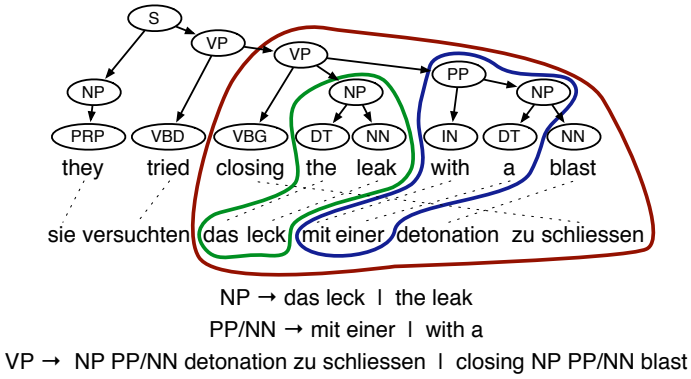

Figure 3.1: Synchronous grammar rules for translation are extracted from sentence pairs in a bitext which have been automatically parsed and word-aligned. Extraction methods vary on whether they extract only minimal rules for phrases dominated by nodes in the parse tree, or more complex rules that include non-constituent phrases.

constitutes a one-to-one correspondency function between the nonterminals in γ and α. A non-negative weight $w \geq 0$ is assigned to each rule, reflecting the likelihood of the rule.

3.1.2 Rule Extraction

Phrase-based approaches to statistical machine translation (and their successors) extract pairs of phrases (e, f) from automatically word-aligned parallel sentences in a given foreign language F and English (E). Koehn [2010] describes a variety of heuristics for extracting phrase alignments from the Viterbi word-level alignments that are commonly estimated using word-alignment models [Brown et al., 1993].

These phrase extraction heuristics have been extended so that they extract synchronous grammar rules [Galley et al., 2004, Chiang, 2005, Zollmann and Venugopal, 2006, Liu et al., 2006]. Most of these extraction methods require that one side of the parallel corpus be parsed. This is typically done automatically with a statistical parser.

Figure 3.1 shows examples of rules obtained from a sentence pair. To extract a rule, we first choose a source side span f like *das leck*. Then we use phrase extraction techniques to find target spans e that are consistent with the word alignment (in this case *the leak* is consistent with our f). The nonterminal symbol that is the left-hand side of the SCFG rule is then determined by the syntactic constituent that dominates e (in this case *NP*). To introduce nonterminals into the right-hand side of the rule, we can apply rules extracted over sub-phrases of f, synchronously substituting the corresponding nonterminal symbol for the sub-phrases on both sides. The synchronous substitution applied to f and e then yields the correspondency $\sim$.

One significant differentiating factor between the competing ways of extracting SCFG rules is what nonterminal labels are considered valid for an extracted rule. Work like that of Galley et al. [2004] or Cohn and Lapata [2008] only considers a rule valid if all nonterminals are full syntactic constituents, i.e. the span covered is dominated by a node in the parse tree. In doing so, valid nonterminal labels can only be assigned to a small portion of the possible phrases in a sentence pair. To allow for broader coverage, approaches like Zollmann and Venugopal [2006] and Callison-

Burch [2008] consider more complex labeling schemes that allow for concatenated constituents as nonterminals, or CCG-style slashed labels [Steedman, 1999].

3.1.3 Feature Functions

In statistical machine translation, rather than assigning a single weight w to each rule, a set of feature functions $\vec{\varphi} = \{\varphi_1 ... \varphi_N\}$ are combined in a log-linear model:

$$w = -\sum_{i=1}^{N} \lambda_i \log \varphi_i. \tag{3.4}$$

The weights $\vec{\lambda}$ of these feature functions are set to maximize some objective function like Bleu [Papineni et al., 2002] using a procedure called minimum error rate training (MERT), owing to Och [2003]. MERT iteratively adjusts the weights until the decoder produces output that best matches reference translations in a development set, according to the objective function.

Typical features used in statistical machine translation include phrase translation probabilities (calculated using maximum likelihood estimation over all phrase pairs enumerable in the parallel corpus), word-for-word lexical translation probabilities (which help to smooth sparser phrase translation estimates), a "rule application penalty" (which governs whether the system prefers fewer longer phrases or a greater number of shorter phrases), and a language model probability. Syntax-based machine translation systems typically extend this feature set by adding translation probability estimates that are conditioned on syntactic information.

The following list gives a more detailed overview of the features used in the Joshua decoder:

- Phrase translation probabilities $p_{phrase}(e \mid f)$ and $p_{phrase}(f \mid e)$, which are computed using maximum likelihood estimation (MLE) over all phrase pairs that are extracted from the parallel corpus.

- Syntactically informed translation probabilities $p(e \mid f, C)$, $p(f \mid e, C)$, that additionally condition on the nonterminal label C. These features are computed as maximum likelihood estimates, but are limited to corpus occurrences of e and f that were dominated by C in the syntactic parse.

- Syntactic parsing probabilities $p(e \mid C)$, $p(f \mid C)$, $p(C \mid e)$, and $p(C \mid f)$, similarly computed from the parsed bitext via MLE.

- Lexical translation probabilities $p_{lex}(e \mid f)$ and $p_{lex}(f \mid e)$, which are based on the word alignment

$$A = \{a_{ij} \in \{0, 1\} \mid i \in \{1 \ldots n\}, j \in \{1 \ldots m\}\}$$

between $e = e_1 \ldots e_n$ and $f = f_1 \ldots f_m$. The lexical translation probability for the rule is computed from the word-level translation probabilities $p_{word}(e_i \mid f_j)$ and $p_{word}(f_j \mid e_i)$:

$$p_{lex}(e \mid f) = \prod_{i=0}^{n} \frac{1}{\sum_{j=0}^{m} a_{ij}} \sum_{j=1}^{m} p_{word}(e_i \mid f_j) \tag{3.5}$$

The word probabilities, in turn, are maximum likelihood estimates over the aligned bitext.

- The language model probability of the rule's target side e.
- A rarity penalty $\varphi_{rarity} = e^{(1-c_{rule})}$ that allows us to down-weigh a rule that only appears in the corpus a few times. The rarity penalty is 1 for singleton rules, and quickly drops off towards 0 as the number of rule occurrences in the parallel text increases.
- An log-occurrence count feature $\log c$ that permits the decoder to express preference for frequent, well-estimated rules.
- Word count features c_{src} and c_{tgt} indicating the number of words on either side of the rule. This allows the system to learn a preference for longer or shorter rules.
- A rule application feature c_{rule} that is always 1 and counts the number of rules applied in a synchronous derivation. This permits the decoder to prefer longer, more complex parse trees over flatter ones – or vice versa.

Furthermore, machine translation systems like the Joshua decoder often consider an array of boolean indicator features that describe structural aspects about a rule, enabling the system to learn preferences for operations like reordering, or deletion of words:

- δ_{lex} for when a rule only contains terminal symbols on the right-hand side.
- $\delta_{abstract}$ for rules that do not contain any terminal symbols.
- δ_{del} for when the source side contains terminals, but the target side does not.
- δ_{gen} for rules where the target side contains terminals but the source side does not.
- $\delta_{src-unal}$ and $\delta_{tgt-unal}$ for when there are unaligned words on the source or target side, respectively.
- Indicators for whether the rule swaps the order of two nonterminals ($\delta_{reorder}$) and whether there are two adjacent nonterminals (δ_{adj}).
- δ_X for whether the rule contains a wildcard nonterminal.
- $\delta_{identity}$, which fires when the source and target sides are identical.

3.1.4 Decoding

Given an SCFG and an input foreign sentence, the decoder performs a search for the single most probable derivation via the CKY algorithm. In principle the best translation should be the English sentence e that is the most probable after summing over all $d \in D$ derivations, since many derivations yield the same e. In practice, we use a Viterbi approximation and return the translation that is the yield of the single

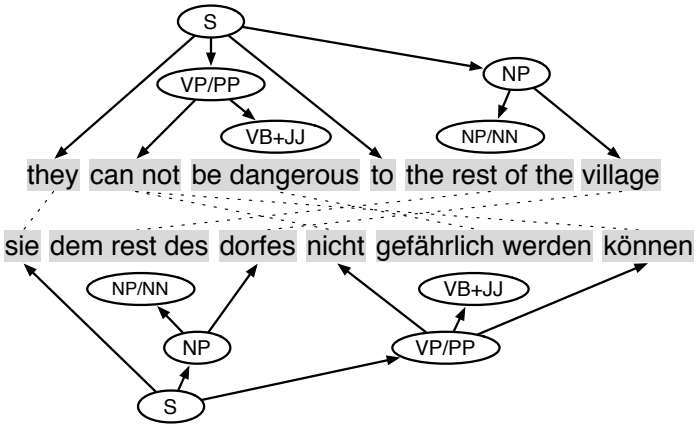

Figure 3.2: An example derivation produced by a syntactic machine translation system. Although the synchronous trees are unlike the derivations found in the Penn Treebank, their yield is a good translation of the German.

best derivation:

$$\begin{aligned} \hat{e} &= \arg \max_{e \in Trans(f)} \sum_{d \in D(e,f)} p(d, e|f) \\ &\approx yield(\arg \max_{d \in D(e,f)} p(d, e|f)). \end{aligned} \tag{3.6}$$

Synchronous derivations, like the one shown in Figure 3.2, are obtained by successive application of a set of SCFG rules. Figure 3.2 for instance is obtained by applying the below rule set:

$NP/NN \rightarrow$ dem rest des | the rest of the
$VB + JJ \rightarrow$ gefährlich werden | be dangerous
$NP \rightarrow NP/NN$ dorfes | NP/NN village
$VP/PP \rightarrow$ nicht $VB + JJ$ können | can not $VB + JJ$
$S \rightarrow$ sie NP VP/PP | they VP/PP to NP

3.2 Syntactic Paraphrases from Bilingual Data

The novel contribution of this thesis is to extend the notion of paraphrases as SCFGs to include syntactically informed SCFGs. Bannard and Callison-Burch [2005] used techniques from phrase-based machine translation to extract lexical and phrasal paraphrases. *Lexical paraphrases*, or synonyms, are pairs of words that, in an appropriate context, share a meaning:

committee | board
proposal | draft

Similarly, *phrasal paraphrases* denote pairs of contiguous surface text strings with the same meaning:

the commission's role | the role of the commission

Phrasal paraphrases can capture a wide range of meaning-preserving transformations. However they do so by memorization: having seen only the above phrase, we would not be able to learn anything general about the English possessive rule. Madnani et al. [2007] add generalization capabilities by introducing nonterminals (or *slots*) into the surface forms, resulting in *paraphrase patterns*:

$$X \rightarrow \text{the } X_1\text{'s } X_2 \mid \text{the } X_2 \text{ of the } X_1.$$

The introduction of nonterminals into the paraphrasing formalism also allows for reordering, as shown by the nonterminal indices. By adding a left-hand side nonterminal

(X, again), Madnani et al. [2007] extend the paraphrase collection to a synchronous context-free grammar (SCFG). The extension to SCFGs allows paraphrase generation and recognition via synchronous parsing, nesting paraphrase patterns into a sentence-level parse tree in a way similar to SCFG-based machine translation as popularized by Chiang [2005].

The nonterminals added by Madnani et al. [2007] are wildcards: they make no restrictions on what can fill a slot. This can lead to significant over-generation. To gain a better notion of fitting syntactic types while maintaining generalization capabilities, we annotates the nonterminals with syntactic constraints, yielding *syntactic paraphrases*:

$$NP \rightarrow \text{the } NP_1\text{'s } NP_2 \mid \text{the } NP_2 \text{ of the } NP_1.$$

It is evident that the syntactically labeled paraphrases have a much higher potential for generalization and for capturing interesting paraphrastic transformations than phrasal paraphrases. Yet they are more likely to produce grammatically correct results than unconstrained wildcard paraphrase patterns. This is especially interesting for applications in sentential paraphrasing, where syntactic information can lead to higher precision synchronous parses.

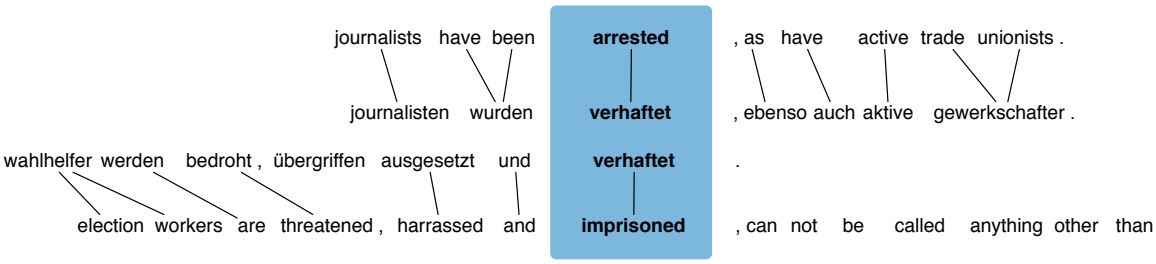

Figure 3.3: An example of phrasal pivoting: the word "arrested" is seen aligned to the German word "verhaftet." In another sentence pair in the bitext, "verhaftet" aligns to the English "imprisoned." By pivoting over the shared German translation, the two English words can be extracted as a pair of paraphrases.

3.2.1 Rule Extraction

Here we use bilingual parallel text to extract paraphrase rules. Our approach builds on the work of Bannard and Callison-Burch [2005] and Madnani et al. [2007]. Bannard and Callison-Burch [2005] extract English paraphrases from English-to-foreign bitext data by assuming that two English phrases that are observed translating to the same foreign phrase are paraphrases. Figure 3.3 illustrates this idea: in one parallel sentence pair, the word "arrested" is seen aligned to the German word "verhaftet." In another sentence pair in the bitext, "verhaftet" aligns to the English "imprisoned." Bannard and Callison-Burch [2005] propose that by *pivoting* over the German word we can extract the paraphrase pair:

arrested | imprisoned.

In order to both make this pivot-based extraction more practicable, a typical implementation of pivot-based paraphrase extraction will not operate directly on the bitext, but on a translation phrase table, i.e. a collection of English and foreign phrases that have been seen as translations of one another, annotated with the typical features used in phrase-based machine translation like their occurrence counts and translation probabilities Koehn [2010]:

$$PT = \{\langle f, e, \vec{\varphi}\rangle\}.$$

Bannard and Callison-Burch [2005] find rule pairs with matching foreign phrases in the phrase table:

$$\langle f, e_1, \vec{\varphi}_1\rangle \text{ and } \langle f, e_2, \vec{\varphi}_2\rangle,$$

and combined them to create phrasal paraphrase rules:

$$\langle e_1, e_2, \vec{\varphi}\rangle.$$

Later work extends this phrasal pivoting approach to include some syntactic information. Callison-Burch [2008] generate syntactic parses for the English side of the bitext. They then keep track of the syntactic constituent C that dominates the English phrase e in the parallel corpus. This yields a syntactically annotated phrase table:

$$PT_{syn} = \{\langle f, e, C, \vec{\varphi}\rangle\}.$$

The pivot rule pairs now are required to have both matching foreign phrases, and syntactic labels C:

$$\langle f, e_1, C, \vec{\varphi}_1 \rangle \text{ and } \langle f, e_2, C, \vec{\varphi}_2 \rangle,$$

there yield syntax-annotated phrasal paraphrase rules:

$$\langle e_1, e_2, C, \vec{\varphi} \rangle.$$

Madnani et al. [2007] extend the pivoting approach to unlabeled SCFGs. The pivoting notion remains the same in their work, but instead of a translation phrase table now builds on an unlabeled translation SCFG. They therefore pair up two rules of the form:

$$X \rightarrow \langle f, e_1, \sim_1, \vec{\varphi}_1 \rangle$$

$$X \rightarrow \langle f, e_2, \sim_2, \vec{\varphi}_2 \rangle,$$

to generate a paraphrase rule:

$$X \rightarrow \langle e_1, e_2, \sim, \vec{\varphi} \rangle,$$

where the nonterminal correspondency relation $\sim$ is been set to reflect the combined nonterminal alignment:

$$\sim \quad = \quad \sim_1^{-1} \circ \sim_2 .$$

This composition reflects the way the nonterminal correspondence changes through the pivot. Since $\sim$ is defined from source to target, $\sim_1$ is reversed (going from e_1 to f) and composed with $\sim_2$ (going from f to e_2).

In our work we use a bitext in which the English side is syntactically parsed. However, we aim to syntactically labeled SCFGs. This expands on both Madnani et al. [2007] and Callison-Burch [2008]. Our extraction method uses a syntactically labeled translation SCFG learned from the bitext as its basis. To create a paraphrase grammar from a translation grammar, we use syntax as a constraint for pivoting. For each pair of translation rules with English strings e_1 and e_2, where the left-hand side label C and the foreign string f match:

$$C \rightarrow \langle f, e_1, \sim_1, \vec{\varphi}_1 \rangle$$

$$C \rightarrow \langle f, e_2, \sim_2, \vec{\varphi}_2 \rangle,$$

we create a paraphrase rule:

$$C \rightarrow \langle e_1, e_2, \sim, \vec{\varphi} \rangle.$$

The nonterminal correspondency $\sim$ is computed analogously to Madnani et al. [2007].

To illustrate the pivoting process, we consider the following German-English translation rules with matching source sides:

$$NN \rightarrow \text{rede} \mid \text{address}$$
$$NN \rightarrow \text{rede} \mid \text{presentation}$$
$$VP\backslash PP \rightarrow \text{rede} \mid \text{am discussing .}$$

Even though all source phrases match, we may only combine the first two of the translation rules. The third rule's left-hand side doesn't match, and in this case that is indeed indicative of a syntactic mismatch on the English side – "am discussing" is not a valid substitute for either "presentation" or "address." By pivoting over the matching rules, we thus obtain the following paraphrases:

$$NN \rightarrow \text{address} \mid \text{presentation}$$
$$NN \rightarrow \text{presentation} \mid \text{address}.$$

Similarly, we can mutually combine the syntactic translation rules

$$VP \rightarrow NP_1\ NP_2 \text{ geben} \mid \text{give } NP_1\ NP_2$$
$$VP \rightarrow NP_1\ NP_2 \text{ geben} \mid \text{give } NP_2 \text{ to } NP_1$$
$$VP \rightarrow NP_1\ NP_2 \text{ geben} \mid \text{provide } NP_1 \text{ with } NP_2$$

to generate the paraphrase rules. Because the translation grammar is synchronous, our use of matching foreign sides as a constraint guarantees that the nonterminal labels between the three English sides will match: each above English target phrase is guaranteed to have two NP-labeled nonterminals. Through pivoting, we obtain the following paraphrase rules:

$$VP \rightarrow \text{give } NP_1\ NP_2 \mid \text{give } NP_2 \text{ to } NP_1$$
$$VP \rightarrow \text{give } NP_1\ NP_2 \mid \text{provide } NP_1 \text{ with } NP_2$$

$$VP \rightarrow \text{give } NP_1 \text{ to } NP_2 \mid \text{give } NP_2\ NP_1$$
$$VP \rightarrow \text{give } NP_1 \text{ to } NP_2 \mid \text{provide } NP_2 \text{ with } NP_1$$

$$VP \rightarrow \text{provide } NP_1 \text{ with } NP_2 \mid \text{give } NP_1\ NP_2$$
$$VP \rightarrow \text{provide } NP_1 \text{ with } NP_2 \mid \text{give } NP_2 \text{ to } NP_1\ .$$

Note that in our notation the source phrase determines the indexing of the nonterminals in order of appearance, while the indexing on the target phrase indicates whether a reordering is taking place. When compared to the others, the phrase "give NP_1 to NP_2" reverses the order of the direct and indirect object. The paraphrases that include it therefore are reordering ones, while the phrases with matching order maintain the nonterminal order without reordering.

3.2.2 Assigning Syntactic Labels to SCFG Rules

As described above, our syntactic paraphrase rules are obtained from an intermediate syntactically labeled translation SCFG. The types of labels we will see in the paraphrase grammar therefore are fully determined by the labeling in the extraction of the translation grammar. While considerable efforts have been made to compare syntactic rule extraction and labeling strategies for machine translation, the same conclusions cannot be assumed to hold for paraphrasing. We therefore consider and compare a variety of labeling strategies for our syntactic paraphrases.

Typically, during the initial rule extraction we are considering a span and seek to produce a label for it. If it is possible to label the span in accordance with our labeling strategy, we can replace the span with the corresponding nonterminal symbol and proceed. If no valid label is found, we reject the rule.

For our paraphrases, we seek a labeling strategy that strikes a balance between having a well-formed and not overly sparse set of nonterminals, and maintaining good coverage of the data (i.e. not throwing away too many rules).

To this end, we evaluate a tiered approach. First we only allow full Penn Treebank constituents as nonterminal labels. If the span is exactly dominated by a node in the constituency parse tree, we will use that node as our label. Next, we expand to CCG-style slashed categories. If a full constituent cannot be found, we consider if the span can be labeled as a constituent subtraction, for example "an *VP* missing a *NP* to its right." Finally, we can add concatenations of constituent spans to our

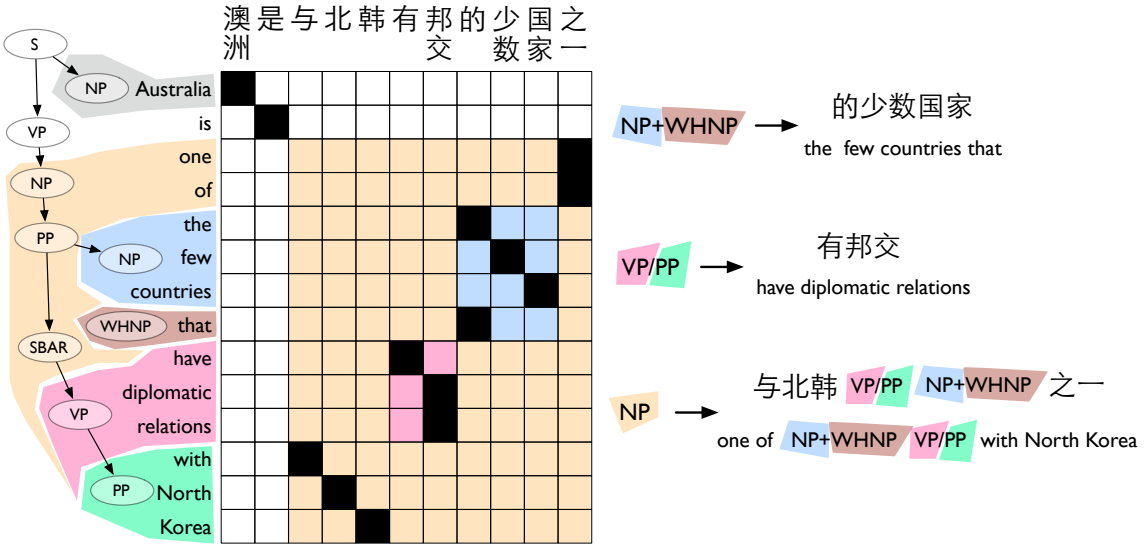

Figure 3.4: Another example of the extraction of syntactically labeled rules from an aligned bitext. The phrase "the few countries that" is extracted along with its translation. Since no one constituent covers this span, it is labeled with a concatenative nonterminal symbol, $NP + WHNP$. Similarly, "have diplomatic relations" is labeled VP/PP. Both these phrase can be cut out of the enveloping NP "one of the few countries that have diplomatic relations with North Korea" and replaced by their respective left-hand side nonterminals to form a complex rule.

pool of valid labels. If neither full nor slashed constituents suffice to label a span, we consider concatenations of up to three constituents as a label. This follows the SAMT labeling algorithm [Zollmann and Venugopal, 2006].

Figure 3.4 illustrates this extraction and labeling process: when possible, we choose simple single-constituent labels. If a nonterminal's span does not match a full constituent we back off to more complex labels to maintain high coverage.

3.2.3 Feature Functions

Having extracted a paraphrase rule, we next turn to computing the rule features $\vec{\varphi}$. The feature vector can contain different types of features, as described in Section **??**. Broadly, we can distinguish three feature types: rule type features, features that quantify goodness, and task-centric utility features.

The features that describe the rule type are primarily useful in helping the paraphrasing system learn to guide its search on a global level: are longer rules preferable to shorter rules? Are more complex derivations better than simple left- or right-branching trees? Do we trust rare rules? As such, the rule type features can include information such as the number of lexical tokens in e_1 or e_2, the rule's arity, or the number of times the rule has occurred in the supporting data. In addition to the type-centric features commonly used in translation systems, we add a boolean indicator for whether the rule is an identity paraphrase, $\delta_{identity}$. Another indicator feature, $\delta_{reorder}$, fires if the rule swaps the order of two nonterminals, which enables us to

promote more complex paraphrases that require structural reordering.

Utility-centric features for a paraphrase rule will quantify the rule's contribution to achieving a specific quality in the paraphrasing task at hand. This group of features depends on the specific text-to-text generation task. We discuss a set of example features for sentence compression in Section 4.2.1.

The main concern of this section is to find a way to reliably quantify the goodness of a paraphrase rule, i.e. the meaning preservation between e_1 and e_2. To achieve this, we look to estimate a conditional *phrasal paraphrase probability* $p(e_2|e_1)$ for the paraphrase rule, i.e. the probability that given e_1 we can substitute e_2 without altering the sentence's meaning. This is analogous to the phrasal translation probability feature $p(e|f)$ used in machine translation.

Additionally, incorporating syntactic information into our paraphrases (by way of the rule label C) allows us to consider features that quantify syntactic well-formedness, like $p(e_1|C)$ and $p(C|e_1)$. Furthermore, by conditioning on both C and e_1, we can use the syntax to disambiguate: the *syntactic paraphrase probability* $p(e_2|e_1, C)$ quantifies the likelihood of e_1 paraphrasing as e_2 when it is labeled C.

While it is trivial to look up the $p(e_i|C)$ and $p(C|e_i)$ from the translation grammar, estimating the phrasal and syntactic paraphrase probabilities is a more challenging task. We consider two approaches: pivot-based probability estimation, and probability estimation via virtual counts.

3.2.3.1 Feature Overview

The features we estimate for each paraphrase rule are related to features typically used in machine translation systems. As such, we follow traditional SMT notation in designating the input phrase as f and its paraphrase as e. To estimate the count- and probability-based features, we rely on Equation 3.3. Following the log-linear feature model, the resulting (un-normalized) probability estimates, like $p(e|f)$, are stored as their negative logarithm $-\log p(e|f)$. In detail, the features computed for our paraphrases are as follows:

3.2.3.1.1 Goodness features:

- Lex(e|f) – the "lexical translation" probability of the paraphrase given the original phrase. This feature is estimated as defined by Koehn et al. [2003].
- Lex(f|e) – the lexical translation probability of the phrase given the paraphrase.
- LogCount – the log of the frequency estimate for this paraphrase pair.
- RarityPenalty – this feature marks rules that have only been seen a handful of times. It is calculated as $\exp(1 - c(e, f))$, where $c(e, f)$ is the estimate of the frequency of this paraphrase pair.
- p(LHS|e) – the probability of the lefthand side nonterminal symbol given the paraphrase.

[VP]	[NP\DT]
accord [NP,1] to [NP,2]	member states of the european union
grant [NP,2] [NP,1]	union countries
Abstract=0	Abstract=0
Adjacent=0	Adjacent=0
GlueRule=0	GlueRule=0
Identity=0	Identity=0
Lex(e|f)=10.68194	Lex(e|f)=19.48047
Lex(f|e)=7.00455	Lex(f|e)=8.36812
Lexical=0	Lexical=1
Monotonic=0	Monotonic=1
p(e|f)=5.13580	p(e|f)=7.24783
p(e|f,LHS)=4.61512	p(e|f,LHS)=8.51143
p(e|LHS)=13.51532	p(e|LHS)=15.06318
p(f|e)=6.23441	p(f|e)=4.87293
p(f|e,LHS)=5.71373	p(f|e,LHS)=3.88156
p(f|LHS)=16.34853	p(f|LHS)=10.43332
p(LHS|e)=0.65678	p(LHS|e)=2.56495
p(LHS|f)=0.51083	p(LHS|f)=0.30999
PhrasePenalty=1	PhrasePenalty=1
RarityPenalty=0.36788	RarityPenalty=0.36788
SourceTerminalsButNoTarget=0	SourceTerminalsButNoTarget=0
SourceWords=2	SourceWords=6
TargetTerminalsButNoSource=0	TargetTerminalsButNoSource=0
TargetWords=1	TargetWords=2
UnalignedSource=1	UnalignedSource=2
UnalignedTarget=0	UnalignedTarget=0

Table 3.1: Two example paraphrase rules extracted using bilingual pivoting. Due to the syntactic annotations, the first rule is capable of accurately capturing a reordering between direct and indirect object. The second rule illustrates how CCG-style slashed labels can help accurately fit larger spans into a syntactic parse.

- p(LHS|f) – the probability of the lefthand side nonterminal symbol given the original phrase.
- p(e|LHS) – the probability of the paraphrase given the lefthand side nonterminal symbol (this is typically a very low probability).
- p(e|f) – the paraphrase probability of the paraphrase given the original phrase, as defined in Equation 3.3.
- p(e|f,LHS) – the probability of paraphrase given the the lefthand side nonterminal symbol and the original phrase.
- p(f|LHS) – the probability of original phrase given the the lefthand side nonterminal (this is typically a very low probability).
- p(f|e) – the paraphrase probability of the original phrase given the paraphrase, as defined in Equation 3.3.
- p(f|e,LHS) – the probability of original phrase given the the lefthand side nonterminal symbol and the paraphrase.

3.2.3.1.2 Type features:

- Abstract – a binary feature that indicates whether the rule is composed exclusively of nonterminal symbols.

- Adjacent – a binary feature that indicates whether rule contains adjacent nonterminal symbols.

- ContainsX – a binary feature that indicates whether the nonterminal symbol X is used in this rule. X is the symbol used in Hiero grammars Chiang [2007], and is sometimes used by our syntactic SCFGs when we are unable to assign a linguistically motivated nonterminal.

- GlueRule – a binary feature that indicates whether this is a glue rule. Glue rules are treated specially by the Joshua decoder Post et al. [2013]. They are used when the decoder cannot produce a complete parse using the other grammar rules.

- Identity – a binary feature that indicates whether the phrase is identical to the paraphrase.

- Lexical – a binary feature that says whether this is a single word paraphrase.

- Monotonic – a binary feature that indicates whether multiple nonterminal symbols occur in the same order (are monotonic) or if they are re-ordered.

- PhrasePenalty – this feature is used by the decoder to count how many rules it uses in a derivation. Turning helps it to learn to prefer fewer longer phrases, or more shorter phrases. The value of this feature is always 1.

- SourceTerminalsButNoTarget – a binary feature that fires when the phrase contains terminal symbols, but the paraphrase contains no terminal symbols.
- SourceWords – the number of words in the original phrase.
- TargetTerminalsButNoSource – a binary feature that fires when the paraphrase contains terminal symbols but the original phrase only contains nonterminal symbols.
- TargetWords – the number of words in the paraphrase.
- UnalignedSource – a binary feature that fires if there are any words in the original phrase that are not aligned to any words in the paraphrase.
- UnalignedTarget – a binary feature that fires if there are any words in the paraphrase that are not aligned to any words in the original phrase.

3.2.3.2 Pivot-Based Probability Estimation

We can directly estimate the paraphrase probabilities from the translation probabilities in the underlying translation SCFG by followingBannard and Callison-Burch [2005]. The idea of pivot-based probability estimation is quite intuitive: we view a French-English translation grammar as a directed bipartite graph. In this graph, the vertices are English and French expressions, and we add a corresponding edge for every translation rule in the grammar, labeled with its probability. To extract

the paraphrases for a phrase e_1, we fully explore the two-hop neighborhood of the corresponding node. Figure 3.5 illustrates this idea.

To compute $p(e_2|e_1)$, we thus start at e_1, and consider an f it may translate into along with the probability of doing so, $p(f|e_1)$. We then look up the probability of f translating to e_2, $p(e_2|f)$, and combine the two. By summing over all possible f, we obtain our probability estimate:

$$p(e_2|e_1) = \sum_f p(e_2|f)p(f|e_1). \tag{3.7}$$

Similarly, we can apply this method to compute an estimate for $p(e_2|e_1, C)$ by conditioning the translation probabilities on C:

$$p(e_2|e_1, C) = \sum_f p(e_2|f, C)p(f|e_1, C). \tag{3.8}$$

The reverse probabilities, $p(e_1|e_2)$ and $p(e_1|e_2, C)$ are computed in analogous fashion.

The advantage of this approach is that we can straightforwardly extract paraphrases and estimate their probabilities from any translation grammar given to us, without needing explicit access to the bitext. In practice, however, a different problem arises: due to the "long tail" of potential translations $\langle e_1, f\rangle$ and back-translations $\langle f, e_2\rangle$, we see a combinatorial explosion of the number of paraphrases extracted, especially for frequently occurring f and e_i.

A straightforward solution to this problem is quickly found – pruning away unlikely translation grammar rules and thereby controlling the volume of paraphrases extracted to only include the high-probability ones. Without proper renormalization

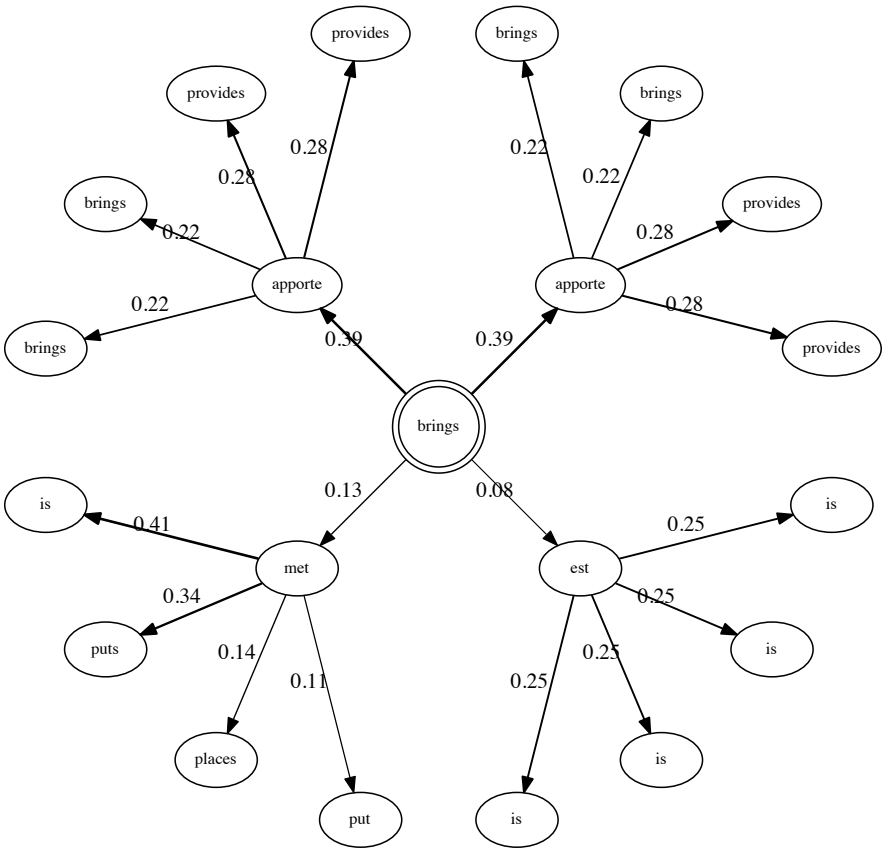

Figure 3.5: An example of the graph representation that is implicit in pivot-based probability estimation. From the source word, here "brings," we pivot into French, and then hop back into English. The edges are labeled with $p(\textit{French} \mid \textit{English})$ and $p(\textit{English} \mid \textit{French})$, respectively. Aggregating over all possible two-hops, we can estimate probabilities for the thusly obtained paraphrases from these translation probabilities.

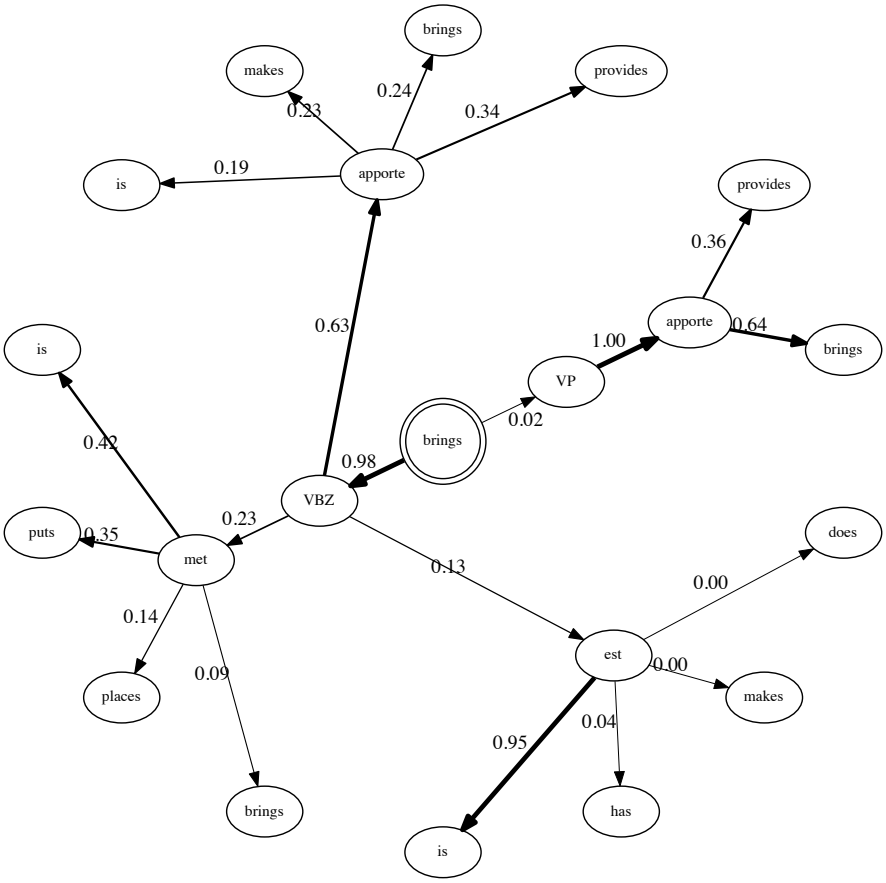

Figure 3.6: A syntax-constrained view of the pivot graph, separating the paraphrases into sub-graphs by syntactic label. Here, the edges are labeled with $p(French \mid English, LHS)$ and $p(English \mid French, LHS)$. The links between the seed phrase "brings" and its known syntactic labels indicate the probability of that syntactic label within the data, $p(LHS \mid English)$.

of the probability estimates $p(e_i|f)$ and $p(f|e_i)$, however, this causes the paraphrase probability estimates to become deficient. The conditional paraphrase probabilities are no longer guaranteed to sum to one: $\sum_j p(e_j|e) \geq 1$. For reasons of efficiency, we choose to skip the renormalization step for paraphrase probabilities in our setup. While this is not a problem for the paraphrases' use in our text-to-text generation setup – the log-linear model does not require its features to be normalized – however other use cases may depend on the proper distributions.

3.2.3.3 Probability Estimation Via Virtual Counts

An alternative means of probability estimation that may offer higher flexibility for the manipulation of paraphrase probabilities was proposed by Madnani et al. [2007]. Instead of relying on the the probability features obtained from the translation grammar, we can use standard maximum-likelihood estimation based on virtual counts $c(e_1, e_2)$:

$$p(e_2|e_1) = \frac{c(e_1, e_2)}{\sum_e c(e_1, e)}. \tag{3.9}$$

The virtual counts, in turn, can be computed in a fashion similar to the probability pivoting. For every f that the English paraphrases e_1 and e_2 have in common in the translation grammar, we aggregate an estimate of many occurrences of $\langle e_1, e_2 \rangle$ we would see in a virtual paraphrase bitext, based on the counts $c(e_1, f)$ and $c(e_2, f)$.

This estimate can be generalized to a function $g(n, m)$:

$$c(e_1, e_2) = \sum_f g(c(e_1, f), c(e_2, f)). \tag{3.10}$$

Madnani et al. [2007] use an upper bound approximation here:

$$g(c(e_1, f), c(e_2, f)) = c(e_1, f) \cdot c(e_2, f). \tag{3.11}$$

This estimate states that after seeing the translation pair $\langle f, e_1 \rangle$ occur n times while the pair $\langle f, e_2 \rangle$ is seen m times, we assume that in the virtual corpus described by these counts the paraphrase pair $\langle e_1, e_2 \rangle$ appears $n \cdot m$ times.

The advantage of using virtual counts is two-fold: firstly, by using the notion of an intermediate function g we can experiment with a variety of count estimation strategies without having to worry about the rest of the paraphrase extraction pipeline. The flexibility of count-based pivoting has the potential to be especially useful when performing tasks like domain-adaptation. These tasks require a consideration of the reliability or domain-appropriateness of the underlying data, or the similarity to a given expected paraphrase probability distribution.

Secondly, when implementing a full paraphrase extraction pipeline the use of virtual counts is simply more efficient. A standard map-reduce-based implementation will extract translation rules with counts and perform several map-reduce steps to calculate the conditional translation probabilities. When using pivot-base probability estimation with renormalization, these steps will have to be repeated on the paraphrase grammar. With virtual count-based estimation, they only need to be

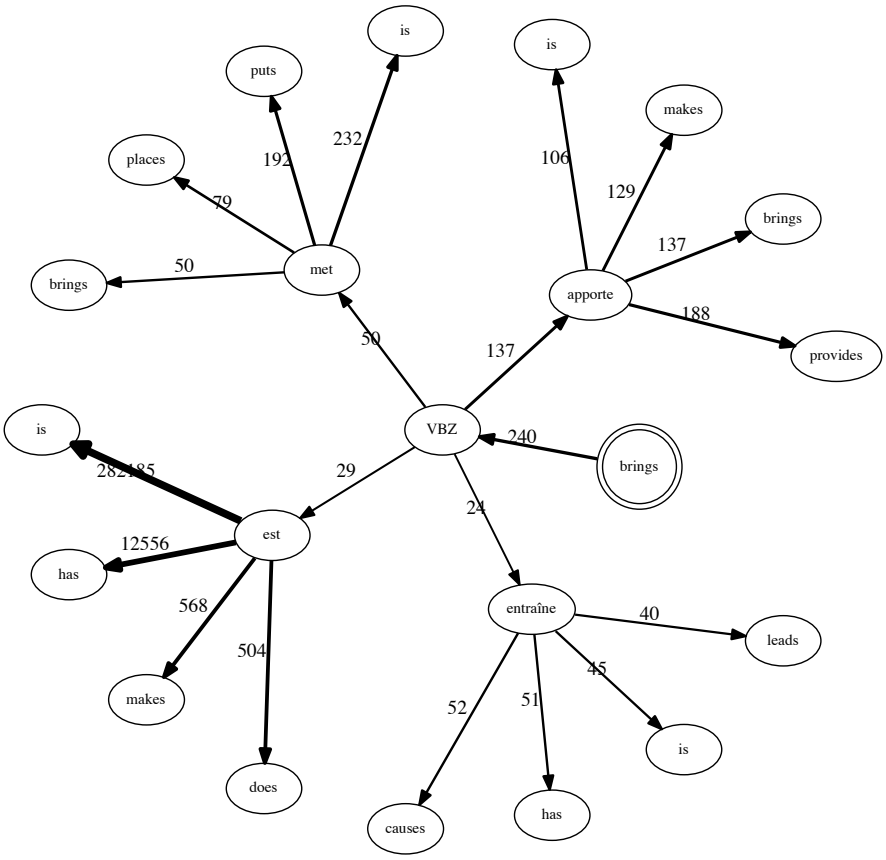

Figure 3.7: The syntax-constrained view of the pivoting graph, this time relying on occurrence counts as done when using virtual counts-based estimation. Note that the inclusion of a extremely frequently occurring translation link, like *est* | *is* will heavily influence the probability estimate of *brings* | *is*.

performed once to yield a properly normalized paraphrase distribution in the output. We discuss details of these implementation differences in Section **??**.

We expect the virtual counts-based probability estimates to behave differently from the pivot-based ones. As illustrated in Figure 3.7, when relying on virtual counts frequently occurring phrases can have a stronger influence over the final paraphrase probability estimate than they would in the normalized pivot-based estimate. We present a qualitative and quantitative comparison of the two estimation techniques in Section 3.4.

3.2.4 Decoding with Paraphrases

In this chapter, we have previously described a view of sentential text-to-text rewriting tasks that restates them as a targeted English-to-English paraphrasing. Applying the syntactic pivoting method described above to extract paraphrases yields a featurized paraphrase SCFG. The formalism of this grammar is identical to the ones used in syntactic machine translation. We therefore can adopt the search algorithm commonly used in statistical machine translation to do paraphrasing:

When approaching a given sentential text-to-text generation task, we seek to find the best paraphrase of the input sentence. Here, we define "best" as "most likely, according to our log-linear model." The log-linear model formulation allows us to straightforwardly combine the features we attach to our paraphrase rules. For instance, it allows us to find a balance between highly likely paraphrases, and para-

phrases that may be less likely but are more appropriate for the rewriting task at hand.

The problem of finding the most likely paraphrase for the whole input sentence can be formulated analogously to the search problem in machine translation (c.f. Equation 3.6):

$$\hat{e}_{out} \approx yield(\arg \max_{d \in D(e_{out}, e_{in})} p(d, e_{out}|e_{in})).$$

Out of all possible paraphrase derivations $d \in D(e_{out}, e_{in})$ that rewrite the input e_{in} into some output e_{out}, we choose the d estimated to be the most likely by our log-linear model. The yield of this max-derivation, $\hat{e}_{out}$, is our paraphrasing system's output.

Figure 3.8 shows an example derivation produced as a result of applying our paraphrase rules in the decoding process. Another advantage of using the decoder from statistical machine translation is that n-gram language models, which have been shown to be useful in natural language generation [Langkilde and Knight, 1998], are already well integrated [Huang and Chiang, 2007].

Unlike in machine translation, the use of identity rules is a complex challenge in paraphrasing. In most translation cases translating (or otherwise transliterating) in input word will always be preferable to leaving it unchanged. Therefore identity rules, unless explicitly present in the translation grammar, are typically heavily penalized. For paraphrasing that is not the case: leaving key terms unchanged may be beneficial to the overall output quality (although it decreases the diversity of the output

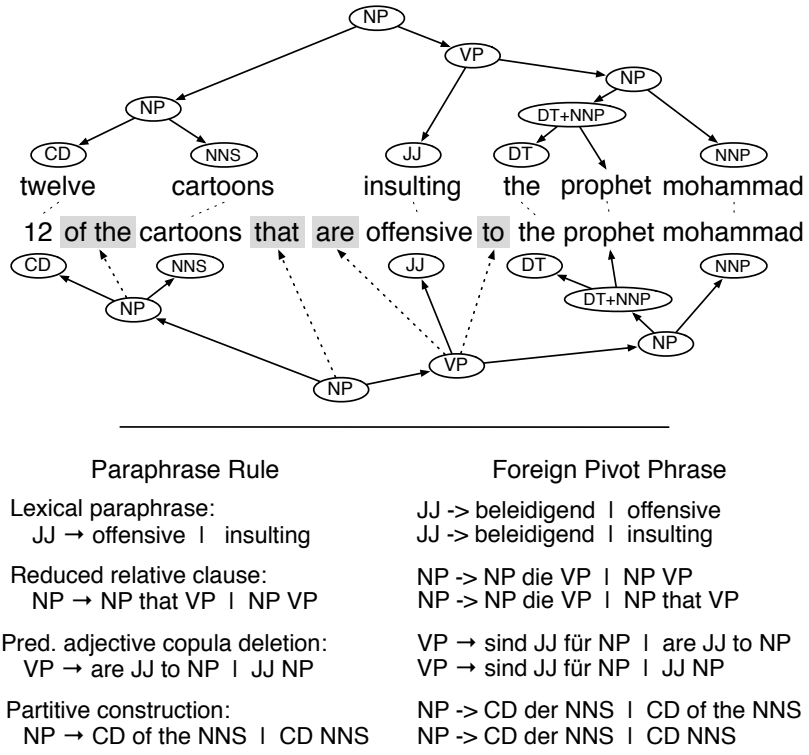

Figure 3.8: An example of a synchronous paraphrastic derivation. A few of the rules applied in the parse are show in the left column, with the pivot phrases that gave rise to them on the right.

paraphrases).

3.3 Analysis of Pivot-Based Syntactic Paraphrases

A key motivation for the use of syntactic paraphrases over their phrasal counterparts is their potential to capture meaning-preserving linguistic transformations in a more general fashion. A phrasal system is limited to memorizing fully lexicalized transformations in its paraphrase table, resulting in poor generalization capabilities. By contrast, a syntactic paraphrasing system intuitively should be able to address this issue and learn well-formed and generic patterns that can be easily applied to unseen data.

To put this expectation to the test, we investigate how our grammar captures a number of well-known paraphrastic transformations.

3.3.1 Examples of Linguistic Transformations

Table 3.2 shows the transformations along with examples of the generic grammar rules our system learns to represent them. When given a transformation to extract a syntactic paraphrase for, we want to find rules that neither under- nor over-generalize. This means that, while replacing the maximum number of syntactic arguments with

nonterminals, the rules ideally will both retain enough lexicalization to serve as sufficient evidence for the applicability of the transformation and impose constraints on the nonterminals to ensure the arguments' well-formedness.

Possessive rule	$NP \rightarrow$	the NN of the NNP	|	the NNP's NN
	$NP \rightarrow$	the NNS_1 made by NNS_2	|	the NNS_2's NNS_1
Dative shift	$VP \rightarrow$	give NN to NP	|	give NP the NN
	$VP \rightarrow$	provide NP_1 to NP_2	|	give NP_2 NP_1
Adv./adj. phrase move	$S/VP \rightarrow$	$ADVP$ they VBP	|	they VPB $ADVP$
	$S \rightarrow$	it is $ADJP$ VP	|	VP is $ADJP$
Verb particle shift	$VP \rightarrow$	VB NP up	|	VB up NP
Reduced relative clause	$SBAR/S \rightarrow$	although PRP VBP that	|	although PRP VBP
	$ADJP \rightarrow$	very JJ that S	|	JJ S
Partitive constructions	$NP \rightarrow$	CD of the NN	|	CD NN
	$NP \rightarrow$	all $DT\backslash NP$	|	all of the $DT\backslash NP$
Topicalization	$S \rightarrow$	NP, VP .	|	VP, NP .
Passivization	$SBAR \rightarrow$	that NP had VBN	|	which was VBN by NP
Light verbs	$VP \rightarrow$	take action $ADVP$	|	to act $ADVP$
	$VP \rightarrow$	TO take a decision PP	|	TO decide PP

Table 3.2: A selection of meaning-preserving transformations and hand-picked examples of syntactic paraphrases that our system extracts capturing these.

The paraphrases implementing the *possessive rule* and the *dative shift* shown in Table 3.2 are a good examples of this: the two noun-phrase arguments to the expressions are abstracted to nonterminals while each rule's lexicalization provides an appropriate frame of evidence for the transform. This is important for a good representation of dative shift, which is a reordering transformation that fully applies to certain ditransitive verbs while other verbs are uncommon in one of the forms:

give *decontamination equipment* to *Japan*
give *Japan decontamination equipment*

provide *decontamination equipment* to *Japan*
? provide *Japan decontamination equipment*

Note how our method extracts a dative shift rule for *to give* and a rule that both shifts and substitutes a more appropriate verb for *to provide*.

The use of syntactic nonterminals in our paraphrase rules to capture complex transforms also makes it possible to impose constraints on their application. For comparison, as Madnani et al. [2007] do not impose any constraints on how the nonterminal X can be realized, their equivalent of the *topicalization* rule would massively overgeneralize:

$$S \rightarrow \quad X_1,\, X_2\ . \quad | \quad X_2,\, X_1\ .$$

Additional examples of transforms our use of syntax allows us to capture are the *adverbial phrase shift* and the *reduction of a relative clause*, as well as other phenomena listed in Table 3.2.

Our survey shows that we are able to extract appropriately generic representations for a wide range of paraphrastic transformations. This result shows that bilingual parallel corpora can be used to learn sentential paraphrases, and that they are a viable alternative to other data sources like monolingual parallel corpora, which more obviously contain sentential paraphrases, but are scarce.

3.3.2 Limitations of Syntactic SCFGs

Unsurprisingly, syntactic information alone is not sufficient to capture all transformations. For instance it is hard to extract generic paraphrases for all instances of *passivization*, since our syntactic model currently has no means of representing the morphological changes that the verb undergoes:

the reactor *leaks* radiation
radiation *is leaking* from the reactor .

Still, for cases where the verb's morphology does not change, we manage to learn a rule:

the radiation that the reactor had *leaked*
the radiation which *leaked* from the reactor .

Another example of a deficiency in our synchronous grammar models are *light verb* constructs such as:

to take a *walk*
to *walk* .

Here, a noun is transformed into the corresponding verb – something our synchronous syntactic CFGs are not able to capture except through memorization.

3.3.3 Possible Extensions

The above examples illustrate a core deficiency of syntactic paraphrase SCFGs. As the passivization example shows, our use of syntactic labels for nonterminals limits us to matches only occurring between verbs with the exact same form. The light verb

example generalizes this issue: syntactic SCFGs are unable to transfer conceptual information across morphosyntactic boundaries. While the morphosyntactic changes are small for passivization, they become more clear with light verbs, were the concept of "walk" or "talk" may move from a verb form to a noun. Yet more generally, an ideal paraphrase system would be able to generalize constructions like

is a *lover* of music | *loves* music

A possible way to extent the SCFG formalism and labeling would be by providing the means to un-couple the a word's stem, or a conceptual representation thereof, from its concrete syntactic realization. It would be an interesting step towards more semantically powerful paraphrase formalisms to extend the parse tree we use to produce labeling to the sub-word level and introducing a morphological component that could reassemble a newly paraphrased stem and a morphological token into a surface word. With this, the above phrasal paraphrase could be generalized to a paraphrase rule like:

is a CC_1-*NN* of NP_2 | CC_1-*VBZ* NP_2,

wherein CC_1 represents a conceptual paraphrase, for instance *love* | *adore*, and the suffixes -*NN* and -*VBZ* signify a morphological transform into the appropriate surface words.

Another extension of the SCFG formalism that would increase the paraphrases' versatility would be the a relaxation of the synchronousness constraint of the gram-

mar. Many applications, like compression, summarization, or simplification, do not call for strictly paraphrastic transforms. Rather, for these cases, judicious omission of extraneous information may be of more use. This would expand the paraphrase grammar away from strict meaning-equivalence and toward hypo- and hypernymy, increasing its flexibility in real-world applications. One way to accomplish this, would be to expand the formalism used towards quasi-synchronous grammars.

3.4 Impact of Probability Estimation for Paraphrases

Here, we investigate the effect the chosen estimation method has on the resulting paraphrase probability distributions (i.e. the allotment of probability mass onto the paraphrases for one given phrase and syntactic label). Intuitively, the main difference between pivot-probability estimation and the virtual counts method is in how high-frequency words and phrases factor into the probability estimation. We present a two-part analysis. In the first part, we use a human-labeled set of paraphrases to quantitatively compare the predictive power of the two probability estimates. In the second part we perform a qualitative analysis by looking at phrases for which the two methods' estimates diverge strongly.

To perform our analysis, we extract two paraphrase grammars with the same, minimal pruning levels. In both cases we filter out translation rules that occur fewer

Europarl Fr-En v7	Sentences	Words	Word Types
English	1,738,895	40,328,159	102,114
French	1,738,895	44,133,219	123,711

Table 3.3: Corpus statistics for the French-English Europarl v7 bitext used for paraphrase extraction in our experiments.

than twice in the corpus. With the pruning settings being identical, we guarantee that each expression will have the same set of paraphrases in both grammars – the differences will solely lie in how the probability mass is distributed.

The grammars are extracted from the French-English Europarl corpus (v7). The bitext was aligned using the `GIZA++` aligner and the English side was parsed with the Berkeley parser [Petrov et al., 2006]. Table 4.3 gives an overview of corpus size and statistics.

3.4.1 Human-Judged Paraphrase Set

To evaluate how well our features can predict paraphrase quality, we collected a set of human judgments. Ganitkevitch et al. [2013] judged 1962 randomly sampled paraphrases of Propbank [Kingsbury and Palmer, 2002] predicates. The scores are on a scale of 1 to 5, 5 being a perfect paraphrase, and 1 signifying no semantic overlap or substitutability at all. Table 3.4 shows a randomly selected set of paraphrase pairs and their human judgments.

Paraphrase					Score
VBN	→	quoted	|	regarded	2
VBZ	→	involves	|	presupposes	3
VB	→	reject	|	overrule	4
VBN	→	replaced	|	substituted	5
VBZ	→	mentions	|	replies	1
VBD	→	helped	|	favoured	3
VBG	→	launching	|	boosting	2
VBN	→	privatized	|	privatised	5
VBZ	→	demonstrates	|	shows	5
VBZ	→	lets	|	ceases	1

Table 3.4: A selection of Propbank predicate paraphrase pairs and their manually assigned scores.

3.4.2 Correlation With Human Judgments

We calculate the Pearson product-moment correlation coefficient, as well as Spearman's rank coefficient between the human-assigned scores and the probability estimates. As Table 3.5 shows, the pivot-based probability estimate consistently has a slightly better correlation with the human scores than the virtual counts-based estimate. However, the difference in between the two methods is rather small. In fact, the correlation between the two estimates themselves is substantial, for both Pearson's ($r = 0.925$) and Spearman's ($\rho_s = 0.943$) coefficients.

	Pearson's ρ	Spearman's ρ
$-\log p_{pivot}(e_2\|e_1, LHS)$	-0.622	-0.630
$p_{pivot}(e_2\|e_1, LHS)$	0.397	0.630
$-\log p_{count}(e_2\|e_1, LHS)$	-0.592	-0.597
$p_{count}(e_2\|e_1, LHS)$	0.341	0.597

Table 3.5: Pearson's and Spearman's correlation coefficients with human-assigned scores for pivot- and virtual counts-based estimation.

3.4.3 Divergent Phrases

We process the two paraphrase grammars to individually compare paraphrase probability distributions for each phrase. Table 3.6 shows a few phrases e that exhibit the greatest change as measured by reduction in entropy when going from pivot-based estimation to virtual counts.

Looking at the translation probabilities and counts, respectively, that our estimates are based on, we can see the reason for this divergence. Figure 3.9 shows the highest-frequency links in the pivot graph for "stand." Even though the translation links to "rester" and "tre" are equally strong, the latter is an extremely frequent word that near-unambiguously translates as "be." The overwhelming weight of this pair reshapes the final paraphrase distribution for "stand" to be heavily biased towards "be" (Table 3.6). Comparatively, the probability-based pivot graph (Figure 3.10) shows a more even distribution of link strength.

Our analysis suggests that the virtual counts based probability estimate is more

volatile than the pivot-based one. However there is significant room for improvement, considering that we have only considered a naive upper-bound virtual count approach. Possible extensions could include weighing measures like TF/IDF, or more content and domain-conscious methods for computing virtual counts from bitexts.

3.5 Conclusion

In this chapter we introduced a method to learn syntactically informed paraphrases from bilingual parallel texts. We discussed the expressive power and limitations of our formalism, and explored different approaches to estimation of paraphrase probabilities from translation probabilities and counts over the bitext. In the following, we will outline a straightforward adaptation strategy to enable the application of syntactic paraphrases in text-to-text generation.

VBZ brings			
	$p_{counts}(e' \mid e, LHS)$		$p_{pivot}(e' \mid e, LHS)$
is	0.855	is	0.186
has	0.100	brings	0.138
makes	0.005	provides	0.074
provides	0.004	makes	0.062
brings	0.003	has	0.054
does	0.002	leads	0.049
offers	0.002	gives	0.034
gives	0.002	offers	0.028
VB stand			
	$p_{counts}(e' \mid e, LHS)$		$p_{pivot}(e' \mid e, LHS)$
be	0.846	be	0.226
take	0.0233	stand	0.148
support	0.020	remain	0.079
remain	0.017	take	0.049
have	0.014	support	0.038
make	0.009	have	0.024
become	0.005	defend	0.023
NP the board			
	$p_{counts}(e' \mid e, LHS)$		$p_{pivot}(e' \mid e, LHS)$
the council	0.949	the council	0.331
the areas	0.008	the levels	0.069
council	0.004	the board	0.064
the european council	0.003	levels	0.060
the fields	0.003	the areas	0.055
the council's	0.002	the executive board	0.037
the board	0.002	the line	0.033
the committee	0.002	the committee	0.030

Table 3.6: Some of the phrases with the largest shifts in entropy. The count-based estimate, shown on the left, in these case is dominated by high-frequency translation pairs. This results in heavily peaked distributions in the paraphrases.

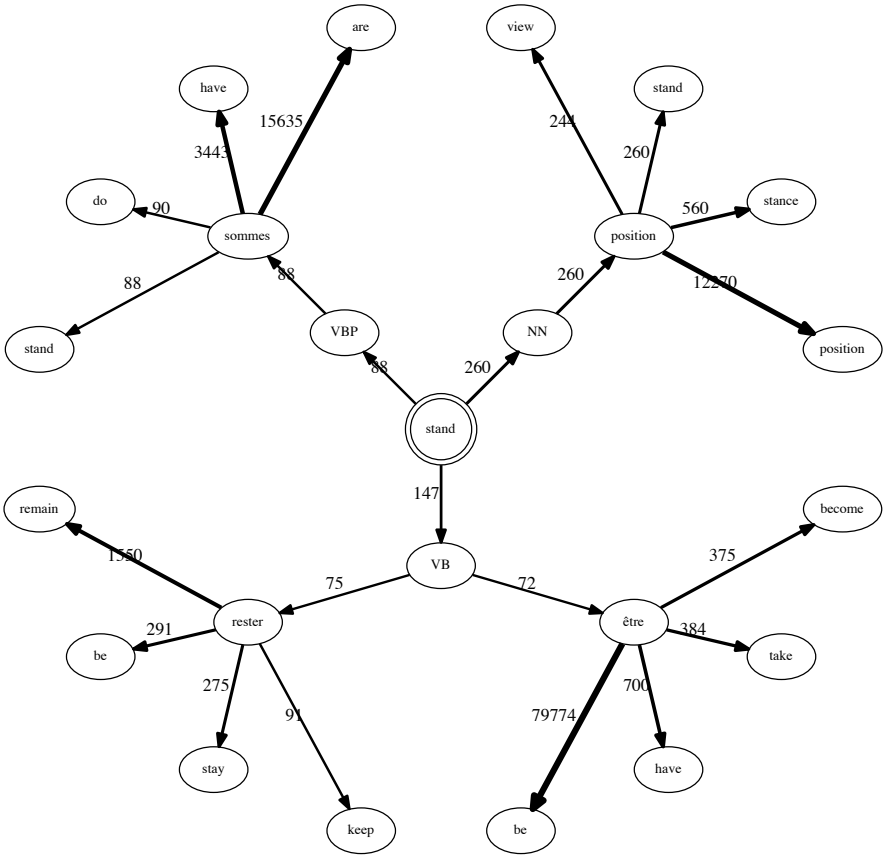

Figure 3.9: The virtual count-based pivot graph for "stand." While several strong links are present, the occurrence of "être" dominates the resulting probability estimate.

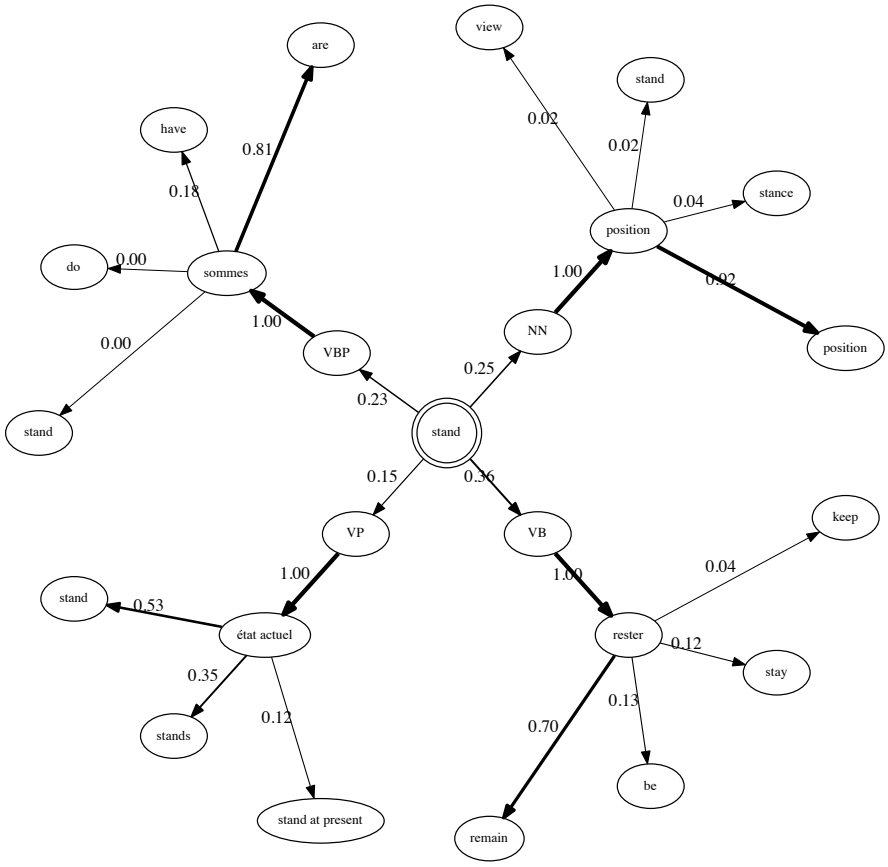

Figure 3.10: The probability-based pivot graph for "stand" is more balanced and results in a less peaked paraphrase distribution for the phrase.

Chapter 4

Text-to-Text Generation with Paraphrases

Many tasks in natural language processing can be characterized as text-to-text generation: an NLP system takes as its input a snippet of text, and returns another. This pattern is especially frequent in directly user-facing NLP systems. Tasks like document summarization, text simplification, sentence compression, and machine translation are all examples of text-to-text generation problems – and highly visible representatives of NLP in the wild. Further instances of text-to-text generation include applications such as poetry generation, query expansion for search engines, transforming declarative sentences into questions, and deriving additional hypotheses for textual entailment recognition.

The generation of output in the above example tasks is typically strongly tied

to the meaning of the input. In fact, the systems' functionality can be seen as a meaning-preserving text transformation. Therefore, in the monolingual case, we can express text-to-text generation as a paraphrasing task. For instance, in abstractive document summarization paraphrases are used to detect and collapse redundancies. In both generative simplification and sentence compression paraphrases can be applied to choose simpler or shorter rewrites of the input, respectively.

While the main concern of this thesis is with the extraction of paraphrases and estimation of their quality, we use text-to-text generation as both a general motivation and, in the evaluation of our systems, an example application.

In this chapter, we introduce sentential text-to-text generation, a class of text-to-text generation tasks, and its relation to paradigms used in machine translation (Section 4.1). To tackle text-to-text generation via paraphrasing, we outline and implement a training paradigm that tailors syntactic, sentential paraphrase models to different sentential text-to-text generation tasks. Our adaptation scheme takes into account a paraphrase's utility for the task at hand, and uses small, appropriately selected data sets to tune the paraphraser (Section 4.2). We posit our framework's suitability for a variety of text-to-text generation tasks, and demonstrate this by obtaining competitive results on the example task of sentence compression (Section 4.3).

4.1 Text-to-Text Generation as Paraphrasing

Many generally stated text-to-text generation tasks can be, with minor relaxations, restated as sentential processing problems. This means that the system no longer processes arbitrarily-sized input, but is limited to looking at one sentence at a time to generate a single output sentence. While this approach may sacrifice access to a wider context, the limited scope helps reduce the computational difficulty of the problem. It also allows for additional speed-up by reducing the task to an embarrassingly parallel setting: input sentences can be trivially processed simultaneously in separate processes or on other machines. This property is important when scaling text-to-text generation systems to perform in real world scenarios.

The broader, document-level context we drop when moving to sentential processing can carry important information. For instance disambiguating signals such as the topics occurring in a document, or entities that are mentioned initially and later only referred to via pronouns. Still, much work in text-to-text generation – most notably in machine translation – has been focussed on developing methods and machinery for the sentence-to-sentence setting.

Limiting a given text-to-text task, for instance text compression, to its sentential equivalent allows us to adopt the approaches developed in machine translation, and interpret the problem as a kind of "monolingual translation." For the compression

case this means that we are now "translating" from English into English, however with the express goal of shortening the input while preserving the meaning. It is evident that the kind of monolingual translation rules used in such a system are paraphrases, and the "monolingual translation" system, in fact, a paraphraser. In this thesis, we set out to develop such a paraphrasing system that is able to tackle text-to-text generation tasks.

An equivalent sentence-level specification is possible for a number of text-to-text generation problems. Beyond text compression, any kind of task that does explicitly require document-level context can be recast as a sentential paraphrasing task. As such, sentential paraphrasing can be applied to text simplification, prose-to-poetry generation, query expansion, reference generation for translation systems, error correction, and other tasks.

Each of these settings can impose specific constraints and objectives on the output of the paraphrasing system. For example in sentence compression, the output may need to be 30% shorter than the input. For simplification the system may be targeting a specific grade level of English, while a domain-generalization task like lawyer-to-English rewriting may strive to paraphrase technical jargon into layman's terms and untangle complex phrases. Finally, in poetry generation the style of poetry chosen will impose constraints on length, meter, and rhyme.

The different kinds of constraints imposed by such tasks differ substantially in type. Where tasks like simplification and domain-generalization are largely focussed

on changes in vocabulary and style, others, like prose-to-poetry generation, impose complex constraints on the output text that involve substantial extensions to the state space and search during decoding [Genzel et al., 2010]. Our focus in this work is on the extraction of high-quality paraphrases and the estimation of their goodness and utility for a given task to successfully capture stylistic changes. For more complex constraints, we derive easy-to-implement heuristics and approximations that will allow us to build paraphrase-based text-to-text generation systems in a matter of days.

4.2 Task-Oriented Adaptation Schemes

We propose a paraphrasing framework that can be adapted to tackle many different text-to-text generation tasks. The main parts of our framework are the paraphrase grammar and decoder as described in Section 3.2, and a set of straightforward extensions that yield a paraphrase-based, task-specific text-to-text generation system capable of producing competitive quality results. The major adaptation steps are:

- A mechanism for extracting synchronous grammar rules (we argue that pivot-based paraphrasing is widely applicable).
- An appropriate set of rule-level features that capture information pertinent to the task (e.g. whether a rule simplifies a phrase).
- An appropriate "objective function" that scores the output of the decoder, i.e.

a task-specific equivalent to the BLEU metric in SMT.

- A development set with examples of the sentential transformations that we are modeling.
- Optionally, a way of injecting task-specific rules that were not extracted automatically.

In the remainder of this section, we illustrate how our bilingually extracted paraphrases can be adapted to perform *sentence compression*, the task of reducing the length of a sentence while preserving its core meaning. Most previous approaches to sentence compression focused only on the deletion of a subset of words from the sentence [Knight and Marcu, 2002]. By virtue of being paraphrase-centric, our approach bears closer resemblance Cohn and Lapata [2008], who expand the task to include substitutions, insertions and reorderings that are automatically learned from parallel texts.

4.2.1 Feature Design

In Section 3.2 we discussed phrasal paraphrase probabilities as a measure of the *goodness* of a paraphrase rule. While these features help quantify how well a paraphrase application preserves meaning in general, they do not make any statement on the task-specific *utility* of the paraphrase rule. Depending on the task, utility can mean things such as the change in language complexity, text length, or style.

For sentence compression, we are interested in paraphrase rules that affect the sentence length. To make this information available to the decoder, we enhance our feature set with compression-targeted features. Specifically, we add:

- The count features c_{src} and c_{tgt}, indicating the number of words on either side of the rule.
- The difference features $c_{dcount} = c_{tgt} - c_{src}$ and the analogously computed difference in the average word length in characters, c_{davg}.
- *CharCountDiff* – a feature that calculates the difference in the number of characters between the phrase and the paraphrase. This feature is used for our sentence compression experiments [Napoles et al., 2011b].
- *CharLogCR* – the log-compression ratio in characters, $\log \frac{chars(e_2)}{chars(e_1)}$, another feature used in sentence compression. The feature is represented in log-space to take advantage of the sign flipping between shortening and lengthening rules. This enables the decoder to learn $\lambda_{CharLogCR}$ as both a reward and penalty.
- WordCountDiff – the difference in the number of words in the original phrase and the paraphrase.
- WordLenDiff – the difference in average word length between the original phrase and the paraphrase.

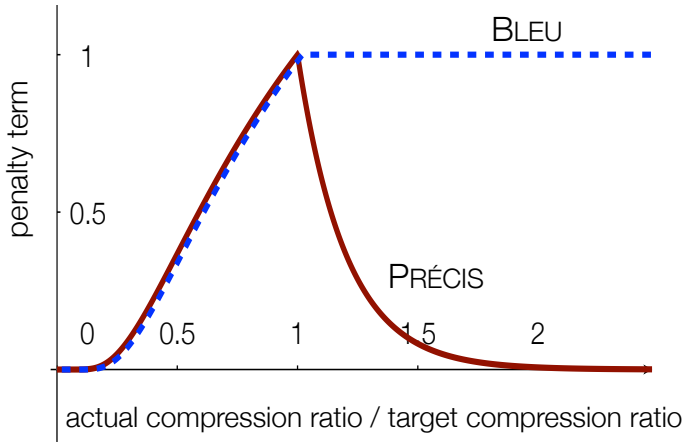

Figure 4.1: An illustration of the difference in output length penalty terms between BLEU and PRÉCIS. The graph plots the penalty factor against the ratio of achieved to targeted compression ratio. While BLEU does not penalize outputs longer than the reference length, PRÉCIS's penalty term generates a sharp drop, effectively creating a tapered window around the target compression ratio.

- WordLogCR – the log-compression ratio in words, estimated as $\log \frac{words(e)}{words(f)}$. This feature is used for our sentence compression experiments.

All the additional features are local to the rule and can therefore be cheaply computed in a single pass over the paraphrase grammar.

4.2.2 Objective Function

Given our paraphrasing system's connection to SMT, the obvious choice for parameter optimization would be to optimize for BLEU over a set of paraphrases, for

instance parallel English reference translations for a machine translation task [Madnani et al., 2007]. For a candidate C and a reference R, (with lengths c and r) BLEU is defined as:

$$\text{BLEU}_N(C,R) = \begin{cases} e^{(1-c/r)} \cdot e^{\sum_{n=1}^{N} \log w_n p_n} & \text{if } c/r \leq 1 \\ e^{\sum_{n=1}^{N} \log w_n p_n} & \text{otherwise} \end{cases},$$

where p_n is the modified n-gram precision of C against R, with typically $N = 4$ and $w_n = \frac{1}{N}$. The "brevity penalty" term $e^{(1-c/r)}$ is added to prevent short candidates from achieving perfect scores.

Naively optimizing for BLEU, however, will result in a trivial paraphrasing system heavily biased towards producing identity "paraphrases" (c.f. Section **??**).[1] This is obviously not the desired output for a compression system. Moreover, BLEU does not provide a mechanism for directly specifying a per-sentence compression rate, which is desirable for the compression task.

Instead, we propose tuning the system to optimize PRÉCIS, an objective function tailored to the text compression task:

$$\text{PRÉCIS}_{\lambda,\varphi}(I,C,R) = \begin{cases} e^{\lambda(\varphi - c/i)} \cdot \text{BLEU}(C,R) & \text{if } c/i \geq \varphi \\ \text{BLEU}(C,R) & \text{otherwise} \end{cases}.$$

For an input sentence I, an output C and reference compression R (with lengths i, c and r), PRÉCIS combines the precision estimate of BLEU with an additional "verbosity penalty" that is applied to compressions that fail to meet a given target compression

[1] Madnani et al. [2007] avoid this issue by disallowing identity paraphrases in their grammar. While this is an appropriate choice for their task, for many applications retaining the capability to self-paraphrase where appropriate is crucial.

rate φ. Simultaneously, we rely on the BLEU brevity penalty to prevent the system from producing overly aggressive compressions. The scaling term λ determines how severely we penalize deviations from φ. In our experiments we use $\lambda = 10$.

It is straightforward to find similar adaptations for other tasks. For text simplification, for instance, the penalty term can include a readability metric. For poetry generation we can analogously penalize outputs that break the meter [Greene et al., 2010].

4.2.3 Development Data

To tune the parameters of our paraphrase system for sentence compression, we need an appropriate corpus of reference compressions. Since our model is designed to compress by paraphrasing rather than deletion, the commonly used deletion-based compression data sets like the Ziff-Davis corpus are not suitable. We have thus created a corpus of compression paraphrases.

Beginning with 9570 tuples of parallel English–English sentences obtained from multiple reference translations for machine translation evaluation, we construct a parallel compression corpus by selecting the longest reference in each tuple as the source sentence and the shortest reference as the target sentence. We further retain only those sentence pairs where the compression rate cr falls in the range $0.5 < cr \leq 0.8$. From these, we randomly select 936 sentences for the development set, as well as 560 sentences for a test set that we use to gauge the performance of our system.

# words	Reference Translations
21	He hoped that the Middle Asian region will maintain stability and development while continuously improving the living standard of its people.
20	He indicated that he hopes Central Asia will enjoy stability and development and their people's lives will be continuously improved.
15	**He wished stability, development and better living standard to the people in Central Asia region.**
19	He hoped that middle Asian region's situation would be stable, economy developed and people's living standard would be improved.
21	He expressed his wishes for the middle Asian region to remain stable and develop further, while improving the region's living standards.
21	He expressed his wishes for the stability and development of the central Asia region with the people's living standard increasing continuously.
22	He wished the Central Asian region stability and development, as well as continuous improvement of the people's living standards here, he added.
17	He expected the stability and development of Middle Asia, and the constant improvement of people's living standards.
20	He expresses hope for the stability and development of Central Asia, and the constant rise of people's living standards here.
25	**He expressed that he hoped the steady development in the Region of Central Asia will continue and the peoples standards of living will be higher.**

Table 4.1: An example of a multi-reference set used. The left column lists sentence length in words. By selecting the longest and shortest of the set, we can obtain a

CR	Shortest Reference	Longest Reference
0.6	He wished stability, development and better living standard to the people in Central Asia region.	He expressed that he hoped the steady development in the Region of Central Asia will continue and the peoples standards of living will be higher.
0.62	Most people guess the Board will favor another cut to stabilize the economy.	It is generally conjectured that the Federal Reserve will decide to drop interest rates again in order to maintain economic stability.
0.74	It was stated that the two were forced to serve as slaves in North Korea before fleeing.	The statement said that the two men were forced to work as slaves in a coal mine in North Korea before they escaped.
0.65	Those countries were effected by the low prices of export products.	These countries were for a large part affected by the low prices of their major export products.
0.73	The severely wounded man was later rescued by an armored carrier.	Later, some soldiers arrived in an armored personnel carrier and rescued the seriously wounded man.
0.46	Government leaders, specialists, and users alike have applauded Wangma Computer Company's selfless decision.	With regard to Wangma Computer Company's decision of an unselfish contribution to the society, the leaders, experts and users in the conference had all given their high appreciations.

4.2.4 Grammar Augmentations

As we discussed in Section 3.3, the paraphrase grammar we induce is capable of representing a wide variety of transformations. However, the formalism and extraction method are not explicitly geared towards a compression application. For instance, the synchronous nature of our grammar does not allow us to perform deletions of constituents as done by Cohn and Lapata [2007]'s tree transducers. One way to extend the grammar's capabilities towards the requirements of a given task is by injecting additional rules designed to capture appropriate operations.

For the compression task, this could include adding rules to delete target-side nonterminals:

$$JJ \rightarrow JJ \mid \varepsilon$$

However, this would render the grammar asynchronous and require adjustments to the decoding process. Alternatively, we can generate rules that specifically delete particular adjectives from the corpus:

$$JJ \rightarrow \text{superfluous} \mid \varepsilon \ .$$

In our experiments we evaluate the latter approach by generating optional deletion rules for all adjectives, adverbs and determiners. Since these new rules are not backed by data, we cannot estimate paraphrase probabilities for them. Instead, we annotate them with two deletion indicator features. We add a feature for the syntactic label of the deleted word, del_C, corresponding to allowing the system to learn how helpful is

it to delete (for example) an adverb. Another, sparser, deletion feature del_e is added for the word itself, permitting our paraphraser to learn that particular words, like "very" or "new," may often carry little content signal and can be safely deleted. We also retain the parsing-based probability features $p(C|e)$ and $p(e|C)$. These provide a means to tie the application of the deletion rule to the likelihood of e indeed falling into the deletable category C.

The resulting newly generated rules have the form:

$$C \rightarrow e \mid \varepsilon \ ,$$

with $C \in \{RB, DET, JJ\}$ and e being an exhaustive list of words we encountered with the label C.

Furthermore, in our system, we implemented the injection of rules that improve the handling of out-of-vocabulary words. Typically an SMT system will generate heavily penalized identity translation rules for all words in an input sentence, to assure that the OOV rules will only apply when there is no translation rule for a particular word. For our paraphrasing approach however, encountering an unknown word is not critical as long as we can assign it the proper syntactic label. To enable our system to process unseen words in the input, we allow for syntactically parsed input and for every input word w include rules of the form

$$T \rightarrow w \mid w,$$

where T is the part-of-speech tag assigned to w in the input parse. To enable a proper

inclusion into the paraphrasing model, we add a constant penalty feature φ_{oov} and include its weight in our parameter estimation process.

4.3 Paraphrase-Based Sentence Compression

To evaluate the quality of the paraphrases our method extracts, as well as effectiveness of our adaptation scheme, we conduct a series of experiments in paraphrase-based sentence compression. In our experiments, we are interested in experimentally answering the following questions:

- Does the inclusion of syntactic nonterminals and constraints in paraphrase extraction help text-to-text performance?
- Does augmented the paraphrase grammar with deletion rules help the system produce better compressions?
- How does our paraphrase-based text-to-text system compare to deletion-based approaches to text compression?

In the following, we detail the corpora, tools, and parametrizations used as a basis for constructing our fully adapted *topline* paraphrasing system (Section 4.3.1). In Section 4.3.2 we then describe our evaluation methodology, followed by a series of contrastive experiments conducted to provide insights into the above questions.

Europarl Fr-En v5	Sentences	Words	Types
English	1,723,705	47,915,991	102,114
French	1,723,705	51,708,806	123,711

Table 4.3: Corpus statistics for the French-English Europarl v5 bitext used for paraphrase extraction in our experiments.

4.3.1 Paraphrase-Based Compression Setup

The basis of our sentence-to-sentence paraphrasing system is formed by a syntactically informed pivot-extracted paraphrase grammar, and an n-gram language model. The data sets and tools used to extract and estimate the latter two are detailed in the following. On top of the paraphrase grammar we extract, we apply the full set of compression-specific adaptations described in Section 4.2.

4.3.1.1 Paraphrase Grammar

The paraphrase grammars used in this chapter's experiments are extracted from the French-English Europarl corpus (v5). The bitext was aligned using the Berkeley aligner and the English side was parsed with the Berkeley parser [Petrov et al., 2006]. We obtained the initial translation grammar using the SAMT toolkit Venugopal and Zollmann [2009]. On top of CCG-style slashed constituents, the SAMT toolkit produces concatenated nonterminals (e.g. *NP+VP*). In Table 4.4 we list the number of paraphrase rules, breaking out the number of identity paraphrases, as well as the

Grammar	# Rules
total	42,353,318
w/o identity	23,641,016
w/o complex constituents	6,439,923
w/o complex const. & identity	5,097,250

Table 4.4: Number and distribution of rules in our paraphrase grammar. Complex constituents include CCG-style nonterminals as well as concatenated labels like *NP+VP*.

complex constituents.

The grammars we extract tend to be extremely large. To keep their size manageable, we only consider translation rules that have been seen more than 3 times and whose translation probability exceeds 10^{-4} for pivot recombination. Additionally, we only retain the top 25 most likely paraphrases of each phrase, ranked by a uniformly weighted combination of phrasal and lexical paraphrase probabilities.

4.3.1.2 Language Model

The language model used in our paraphraser, as well in the Clarke and Lapata [2008] baseline system is a Kneser-Ney discounted 5-gram model estimated on the Gigaword corpus using the SRILM toolkit Stolcke [2002]. The corpus was cleared of duplicate sentences, punctuation-normalized, tokenized, and lowercased (using the tools provided with the Joshua Decoder Toolkit).

4.3.1.3 System Tuning

For our experiments, we tuned the model parameters to maximize the PRÉCIS score of the output. The optimizer used is the Z-MERT toolkit Zaidan [2009]. For decoding, we used the Joshua Decoder 2.0 Li et al. [2010]. MERT was run for a maximum of 10 iterations. We generated the 300 unique top-scoring paraphrases per input sentence. The PRÉCIS metric was computed over 1-to-4-grams. We varied the target compression ratio between 0.5 and 0.9 for different optimization runs.

4.3.2 Human Evaluation Setup

We solicit human judgments of the candidate compressions along two five-point scales: grammaticality and meaning. Raters are instructed to decide how much the meaning from a reference translation is retained in the compressed sentence, with a score of 5 indicating that all of the important information is present, and 1 being that the compression does not retain any of the original meaning. Similarly, a grammar score of 5 indicates perfect grammaticality, and a grammar score of 1 is assigned to sentences that are entirely ungrammatical. In this, we adopt a translation rating guideline set by the Linguistic Data Consortium [2002], modified to fit the sentence compression task.

To ensure fairness, we perform pairwise system comparisons with compression rates strictly tied on the sentence-level. For any comparison, a sentence is only in-

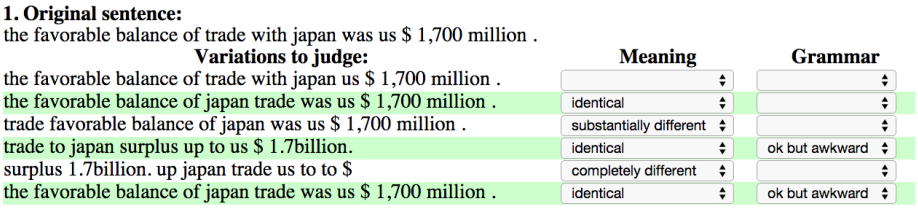

Figure 4.2: An example of the human intelligence task (HIT) the Turkers are presented with.

cluded in the computation of average scores if the difference between both systems' compression rates is < 0.05. Because evaluation quality correlates linearly with compression rate, the community-accepted practice of not comparing based on a closely tied compression rate is potentially subject to erroneous interpretation [Napoles et al., 2011a].

The human judgements are solicited on Amazon Mechanical Turk. We set the evaluation to be three-fold redundant, i.e. every tuple of compressions is judged by three different Turkers independently. For quality control, and to obtain upper-bound scores for actual human-produced compressions, we include the human reference compression with the candidates presented for judgement. Additionally, we use negative controls generated by randomly deleting words until the desired compression rate is achieved.

	CR	Meaning	Grammar
Hiero	0.56	2.57	2.35
Syntax	0.56	2.76	**2.67**
Syntax	0.53	2.70	2.49
Syntax+Feat.	0.53	2.71	**2.54**
Syntax+Feat.	0.54	2.79	**2.71**
Syntax+Aug.	0.54	**2.96**	2.52
Human Compressions	0.73	4.26	4.35

Table 4.5: Human evaluation for shorter compressions and for variations of our paraphrase system. Syntax+Feat. includes the compression features from Section 4.2.1, Syntax+Aug. includes optional deletion rules from Section 4.2.4.

4.3.3 Effects of Syntax and Task Adaptations

We evaluate the usefulness of syntactic information in paraphrases, as well as that of our task-oriented adaptation steps (c.f. Section 4.2). To do this we perform a series of contrastive experiments. Specifically, we compare paraphrase-based compression systems starting with a Hiero-based paraphraser (Hiero), and successively adding syntactic information into the paraphrases (Syntax), then task-specific features (Syntax+Feat.), and finally compression-specific grammar augmentations (Syntax+Aug.).

Intuitively, we expect the different adaptations to have varying effects. For instance, we cannot expect a paraphrasing system like Hiero or Syntax, that is not aware of the impact a rule has on the input length, to successfully learn to compress text. The Syntax-Feat. system, on the other hand, has access to informative features

that reflect the task. It is therefore likely to produce good-quality compressions. While we expect Syntax-Feat. to perform well especially at moderate compression rates, it is likely to run into difficulties as we lower the target compression ratio to levels that are hard to achieve without using deletion operations. Here, we expect Syntax-Aug. to do better, at a cost to grammaticality due to the rather coarse nature of the adaptations.

Table 4.5 shows a suite of pairwise system comparisons for compression rates ≈ 0.5. Each comparison is performed over a set of inputs for which both systems produced outputs with highly comparable ($\delta < 0.05$) compression rates.

We see that going from a Hiero-based to a syntactic paraphrase grammar yields a significant improvement in grammaticality. This intuitively makes sense, as we expect syntactic information to capture more well-formed rewrites. Adding our task-specific features improves grammaticality even further. Further augmenting the grammar with deletion rules (c.f. Section 4.2.4) significantly helps retain the core meaning at compression rates this high, however compared to the un-augmented syntactic system grammaticality scores drop.

4.3.4 Compression Baselines

To assess the output quality of the resulting paraphrase-based sentence compression system, we compare it to two contemporary state-of-the-art sentence compression systems. Specifically, we contrast with our implementation of Clarke and Lapata

[2008]'s compression model which uses a series of constraints in an integer linear programming (ILP) solver, and the T3 tree transducer toolkit [Cohn and Lapata, 2007], which learns a synchronous tree substitution grammar (STSG) from paired monolingual sentences. Unlike SCFGs, the STSG formalism allows changes to the tree topology. Cohn and Lapata [2007] argue that this is a natural fit for sentence compression, since deletions introduce structural mismatches. We trained the T3 software[2] on the 936 $\langle \text{full}, \text{compressed} \rangle$ sentence pairs that comprise our development set. This is equivalent in size to the training corpora that Cohn and Lapata [2007] used (their training corpora ranged from 882–1020 sentence pairs), and has the advantage of being in-domain with respect to our test set. Both these systems reported results outperforming previous systems such as McDonald [2006].

Table 4.6 pits our fully adapted Syntax-Aug. system against the ILP and T3 systems at a high compression rate of ≈ 0.5. We can see that the paraphrase-based system is able to achieve meaning retention results on par with the ILP system. However, ILP significantly outperforms the paraphraser in grammaticality. In part, we can ascribe this result to the difference in difficulty between performing syntactically informed deletions and producing a full rewrite of a sentence. At the same time, the paraphrase-based approach manages to significantly outperform the T3 system, and a trivial baseline that performs random deletions.

In Table 4.7 we compare our system to the ILP approach at a more modest

[2] `www.dcs.shef.ac.uk/people/T.Cohn/t3/`

	CR	Meaning	Grammar
Syntax+Aug.	0.52	2.87	2.40
ILP	0.52	2.83	**3.09**
Syntax+Aug.	0.50	2.41	**2.20**
T3	0.50	2.01	1.93
Random Deletions	0.50	1.94	1.57

Table 4.6: Comparison of human evaluation results against other compression approaches for short compressions.

compression rate of ≈ 0.8. Here, we significantly outperform ILP in meaning retention while achieving comparable results in grammaticality. This improvement is significant at $p < 0.0001$, using the sign test, while the better grammaticality score of the ILP system is not statistically significant ($p < 0.088$). These results indicate that, over a variety of compression rates, our framework for text-to-text generation is performing as well as or better than specifically tailored methods.

Overall, our paraphrase-based approach to compression compares favorably to the T3 system [Cohn and Lapata, 2007], though further experiments at different compression ratios may be warranted. Our system is further performing roughly on par with the ILP-based compression, a contemporary, specialized state-of-the-art system that makes use of syntactic annotations and deletion heuristics. The paraphrase-based approach trades off performs as one would intuit – trading grammaticality of rewrites for better retention of meaning.

	CR	Meaning	Grammar
Reference	0.73	4.26	4.35
Syntax+Feat.	0.80	**3.67**	3.38
ILP	0.80	3.50	3.49

Table 4.7: Results of the human evaluation on longer compressions: pairwise compression rates (CR), meaning and grammaticality scores. Bold indicates a statistically significance difference at $p < 0.05$.

4.3.5 Qualitative Analysis

Table 4.8 shows an example sentence drawn from our test set and the compressions produced by the different systems. While we see that both the paraphrase and ILP systems produce good quality results, with the paraphrase system retaining the meaning of the source sentence more accurately.

4.4 Conclusions

Our experiments show that sentential paraphrasing presents a valid and, above all, flexible approach to tackle text-to-text generation problems. We have:

- Shown that injecting syntactic information into paraphrases yields better quality text-to-text generation systems.
- Presented an analysis of pruning techniques to effectively reduce paraphrase grammar size while retaining good quality paraphrases.

Source	he also expected that he would have a role in the future at the level of the islamic movement across the palestinian territories , even if he was not lucky enough to win in the elections .
Reference	he expects to have a future role in the islamic movement in the palestinian territories if he is not successful in the elections .
Syntax+Feat.	he also expected that he would have a role in the future of the islamic movement in the palestinian territories , although he was not lucky enough to win elections .
ILP	he also expected that he would have a role at the level of the islamic movement , even if he was not lucky enough to win in the elections .
Source	in this war which has carried on for the last 12 days , around 700 palestinians , which include a large number of women and children , have died .
Reference	about 700 palestinians , mostly women and children , have been killed in the israeli offensive over the last 12 days .
Syntax+Feat.	in this war has done for the last 12 days , around 700 palestinians , including women and children , died .
ILP	in this war which has carried for the days palestinians , which include a number of women and children died .
Source	hala speaks arabic most of the time with her son , taking into consideration that he can speak english with others .
Reference	hala speaks to her son mostly in arabic , as he can speak english to others .
Syntax+Feat.	hala speaks arabic most of the time with her son , considering that he can speak english with others .
ILP	hala speaks arabic most of the time , taking into consideration that he can speak english with others .

Table 4.8: Example compressions produced by the two systems in Table 4.7 for three input sentences from our test data.

- Empirically compared two different means of paraphrase probability estimation.
- Outlined and analyzed a four-part task adaptation scheme that allows us to easily convert a general-purpose paraphraser into a specialized system.
- Compared our thusly adapted sentence compression system to specialized models and found that it performs on par with the state of the art.

With this, we have introduced a complete suite of syntactic paraphrase-based tools for text-to-text generation. Our set of methods allows for easy adaptation to a text to text task, and our experiments have given insight into the means to tailor the system to other computational requirements (such as smaller, or less richly labeled grammars).

Chapter 5

Improving Paraphrase Quality with Distributional Similarity

In the previous chapters we have introduced a pivot-based extraction and estimation method for syntactically informed paraphrases. We have then shown that by adapting an off-the-shelf machine translation decoder, we can use our syntactic paraphrases to deliver state of the art text-to-text generation performance.

In this chapter, we set out to improve the quality of our extracted paraphrases. In order to do so, we will extend our pivot-based paraphrase collection by incorporating a new source of information. The signal we are adding to our model is based on the notion of *distributional similarity* between two phrases: two phrases are judged semantically similar when they occur in similar contexts over a large corpus of text. Using distributional similarity significantly increases the amount of data we can use

to improve our paraphrases. While the pivot-based approach is limited to parallel (or made-parallel) corpora, distributional similarity can be computed from any text corpus, specifically including monolingual ones.

Furthermore, we expect the addition of distributional similarity features to contribute a signal that is largely orthogonal to the information leveraged in pivot-based paraphrase estimation. This is due to us being able to use significantly larger monolingual data sets and purely contextual cues for the new features.

Paraphrase collections for text-to-text generation have been extracted from a variety of types of corpora. Several approaches rely on bilingual parallel data [Bannard and Callison-Burch, 2005, Zhao et al., 2008a, Callison-Burch, 2008, Ganitkevitch et al., 2011], while others leverage distributional methods on monolingual text corpora [Lin and Pantel, 2001, Bhagat and Ravichandran, 2008]. However, only preliminary studies have been undertaken to combine the information from these two sources [Chan et al., 2011].

In this chapter, we:

- Introduce the notion of distributional similarity (Section 5.1).
- Describe the data sets used for this work, and the contextual features we define and collect over them (Section 5.2). We incorporate a variety of different types of features in our distributional similarity model, and show that they can be used to achieve significant improvements in grammaticality in text-to-text generation.

- Elaborate on locality-sensitive hashing (LSH), a fast randomized approach to dimensionality reduction that allows us to compactly represent the large feature space used in our work. LSH is the context and feature encoding method used in our experiments (Section 5.2.3).

- Discuss the incorporation of distributional similarity features into our existing paraphrase grammar via paraphrase re-ranking (Section 5.3.1), and define a decompositional approach to scoring paraphrase patterns that contain constituent-level gaps, e.g.

  $$sim(\text{one } JJ \text{ instance of } NP, \text{a } JJ \text{ case of } NP).$$

  This generalizes over distributional similarity for contiguous phrases, enabling us to fully annotate our syntactic paraphrase grammar with monolingual information (Section 5.3.2).

- Compare a distributional similarity-enhanced paraphrase system to several strong baselines on the text-to-text generation task of sentence compression. Our method shows results competitive with a contemporary custom-build compression system, significantly improving over our previous, purely bilingually sourced paraphraser.

5.1 Monolingual Distributional Similarity

Distributional similarity is defined between two textual expressions (or phrases), e and e'. The similarity of the two phrases is computed based on comparing their contextual features. To describe the phrase e, we define a set of features that capture the context of an occurrence of e in our corpus. Writing the context vector for the i-th occurrence of e as $\vec{s}_{e,i}$, we can aggregate over all occurrences of e, resulting in a *distributional signature* for e: $\vec{s}_e = \sum_i \vec{s}_{e,i}$. Following the intuition that phrases with similar meanings occur in similar contexts, we can then quantify the *similarity* or goodness of e' as a paraphrase of e by computing the cosine similarity between their distributional signatures:

$$sim(e, e') = \frac{\vec{s}_e \cdot \vec{s}_{e'}}{|\vec{s}_e||\vec{s}_{e'}|}.$$

A wide variety of features can be used to describe the distributional context of a phrase. Rich, linguistically informed feature sets that rely on dependency and constituency parses, part-of-speech tags, or lemmatization have been proposed in widely known work such as by Church and Hanks [1991] and Lin and Pantel [2001]. In these settings, a phrase could be described by the various syntactic relations it has with lexical items in its context, such as: "for what verbs do we see with the phrase as the subject?", or "what adjectives modify the phrase?".

However, when moving to vast text collections or collapsed representations of large text corpora, linguistic annotations may become impractically expensive to produce.

A straightforward and widely used solution is to fall back onto lexical n-gram features, e.g. "what words or bigrams have we seen to the left of this phrase?" A substantial body of work has focussed on using this type of feature-set for a variety of purposes in NLP [Lapata and Keller, 2005, Bhagat and Ravichandran, 2008, Lin and Dyer, 2010, Van Durme and Lall, 2010].

In practice, distributional similarity approaches are implemented by first extracting signatures for a set of expressions or phrases. To then extract paraphrases for a given e one will then search the signature space for any close neighbors e'. Similarly, given two phrases e and e', one will compare their signatures to compute their similarity. Our work follows this two-stage approach.

5.2 Distributional Similarity Model

We contrast two approaches to constructing our distributional signatures: a simple feature set collected over a very large aggregated corpus, and a rich feature set taking into account a variety of automated annotations that we collect over a smaller (but still sizable) text corpus.

5.2.1 n-gram Model

The high-coverage model (from here on: n-gram model) is drawn from a web-scale n-gram corpus [Brants and Franz, 2006, Lin and Dyer, 2010]. Since the Google

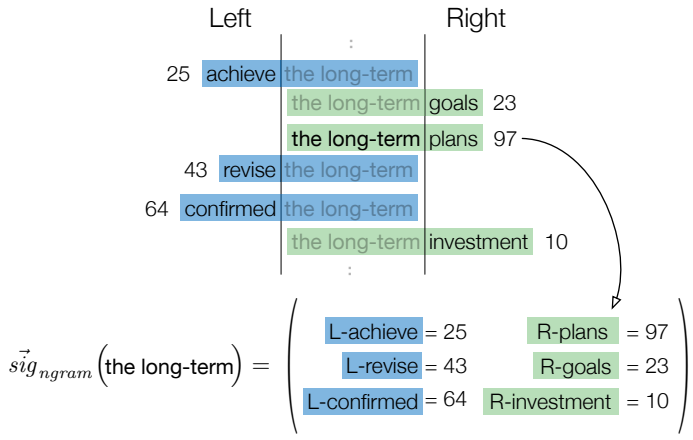

Figure 5.1: An example of the n-gram feature extraction on an n-gram corpus. Here, "the long-term" is seen preceded by "revise" (43 times) and followed by "plans" (97 times). The corresponding left- and right-side features are added to the phrase signature with the counts of the n-grams that gave rise to them.

n-grams consist of 1-to-5-grams with occurrence counts, we are limited in both the type of phrases for which we can collect signatures, and in our choice of feature set.

We thus extract signatures for contiguous phrases up to a length of 4. For each phrase p we look at n-grams of the form wp and pv, where w and v are single words. We then extract the corresponding features w_{left} and v_{right}. The feature count is set to the count of the n-gram, reflecting the frequency with which p was preceded or followed, respectively, by w and v in the data the n-gram corpus is based on. Figure 6.5 illustrates this feature extraction approach. The resulting collection comprises distributional signatures for the 200 million most frequent 1-to-4-grams in the n-gram corpus.

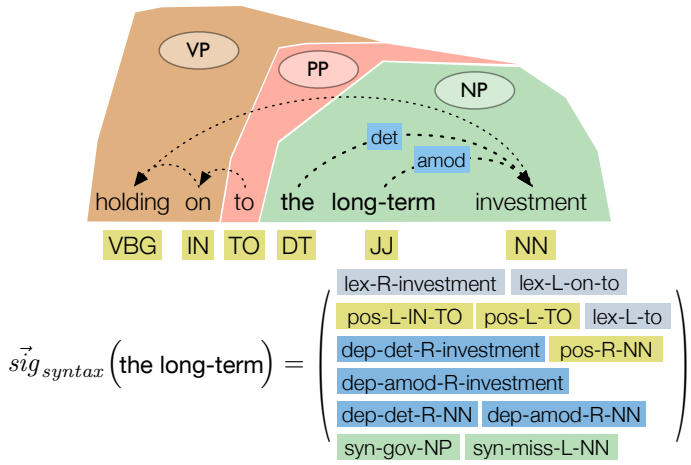

Figure 5.2: An example of the syntactic feature-set. The phrase "the long-term" is annotated with position-aware lexical and part-of-speech n-gram features (e.g. "on to" on the left, and "investment" and "NN" to its right), labeled dependency links (e.g. $amod - investment$) and features derived from the phrase's CCG label NP/NN.

5.2.2 Syntactic Model

For the syntactically informed signature model (from here on: syntax model), we make use the constituency and dependency parses provided in the Annotated Gigaword corpus [Napoles et al., 2012], as well as token-level annotations such as parts of speech and named entity tags. To remain consistent with the n-gram model, we limit ourselves to contiguous phrases up to a length of 4. The following feature set is used to compute distributional signatures for the extracted phrases:

- n-gram based features for words seen to the left and right of a phrase.
- Position-aware lexical, lemma-based, part-of-speech, and named entity class

unigram and bigram features, drawn from a three-word window to the right and left of the phrase.

- Incoming and outgoing (wrt. the phrase) dependency link features, labeled with the corresponding lexical item, lemmata and POS.
- Syntactic features for any constituents governing the phrase, as well as for CCG-style slashed constituent labels for the phrase.
- Position-aware lexical and part-of-speech unigram and bigram features, drawn from a three-word window to the right and left of the phrase.
- Features based on dependencies for both links into and out of the phrase, labeled with the corresponding lexical item and POS. If the phrase corresponds to a complete subtree in the constituency parse we additionally include lexical and POS features for its head word.
- Syntactic features for any constituents governing the phrase, as well as for CCG-style slashed constituent labels for the phrase. The latter are split in governing constituent and missing constituent (with directionality).

Figure 6.6 illustrates the syntax model's feature extraction for an example phrase occurrence. Using this method we extract distributional signatures for over 175 million 1-to-4-gram phrases.

5.2.3 Locality Sensitive Hashing

Collecting distributional signatures for a large number of phrases quickly leads to unmanageably large datasets. Storing the syntax model's 175 million signatures in a compressed readable format, for instance, would require over 290GB of disk space. This is both due the number of phrases, but also to the high dimensionality of our highly lexicalized feature space. To make the signature extraction computationally viable, we therefore need to apply an effective dimensionality reduction technique.

A wide variety of viable approaches to dimensionality reduction exists. For our setup, however, we favor a fast, trivially parallelizable method that also allows us to bypass explicitly computing the feature vectors, which can be memory intensive for frequent phrases. We therefore choose an online variant of locality sensitive hashing (LSH), as described by Van Durme and Lall [2010]. LSH has been previously used to enable the extraction of large collections of paraphrases from vast monolingual corpora [Ravichandran et al., 2005, Bhagat and Ravichandran, 2008]. In our work, we rely on the LSH implementation that is part of the Jerboa toolkit [Van Durme, 2012].

The online LSH variant is based on earlier work of Indyk and Motwani [1998] and Charikar [2002]. It approximates the cosine similarity between two feature vectors based on the Hamming distance between their projections in a dimensionality-reduced, boolean-valued space. To compute this bit-vector representation from a d-dimensional, real-valued input vector $\vec{v} \in \mathbb{R}^d$, we first project it through a random

matrix $P \in \mathbb{R}^{b \times d}$, which is populated with draws from $\mathcal{N}(0, 1)$:

$$p(\vec{v}) = P\vec{v}. \tag{5.1}$$

We then convert the resulting b-dimensional vector $p(\vec{v}) \in \mathbb{R}^b$ into a bit-vector $r(\vec{v})$ by setting each bit conditioned on whether the corresponding projected value is less than 0. Given two bit signatures $r(\vec{u})$ and $r(\vec{v})$, we can approximate the cosine similarity of $\vec{u}$ and $\vec{v}$ as:

$$sim'(u, v) = \cos\Big(\frac{D(r(\vec{u}), r(\vec{v}))}{b}\pi\Big),$$

where $D(\cdot, \cdot)$ is the Hamming distance. Unless otherwise noted, we use 256-bit signatures in our experiments. This reduces the memory requirements for the syntax model to around 8.8GB.

The feature space dimensionality d varies with the input data and the feature templates we choose to use. Since d tends to be very large, and in fact cannot be known until the features have been extracted over the entirety of the input data, it is impractical to hold a fully instantiated projection matrix P in memory. The alternative is, given an input feature (i.e. a v_i), to deterministically draw b samples from $\mathcal{N}(0, 1)$. Computationally, this is a costly and (for frequently occurring features) highly redundant operation. A key advantage of the LSH variant of Van Durme and Lall [2010] is that it circumvents this issue. Instead of fully instantiating or on-demand generating P, they precompute a pool of $S = \{s_1 \dots s_K \mid s_k \ \mathcal{N}(0, 1)\}$ and sample the p_{ij} by hashing into S.

5.3 Incorporating Distributional Information

5.3.1 Paraphrase Re-Ranking

Chan et al. [2011] presented an initial investigation into combining phrasal paraphrases obtained through bilingual pivoting with monolingual distributional information. Their work investigated a reranking approach and evaluated their method via a substitution task, showing that the two sources of information are complementary and can yield improvements in paraphrase quality when combined.

We aim to extend Chan et al. [2011]'s work by fully integrating the similarity scores into our SCFG paraphraser, and evaluating on text-to-text generation.

5.3.2 Decompositional Scoring of Complex Paraphrases

In order to incorporate distributional similarity information into the paraphrasing system, we need to calculate similarity scores for the paraphrastic SCFG rules in our grammar. For rules with purely lexical right-hand sides e_1 and e_2 this is a simple task, and the similarity score $sim(e_1, e_2)$ can be directly included in the rule's feature vector $\vec{\varphi}$. However, if e_1 and e_2 are long, their occurrences become sparse and their

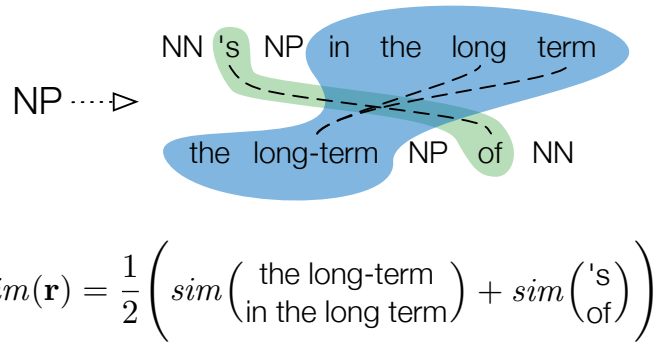

Figure 5.3: Scoring a rule by extracting and scoring contiguous phrases consistent with the alignment. The overall score of the rule is determined by averaging across all pairs of contiguous subphrases.

similarity can no longer be reliably estimated. In our case, the right-hand sides of our rules often contain gaps and computing a similarity score is less straightforward.

Figure 5.3 shows an example of such a discontinuous rule and illustrates our solution: we decompose the discontinuous patterns that make up the right-hand sides of a rule $\mathbf{r}$ into pairs of contiguous phrases $\mathcal{P}(\mathbf{r}) = \{\langle e, e' \rangle\}$, for which we can look up distributional signatures and compute similarity scores. This decomposition into phrases is non-trivial, since our sentential paraphrase rules often involve significant reordering or structural changes. To avoid comparing unrelated phrase pairs, we require $\mathcal{P}(\mathbf{r})$ to be consistent with a token alignment $\mathbf{a}$. The alignment is defined analogously to word alignments in machine translation, and computed by treating the source and target sides of our paraphrase rules as a parallel corpus.

We define the overall similarity score of the rule to be the average of the similarity

scores of all extracted phrase pairs:

$$sim(\mathbf{r}, \mathbf{a}) = \frac{1}{|\mathcal{P}(\mathbf{a})|} \sum_{(e,e') \in \mathcal{P}(\mathbf{a})} sim(e, e').$$

Since the distributional signatures for long, rare phrases may be computed from only a handful of occurrences, we additionally query for the shorter sub-phrases that are more likely to have been observed often enough to have reliable signatures and thus similarity estimates.

Our definition of the similarity of two discontinuous phrases substantially differs from others in the literature. This difference is due to a difference in motivation. Lin and Pantel [2001], for instance, seek to find new paraphrase pairs by comparing their arguments. In this work, however, we try to add orthogonal information to existing paraphrase pairs. Both our definition of pattern similarity and our feature-set (see Section 5.2) are therefore geared towards comparing the substitutability and context similarity of a pair of paraphrases.

Our two similarity scores are incorporated into the paraphraser as additional rule features in $\vec{\varphi}$, sim_{ngram} and sim_{syn}, respectively. We estimate the corresponding weights along with the other λ_i as detailed in Section 5.4.

5.4 Experiments in Text-to-Text-Generation

5.4.1 Task: Sentence Compression

To evaluate the impact of distributional information on paraphrase quality, we again use the sentence compression task. While the datasets used to create our paraphrase grammars are identical to Chapter 4, the extraction stack has been updated to use the Thrax grammar extraction toolkit [Ganitkevitch et al., 2012a]. Due to the resulting changes in heuristics applied during extraction and compression-centric features (Section 5.4.2.1) added, the resulting paraphrase systems are not identical to the ones used for our experiments in Chapter 4.

5.4.2 Experimental Setup

5.4.2.1 Base Paraphrase Grammar and Language Model

As in Chapter 4, we extract our paraphrase grammar from the French–English portion of the Europarl corpus (version 5) [Koehn, 2005]. The Berkeley aligner [Liang et al., 2006] and the Berkeley parser [Petrov et al., 2006] are used to align the bitext and parse the English side, respectively. The paraphrase grammar is produced using the Hadoop-based Thrax grammar extractor's paraphrase mode [Ganitkevitch et al., 2012a]. The syntactic nonterminal labels we allowed in the grammar were limited to constituent labels and CCG-style slashed categories. Paraphrase grammars extracted

via pivoting tend to grow very large. To keep the grammar size manageable, we pruned away all paraphrase rules whose phrasal paraphrase probabilities $p(e_1|e_2)$ or $p(e_2|e_1)$ were smaller than 0.001.

We extend the feature-set used in Chapter 4 with a number of features that aim to better describe a rule's compressive power. Here are the compression-specific features we use in our experimental setup:

- Raw word count features $wcount_{src}$ and $wcount_{tgt}$ for either side of a paraphrase rule.
- The net word count difference effected by applying the rule: $wcount_{diff}$.
- Character-count equivalents for the above features: $ccount_{src}$, $ccount_{tgt}$, and $ccount_{diff}$.
- Features expressing the. log-compression ratio of each rule $word_{cr} = \log \frac{wcount_{tgt}}{wcount_{src}}$ and the analogously defined $char_{cr} = \log \frac{ccount_{tgt}}{ccount_{src}}$.

The language model used in our paraphraser and the Clarke and Lapata [2008] baseline system is a Kneser-Ney discounted 5-gram model estimated on the Gigaword corpus using the SRILM toolkit [Stolcke, 2002].

5.4.2.2 Model Tuning

For model tuning and decoding we used the Joshua machine translation system [Ganitkevitch et al., 2012a]. The model weights are estimated using an implemen-

tation of the PRO tuning algorithm [Hopkins and May, 2011], with PRÉCIS as our objective function.

5.4.2.3 Human Evaluation Setup

To rate the quality of our output, we solicit human judgments of the compressions along two five-point scales: grammaticality and meaning preservation. Judges are instructed to decide how much the meaning from a reference translation is retained in the compressed sentence, with a score of 5 indicating that all of the important information is present, and 1 being that the compression does not retain any of the original meaning. Similarly, a grammar score of 5 indicates perfect grammaticality, while a score of 1 is assigned to sentences that are entirely ungrammatical. We ran our evaluation on Mechanical Turk, where a total of 126 judges provided 3 redundant judgments for each system output. To provide additional quality control, our HITs were augmented with both positive and negative control compressions. For the positive control we used the reference compressions from our test set. Negative control was provided by adding a compression model based on random word deletions to the mix.

5.4.3 Evaluation Results

In Table 5.1 we compare our distributional similarity-augmented paraphrase-based compression systems to the plain pivoting-based baseline and the ILP approach

	CR	Meaning	Grammar
Reference	0.80	4.80	4.54
ILP	0.74	3.44	**3.41**
Paraphases	0.78	3.53	2.98
Paraphases + n-gram	0.80	3.65	3.16
Paraphases + Syntax	0.79	**3.70**	3.26
Paraphases + Syntax + n-gram	0.79	3.57	3.14
Random Deletions	0.78	2.91	2.53

Table 5.1: Results of the human evaluation on longer compressions: pairwise compression rates (CR), meaning and grammaticality scores. Bold indicates a statistically significance difference at $p < 0.05$.

[Clarke and Lapata, 2008]. The compression ratios of the paraphrasing systems are tuned to match the average compression ratio seen on the development and test set. Similarly, the ILP system is configured to loosely match this ratio, as to not overly constrain its search space.

Our results indicate that the paraphrase approach significantly outperforms ILP on meaning retention. However, the baseline system shows notable weaknesses in grammaticality. Adding the n-gram distributional similarity model to the paraphraser recovers some of the difference in grammaticality while simultaneously yielding some gain in the compressions' meaning retention. Moving to distributional similarity estimated on the syntactic feature-set yields additional improvement, despite the model's lower coverage.

Our results suggest that combining syntactic and n-gram based similarity scores

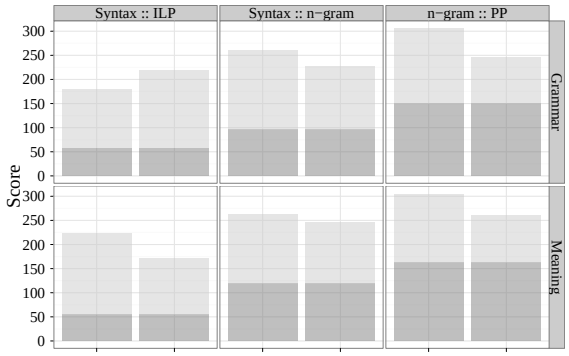

Figure 5.4: A pairwise breakdown of the human judgments comparing the systems. Dark grey regions show the number of times the two systems were tied, and light grey shows how many times one system was judged to be better than the other.

does not yield better results, but causes the compressions to drop in quality below either single similarity model. It is plausible to assume that for our newswire test set the comparatively lower coverage of Annotated Gigaword is sufficient to fully subsume the contributions of the n-gram model, and we thus need not expect significant gains from adding the n-gram similarity features. However, dropping below the syntactic similarity baseline clearly suggests issues with the model optimization. Further experiments on the combined effect of n-gram and syntactic similarities are required.

It is known that human evaluation scores correlate linearly with the compression ratio produced by a sentence compression system [Napoles et al., 2011a]. Thus, to ensure fairness in our comparisons, we produce a pairwise comparison breakdown that only takes into account compressions of almost identical length. We require the compressions to be within $\pm 10\%$ length of one another. Figure 5.4 shows the results

of this analysis, detailing the number of wins and ties in the human judgements.

We note that the Paraphrase + Syntax system's gains in meaning retention over both the baseline and the ILP system are still present in the pairwise breakdown. The gains over the paraphrasing baseline, as well as the improvement in meaning over ILP are statistically significant at $p < 0.05$, using the sign test. It is further interesting to observe that there is more substantial overlap between the baseline paraphraser and the n-gram model, while the syntax model appears to yield noticeably different output more often.

5.4.3.1 Example Compressions

Table 5.2 shows two example sentences drawn from our test set and the compressions produced by the different systems. It can be seen that both the paraphrase-based and ILP systems produce good quality results, with the paraphrase system retaining the meaning of the source sentence more accurately.

5.5 Conclusion

We presented a method to incorporate monolingual distributional similarity into linguistically informed paraphrases extracted from bilingual parallel data. Having extended the notion of similarity to discontiguous pattern with multi-word gaps, we investigated the effect of using feature-sets of varying complexity to compute

distributional similarity for our paraphrase collection. We conclude that, compared to a simple large-scale model, a rich, syntax-based feature-set, even with significantly lower coverage, noticeably improves output quality in a text-to-text generation task. Our syntactic method significantly improves grammaticality and meaning retention over a strong paraphrastic baseline, and offers substantial gains in meaning retention over a contemporary state-of-the-art, deletion-based system.

System	Sentences
Source	should these political developments have an impact on sports ?
Reference	should these political events affect sports ?
Paraphrases + Syntax	should these events have an impact on sports ?
Paraphrases + n-gram	these political developments impact on sports ?
Paraphrases	should these events impact on sports ?
ILP	political developments have an impact
Source	now we have to think and make a decision about our direction and choose only one way . thanks .
Reference	we should ponder it and decide our path and follow it , thanks .
Paraphrases + Syntax	now we think and decide on our way and choose one way . thanks .
Paraphrases + n-gram	now we have and decide on our way and choose one way . thanks .
Paraphrases	now we have and decide on our way and choose one way . thanks .
ILP	we have to think and make a decision and choose way thanks

Chapter 6

Constructing the Paraphrase Database

In this chapter we present the engineering details and data resources that allow us to scale up our syntactic pivoting method. We generate a gargantuan, multilingual collection of paraphrases, called the *ParaPhrase DataBase*, or *PPDB*.

This chapter reviews the creation of our paraphrase database draws on a large, composite bitext assembled from various bilingual corpora between English and a number of languages. We use even larger monolingual data to compute additional features for our paraphrase pairs.

Paraphrase extraction on the scale of PPDB requires substantial amounts of engineering, both for the extraction process itself, and to make very large paraphrase SCFGs usable for applications that require quick access, like text-to-text generation.

We present details on the engineering behind the extraction of PPDB. Both implementations are improvements on the state of the art, yielding substantial gains in performance and reductions in memory use.

In Section 6.1 we discuss the 1.0 release of the English paraphrase database,[1] which has 170 million paraphrase rules [Ganitkevitch et al., 2013] and is currently the largest publicly available paraphrase corpus.

We further present the extension of our paraphrase extraction method to non-English languages. Our multilingual paraphrase resource [Ganitkevitch and Callison-Burch, 2014] massively expands on the initial PPDB release. It includes collections of paraphrases for 23 additional languages: Arabic, Bulgarian, Chinese, Czech, Dutch, Estonian, Finnish, French, German, Greek, Hungarian, Italian, Latvian, Lithuanian, Polish, Portuguese, Romanian, Russian, Slovak, Slovenian, Spanish,[2] Swedish, and Urdu. The multilingual portion of the paraphrase database is freely available from `paraphrase.org`. Section 6.5 discusses the extensions necessary to extract this multilingual paraphrase collection.

6.1 The Paraphrase Database

Collections of paraphrases and paraphrase-like expression pairs have been presented and released in the past. These resources include the DIRT database which

[1] Freely available at `http://paraphrase.org`.

[2] The Spanish PPDB was made available alongside the English version [Ganitkevitch et al., 2013].

	Identity	Paraphrases	Total
Lexical	0.6M	7.6M	8.1M
Phrasal	4.9M	68.4M	73.2M
Syntactic	46.5M	93.6M	140.1M
All	52.0M	169.6M	221.4M

Table 6.1: A breakdown of PPDB:Eng size by paraphrase type. We distinguish lexical (i.e. one-word) paraphrases, phrasal paraphrases and syntactically labeled paraphrase patterns.

contains 12 million paraphrase rules [Lin and Pantel, 2001], and the MSR paraphrase phrase table which has 13 million rules [Dolan et al., 2004], among others. Although several groups have independently extracted paraphrases using Bannard and Callison-Burch [2005]'s bilingual pivoting technique (see Zhou et al. [2006], Riezler et al. [2007], Snover et al. [2010], among others), there has never been an official release of their resource.

Table 6.1 presents a breakdown of the English paraphrase database (PPDB:Eng), by paraphrase type. As previously established, we distinguish *lexical* (a single word), *phrasal* (a continuous string of words), and *syntactic* paraphrases (expressions that may contain both words and nonterminals), and separate out identity paraphrases. While we list lexical and phrasal paraphrases separately, it is possible that a single word paraphrases as a multi-word phrase and vice versa – so long they share the same syntactic label.

6.1.1 Comparison to Related Resources

In this section we present a brief qualitative comparison of PPDB with similar resources. We contrast the paraphrase database with WordNet [Fellbaum, 1998], PATTY [Nakashole et al., 2012], and a collection of entailment rules produced by applying the approach of Berant et al. [2012] to the REVERB set of relation expressions [Fader et al., 2011].

6.1.1.1 WordNet

WordNet [Fellbaum, 1998] is in many ways the ur-resource for language synonymy and entailment hierarchies. It is a fully manually-created data set that groups words and short multi-word expressions into synonym sets (*synsets*), where all member expressions share the same meaning. The synsets are themselves organized in a hierarchy of entailment and subsumption. An word can belong to multiple synsets (for ambiguous or polysemous words). Words' membership to synsets is further qualified with part of speech tags. The word "dog", for instance belongs to at least two synsets with distinct parts of speech, as $\langle dog, NOUN \rangle$ and $\langle dog, VERB \rangle$.

PPDB 1.0 does not contain any entailment information, simply rendering every pair as a quasi-paraphrase with an associated score (this is amended by Pavlick et al. [2015b], who add automatically inferred entailment relations and improved quality estimation to the resource). We therefore limit ourselves to an intersection experiment between PPDB and the synsets of WordNet 3.0.

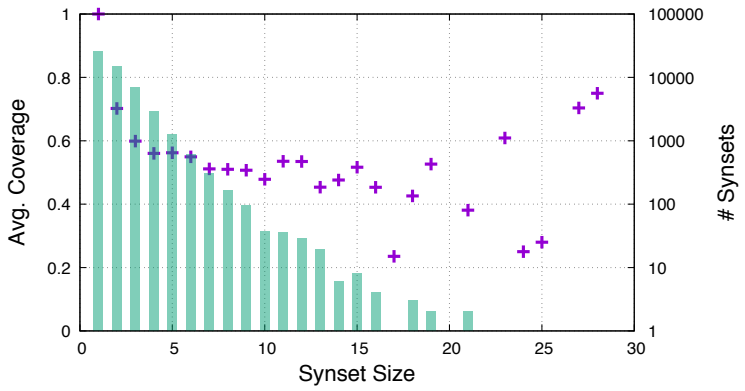

Figure 6.1: The average coverage of WordNet in PPDB 1.0. We can see that PPDB covers an average of about 50% of the touched WordNet synsets even for larger synsets.

For this, we test each paraphrase pair in PPDB against WordNet – if the PPDB source phrase e_1 is present in any synsets $\textsc{Syn}(e_1)$, we check all of these synsets for the presence of the PPDB paraphrase e_2. Since WordNet's synset entries are lemmatized, we use NLTK's WordNet lemmatizer [Bird and Loper, 2004] to lemmatize the e_i for our queries. We respect the syntactic labels PPDB extracts for the paraphrases when testing against WordNet, mapping the various forms of nouns and nounphrases to WordNet's noun tag etc. We count how many of the lemmas present in a WordNet synset are also covered by a PPDB paraphrase overlapping with said synset. Since WordNet is largely limited to single word or short phrasal expressions, PPDB's syntactic paraphrases do not factor in this experiment.

Figure 6.1 visualizes the results for our intersection experiment. We find that of

Syntactic Label	Source	Paraphrase
NN	freeze-drying	lyophilization
NN	clotting	coagulation
JJ	air-to-surface	air-to-ground
NNS	assailants	attackers
NN	foreword	preface
NN	uterus	womb
VBN	immunized	vaccinated
JJ	laudable	praiseworthy
VBD	interpreted	construed
VBP	disconnect	unplug

Table 6.2: Examples of PPDB paraphrase pairs also found in WordNet synsets.

the 117,000 synsets that make up WordNet 3.0, PPDB overlaps with 52,891, or about 45%. Of these, 25,599 are singelton synsets, i.e. synsets with only a single member which therefore do not have any paraphrastic information. For larger synsets, we can see a pattern emerge: even as the size of the matched synset grows, PPDB quite consistently covers about 50% of the lemmas in the set. Overall, 123,144 unique paraphrase pairs in PPDB match synsets in WordNet 3.0. When we filter this set to omit paraphrases that map to the same lemma, such as $\langle e - mail, e - mails \rangle$ or have a character edit distance of less than 3 (to drop simple spelling variations like $\langle sulfur, sulphur \rangle$), we are still left with 86,635 unique paraphrase pairs. Table 6.2 shows a few examples from the overlap set.

Given the gold-standard quality of WordNet and its high recall over textual representations for the concepts it covers, this is an encouraging result. It speaks to the

presence of high-quality paraphrases in PPDB, as well as a good coverage of concepts with a wide variety of lexicalizations.

We further investigate the non-overlap between PPDB contributes over WordNet, using a few samples. For instance, for the adjective *absurd* WordNet's synsets yield the following 9 synonyms:

> derisory, ludicrous, idiotic, preposterous, cockeyed, ridiculous, absurd, laughable, nonsensical.

By comparison, PPDB's proposed set of paraphrases for *absurd* as an adjectival expression (i.e. we allow for *ADJP* and *JJ*) is missing *cockeyed*. It however contains 99 additional paraphrases given here in order of PPDB's goodness estimate:

> nonsense, senseless, quite absurd, unreasonable, pointless, so pointless, perverse, irrational, aberrant, patently absurd, farcical, foolish, silly, illogical, absolutely absurd, too absurd, grotesque, just nonsense, completely absurd, just absurd, pretty silly, plain absurd, insane, stupid, incongruous, bizarre, crazy, paradoxical, totally absurd, meaningless, counterintuitive, quite ridiculous, wrong-headed, ungodly, fallacious, outrageous, injudicious, so stupid, surreal, odd, mad, abhorrent, wrong, ironic, inconceivable, exorbitant, futile, unthinkable, undignified, so ridiculous, rubbish, unwise, rather paradoxical, useless, unfair, incredible, madness, just ridiculous, obscene, idiot, misguided, dumb, mindless, unwarranted, excessive, inappropriate, repugnant, risible, strange, contradictory, really ridiculous, wild, unjust, pretty sappy, "ridiculous, huh," "ridiculous, jenkins," extraordinary, disparate, unconscionable, pathetic, false, totally ridiculous, stupid in this, really silly, ridiculous , okay, so silly, absolutely stupid, absolutely ridiculous, that ridiculous, bloody ridiculous, morally bankrupt, so dorky, completely ridiculous, ridiculously low, too ridiculous, just stupid, kind of stupid, most ridiculous, so lame.

A cursory examination quickly shows that despite the large number of paraphrases, the quality remains quite high throughout. The candidates produced by PPDB largely

cover the relatively formal vocabulary and close semantic similarity in WordNet's synsets, but also expand into more casual and conversational variations. This holds especially we get into the lower-scoring paraphrase candidates.

We see a similar difference in comparing the paraphrases for the noun *contraction*. WordNet's synsets offer 5 synonyms:

> condensation, muscular contraction, muscle contraction, compression, contraction.

Here, PPDB matches *compression*, and further contains 57 additional candidate paraphrases (given in order of estimated quality):

> contracture, shrinkage, downturn, shrinking, contractile, twitch, deflation, retrenchment, crunch, decline, slowdown, recession, tightening, constriction, squeeze, contracting, reduction, the decline, decrease, curtailment, narrowing, contract, austerity, stricture, a fall, collapse, slump, drop, fall, convulsion, drawdown, meltdown, slowing, his contract, the recession, downsizing, the reduction, shrink, celebration, weakening, retreat, borrowing, stringency, contradiction, cut, contamination, setback, shortening, slow-down, compressive, a decrease, the conclusion, a recession, atrophy, a slowdown, a decline, a reduction.

Again, the overall quality of the PPDB paraphrases is quite high. Only towards the tail end do we see not entirely grammatical substitutions proposed, though they remain semantically sound. It appears that in this case WordNet's synonym sets take into account only a few very specific (medical) semantics of *contraction*. Meanwhile PPDB benefits from a large amount of newswire data to extract high quality paraphrases for economy-related semantics.

These examples highlight some weaknesses of manually curated resources such as WordNet, and exemplify the advantages of large-scale datasets like PPDB. While

the contents of WordNet is beyond reproach, it is limited to the vocabulary and the senses its curators included. An NLP system replying on such a resource may run into not only coverage issues, but also inadvertent bias stemming from a mismatch between what senses and domains a manually curated resource contains, and the actual domain distribution on application data.

At-scale, automatically extracted resources like PPDB help mitigate some of this problem provided the resource extraction approach saw data similar to the application domain. A further strength of data-driven resources is that common forms in which concepts occur in text are being captured. Increasingly, the data processed by NLP systems is informal text. Resources like PPDB, which connect a concept's canonical form with its more varied, in-the-wild textual expressions can greatly facilitate this task.

6.1.1.2 REVERB Textual Entailment Rules

This (to the best of our knowledge not officially named) resource stems from Berant [2012]'s application of their algorithms for inference of entailment relations onto a large set of relation expressions stemming from the REVERB data set [Fader et al., 2011].

The data set contains scored pairs of patterns b_1, b_2, where each b_i is the (lemmatized) lexicalization of a relation between two entities. A high score means that the predictive model estimated b_1 to likely entail b_2. An example relation in this data set

is:

$$X \text{ accept } Y \rightarrow Y \text{ be worn by } X,$$

here implying that X is a person and Y is an award (or otherwise an acceptable wearable).

It is worth noting that the resources differ in both style and purpose: PPDB aims to capture quasi-paraphrases: rewrites that strive to preserve the core meaning of a text, while also being fully lexicalized and grammatically viable. The REVERB entailment set expressly tries to capture entailment relationships where the entailed expression may (and is intended to) causally stray from the meaning of the source expression (see above example, where the two phrases clearly describe distinct semantic events). We therefore expect significant portions of the entailment relations set to be out of scope for PPDB. In addition to this, PPDB's extraction approach makes use of sentence parallelism to acquire paraphrases, which is a weaker signal than temporal closeness and entity co-occurrence. The stronger signals used for REVERB extraction allow it to more confidently include rare, long phrases. Many such phrases are likely to be dropped by the the thresholding heuristics used for PPDB, because it would be hard to reliably estimate paraphrase probabilities for rarely occurring expression.

The patterns in the REVERB entailment sets are lemmatizedphrases and suggest a bookending by entity slots, as shown above. We limit our comparison to PPDB's lexical and phrasal portions, since for any syntactic paraphrase with the nonterminals bookending a phrasal paraphrase, PPDB's alignment and extraction approach guar-

Label	Source	Paraphrase
VP/VP	make an important contribution to	make a significant contribution to
VP/NP	consider the relationship between	examine the relationship between
VP/NP	enhance the effectiveness of	increase the efficiency of
VP/NP	pays great attention to	places great importance on

Table 6.3: Examples of PPDB paraphrase pairs matching entailment pairs in the REVERB dataset.

antees that we have also extracted the phrasal portion by itself. As in Section 6.1.1.1, we use NLTK's WordNet lemmatizer to make the fully lexicalized PPDB phrases comparable to the entailment set.

As intuitively expected, the overlap between PPDB and the entailment sets is relatively low. We record only 0.59% overlap with the entailment set generated by Berant [2012]'s local algorithm (91,455 phrase pairs matched of over 15 million, when using the recommended thresholding for the set). However, the overlap with the smaller, higher-quality sets extracted using Berant's global algorithms are much higher: 6% (16,792 of 275,682) for the HTL set, and 11% (11,596 of 102,565) for the TNCF data set. A possible reading of this result is that the higher-precision global algorithms Berant proposes weed out more far-fetched entailment pair candidates, leaving us with a higher proportion of direct paraphrases of the kind that is more readily extracted by PPDB's approach.

In Table 6.3 we show a few examples of the overlapping phrase pairs extracted.

Label	Count
X	14,104
VP/NP	10,687
VP/PP	5,533
VP/VP	3,330
VP	2,790
VB	1,410
$VP/SBAR$	1,096
VP/PRP	641
VP/NN	538
VP/VB	531

Table 6.4: Most frequent labels in the PPDB overlap with the local algorithm REVERB entailment set. The CCG-style slashed categories most frequently observed are consistent with the expected *X-phrase-Y* bookending pattern. The large number of fallback wildcard labels X reflects the often long and difficult-to-parse phrases.

It is notable that the phrases often differ in only a single word. For the purpose of paraphrase extraction, these would be more robustly and generalizably by just that particular lexical paraphrase, e.g. $\langle$*important, significant*$\rangle$ for the first example in the table. These paraphrases exist in PPDB, but they are insufficient for the purpose of the REVERB entailment set because they do not adequately express a full relation between two entities.

In Table 6.4 we list the most common syntactic labels for the overlap paraphrases. The high proportion of verb phrases missing a constituent to the right accurately reflects the bookending pattern that underlies the entailment pairs data set. We

find an uncommonly high proportion of wildcard-labeled phrases (X), a label PPDB assigns when no full constituent or single-slashed CCG label could be found. This is likely caused by the degradation in automated parsing quality on long, complex sentences like the ones giving rise to phrases that match the REVERB data.

6.1.1.3 PATTY

PATTY [Nakashole et al., 2012] is a resource that offers automatically extracted relational expressions that appear in conjunction with particular entity types, or express particular relations. For a repository of entity and relation types, PATTY draws on annotated resources like DBpedia [Auer et al., 2007] and YAGO [Suchanek et al., 2007]. PATTY further groups synonymous relational expressions together in sets, which gives us an interesting point of comparison to PPDB.

We focus our analysis on the Wikipedia and New York Times-based sets of text patterns PATTY contains. The two sets contain 350,569 and 86,982 sets of relational expressions, respectively. PATTY's extraction method frequently applied generalization operations that replace not semantically portent words with part-of-speech tags to improve generalization. For instance *won five awards* may be generalized to *won NUM awards*. In the synonym clustering, these generalizations are not usually synchronous: *won NUM awards* might be grouped with *won ADJ awards* and *won awards*. While this pattern greatly improves PATTYs coverage and applicability, it results in non-paraphrastic pattern groupings that PPDB cannot represent. We there-

PPDB Label	Source	Paraphrase
–	also written songs for artists	has collaborated with artists including
–	has produced artists like	has played bass including with
–	also gueststarred on television shows including	made guest appearances on series including
–	also taught for	recently retired professor emeritus prior to had taught at
–	became head	was head at
VP/*NP*	was established under	was established by
VP/*NP*	become involved in	been involved with

Table 6.5: Examples of PATTY relation pattern pairs missing from (top) and included in (bottom) PPDB.

fore limit our analysis to patterns with no generalization placeholders. This severely limits the overlap set we consider: of the Wikipedia-based patterns, only 12,216 sets have more than one member without any patternization applied. For the NYT-based data the number of remaining multi-element pattern sets is 2,950. Nonetheless, many of the remaining pattern sets are highly fertile. Expanded into possible paraphrase pairs this still leaves us with 357,598 paraphrase pairs to compare against for the Wikipedia set, and 44,920 for the NYT patterns.

Still, the overlap between PATTY and PPDB based on this comparison is quite slim: PPDB only covers 0.39% of the expanded Wikipedia pattern pairs, and 1.9% of the NYT pattern pairs. Table 6.5 shows a few examples of patterns included in and missing from PPDB. Upon examining the PATTY pattern pairs missing from PPDB, we find that indeed the textual expressions are frequently not directly substitutable paraphrases. In other cases, however, the pattern pairs are paraphrastic and highly interesting, but (as similarly encountered in Section 6.1.1.2) too specific and complex to be reliably extracted with just our pivot approach.

Our cursory examination suggests that the apt use of strong entity relationship signals, as demonstrated by both Nakashole et al. [2012] and Fader et al. [2011] can yield complex paraphrases that are hard to recover with purely syntactic and pivoting-based methods. Extending at-scale paraphrase extraction methods to use entity annotations over large text corpora as a source for paraphrase candidates makes for an interesting avenue of future work.

6.1.2 Propbank Coverage

To estimate the usefulness of PPDB as a resource for tasks like semantic role labeling or parsing, we analyze its coverage of Propbank predicates and predicate-argument tuples [Kingsbury and Palmer, 2002]. We use the Penn Treebank [Marcus et al., 1993] to map Propbank annotations to patterns which allow us to search PPDB:Eng for paraphrases that match the annotated predicate. Figure 6.2 illustrates

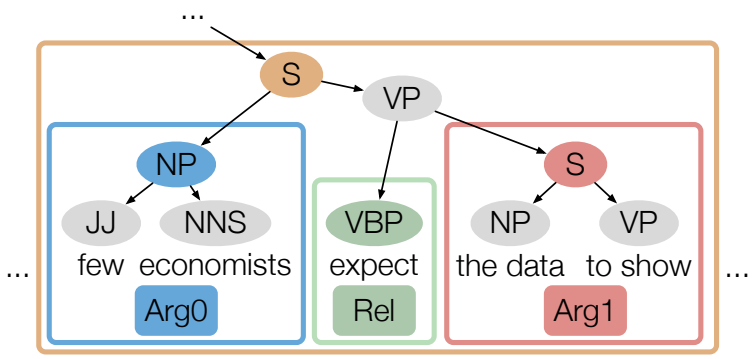

Figure 6.2: To inspect our coverage, we use the Penn Treebank's parses to map from Propbank annotations to PPDB's syntactic patterns. For the above annotation predicate, we extract $VBP \rightarrow$ expect, which is matched by paraphrase rules like $VBP \rightarrow$ expect | anticipate and $VBP \rightarrow$ expect | hypothesize. To search for the entire relation, we replace the argument spans with syntactic nonterminals. Here, we obtain $S \rightarrow NP$ expect S, for which PPDB has matching rules like $S \rightarrow NP$ expect S | NP would hope S, and $S \rightarrow NP$ expect S | NP trust S. This allows us to apply sophisticated paraphrases to the predicate while capturing its arguments in a generalized fashion.

this mapping.

In order to quantify PPDB:Eng's precision-recall tradeoff in this context, we perform a sweep over our collection, beginning with the full set of paraphrase pairs and incrementally discarding the lowest-scoring ones. To guide our sweep, we use the canonical score described in Section 6.2.

The top graph in Figure 6.3 shows PPDB's coverage of predicates (e.g. $VBP \rightarrow$

expect) at the type level (i.e. counting over the set of distinct predicates that appear in Propbank), as well as the token level (i.e. counting each predicate occurrence in the corpus). We find that PPDB has a predicate type recall of up to 52% (which accounts for 97.5% of the annotated predicates in Propbank). We can therefore conclude that, while there is a substantial number of the predicate types not covered by PPDB:Eng, they tend to be extremely rare. For practical applications (on data of domains similar to Propbank), PPDB:Eng therefore appears to provide good coverage.

To gauge the quality of our paraphrases, we manually judged 1900 randomly sampled predicate paraphrases on a scale of 1 to 5, 5 being the best. The bottom graph in Figure 6.3 plots the resulting human score average against the sweep used in the coverage experiment. It is clear that even with the simple weighing approach we choose for our sweep, the PPDB scores show a clear correlation with human judgements. Therefore they can be used to bias the collection towards greater recall or higher precision.

We also keep track of average number of paraphrases per covered predicate type for varying pruning levels, also shown in the top of Figure 6.3. When looking at the full PPDB:Eng set, we average over 150 paraphrases per predicate. Naturally, as we cull worse-scoring paraphrases from consideration, this number drops. However, the plots demonstrate that even when using a demanding threshold for average human score, we maintain good coverage of Propbank and a substantial number of paraphrases per predicate type. For instance, when requiring an average human score of 3.5, we

still cover 49.9% of predicate types, accounting for over 96% of predicate occurrences, and average over 23 paraphrases per predicate. For an average score threshold of 4.0 this only slightly drops to a still-respectable 44.3% of types, 92.2% of tokens, and 8.8 paraphrases on average.

We further extend the experiment to full predicate-argument relations with up to two arguments (e.g. $S \rightarrow NNS$ expect S). Here, we require that the syntactic paraphrase rule cover the entire relation, with correctly labeled nonterminals for each of the arguments. We limit ourselves to relations with two or less arguments, since PPDB itself is limited to syntactic rules of arity 2. As shown in Figure 6.4, we obtain a 27% type coverage rate that accounts for about 40% of tokens. As for the predicate-only case, both coverage rates hold even as we prune the database down to only contain high precision paraphrases. With the full PPDB:Eng set we achieve about 60 paraphrases per predicate-argument expression. We still maintain around 10 paraphrases per predicate-argument expression and virtually undiminished coverage at stricter pruning thresholds that retain only higher-quality paraphrases.

We can conclude that PPDB:Eng has a good coverage of the Propbank data set, providing a rich diversity of high-quality paraphrases. This is indicative of the resource's usefulness in a wide variety of natural language processing work.

```
[ADJP] ||| hard ||| pretty difficult ||| Abstract=0
Adjacent=0 CharCountDiff=12 CharLogCR=1.38629 ContainsX=0
GlueRule=0 Identity=0 Lex(e|f)=4.86514 Lex(f|e)=11.05531
Lexical=1 LogCount=0.69315 Monotonic=1 PhrasePenalty=1
RarityPenalty=0.00012 SourceTerminalsButNoTarget=0 SourceWords=1
TargetTerminalsButNoSource=0 TargetWords=2 UnalignedSource=0
UnalignedTarget=0 WordCountDiff=1 WordLenDiff=3.50000
WordLogCR=0.69315 p(LHS|e)=0 p(LHS|f)=1.84819 p(e|LHS)=15.47166
p(e|f)=10.47819 p(e|f,LHS)=8.36974 p(f|LHS)=8.42517
p(f|e)=1.58351 p(f|e,LHS)=1.32325 ||| 0-0 0-1
```

Table 6.6: An example line from the PPDB release files. The string " ||| " is used as a field separator. Features and their values are given as a space-separated list of key-value pairs. PPDB uses the features introduced in Chapters 4 and 5.

6.2 Curating the PPDB Release

We release the paraphrase database as an SCFG in a gzipped text format, encoding one paraphrase rule per line, along with the features extracted for it. Table 6.6 shows an example line from PPDB. The fields, separated by " ||| " are:

1. The *left-hand side (LHS)* syntactic label of the paraphrase rule.

2. The *source* phrase, the expression being paraphrased. Following the terminology of statistical machine translation, this side of the rule is denoted as f in the feature names.

3. The *target* phrase, i.e. the paraphrase of the source. the target is referred to as e in the feature names.

4. The features computed for the paraphrase rule, as a space-separated list of key-value pairs. We use the conditional probability and utility features discussed in Chapter 4, as well as the distributional scores introduced in Chapter 5.

5. The word alignment for the paraphrase rule, as a space-delimited list of zero-indexed source-to-target pairs.

We recognize that our paraphrase collection can feel unwieldy large, especially for Arabic, English, French, Chinese, Spanish, and Russian. We therefore divide the sets into different sizes. These are named by size: S (small), M (medium), L (large), XL (extra large), XXL (double extra large), and XXXL (Royale with cheese). Each step

up in size is designed to roughly double the number of paraphrases across each type: lexical, phrasal, and syntactic. The larger sizes subsume the smaller sizes.

Before partitioning, we sort the paraphrases according to a canonical *score*. This helps to ensure that the higher quality paraphrases are included in the smaller sized sets. The larger sized sets include these high precision paraphrases, but also contain paraphrases that are not as high quality (but which do offer better coverage or higher recall). The choice of which size to use will depend on the needs of a particular application. The score used to sort PPDB is a lower-is-better *cost*. We compute it heuristically as:

$$\begin{aligned} s_{ppdb}(e|f) &= p(e|f) + p(f|e) + p(e|f, LHS) + p(f|e, LHS) \\ &\quad +100 \cdot RarityPenalty + 0.3 \cdot p(LHS|e) + 0.3 \cdot p(LHS|f). \end{aligned} \quad (6.1)$$

The selection of features and the values for their weights are chosen in an ad hoc fashion, based on our intuitions about which features seem to be useful for sorting higher quality paraphrases from lower quality paraphrases. A more principled approach would be to collect a set of judgments about the quality of a random sample of the paraphrases, and then use linear regression to fit the weights to the human judgments, for instance, in a similar fashion to [Malakasiotis and Androutsopoulos, 2011]. We leave that task to users of our resource. As we provide the full feature set, users can re-sort the paraphrase database to fit native-speaker judgments or the needs of a specific NLP task.

To distinguish between the multilingual variants of PPDB, we use language tags

that are coded following ISO 639-2, using the *terminology* ("T") code where applicable.[3] For instance, we refer to the English PPDB as PPDB:Eng, while the Spanish and German PPDB versions are dubbed PPDB:Spa and PPDB:Deu, respectively. All versions of PPDB are freely available online at `paraphrase.org`.

6.3 Data Sets Underlying the PPDB Release

Following the methods we introduced in the previous chapters, the English portion of PPDB is derived by:

- Extracting lexical, phrasal, and syntactic paraphrases from large bilingual parallel corpora (with associated paraphrase probabilities). We rely on statistical constituency parsers to generate syntactic parses for the English side of the bitext. These parses are our basis for computing syntactic labels for our paraphrases.

- Computing distributional similarity scores for each of the paraphrases using monolingual data like the Google n-grams and the Annotated Gigaword [Brants and Franz, 2006, **?**]. corpus.

For the multilingual PPDB portions we rely solely on the bilingual data.

[3]For 21 languages, the ISO 639-2 standard lists two codes: the *bibliographic* ("B") and *terminology* ("T") codes. We use the latter. The full list can be found at `https://www.loc.gov/standards/iso639-2`.

6.3.1 Bilingual Data

We obtain the paraphrases for both the English and the multilingual versions of PPDB by pivoting through bilingual parallel corpora. The bitext collection we use totals over 100 million sentence pairs between English and the various foreign languages. It combines several English-to-foreign data sets: Europarl v7 [Koehn, 2005], consisting of bitexts for the 19 European languages, the 10^9 French-English corpus [Callison-Burch et al., 2009], the Czech, German, Spanish and French portions of the News Commentary data [Koehn and Schroeder, 2007], the United Nations French- and Spanish-English parallel corpora [Eisele and Chen, 2010], the JRC Acquis corpus [Steinberger et al., 2006], Chinese and Arabic newswire corpora used for the GALE machine translation campaign,[4] parallel Urdu-English data from the NIST translation task,[5] the French portion of the OpenSubtitles corpus [Tiedemann, 2009], and a collection of Spanish-English translation memories provided by TAUS.[6] Table 6.7 shows a breakdown of the bitext sizes across the various languages, listing number of parallel sentence pairs, English and foreign word counts, and the corpora that contribute to the language's portion in our data set.

The resulting composite parallel corpus has more than 106 million sentence pairs, over 2 billion English words, and spans 23 pivot languages. To apply the pivoting technique to this multilingual data, we treat the various pivot languages as a joint

[4] `http://projects.ldc.upenn.edu/gale/data/Catalog.html`
[5] LDC Catalog No. LDC2010T23
[6] `http://www.translationautomation.com/`

Non-English language. This simplifying assumption allows us to share statistics across the different languages and apply Equation 3.3 unaltered.

6.3.2 Monolingual Data

The bilingual pivoting approach anchors paraphrases that share an interpretation because of a shared foreign phrase. Paraphrasing methods based on monolingual text corpora, like DIRT [Lin and Pantel, 2001], measure the similarity of phrases based on distributional similarity. This results in a range of different types of phrases, including paraphrases, inference rules and antonyms. For instance, for *thrown into prison* DIRT extracts good paraphrases like *arrested, detained*, and *jailed*. However, it also extracts phrases that are temporarily or causally related like *began the trial of, cracked down on, interrogated, prosecuted* and *ordered the execution of*, because they have similar distributional properties. It can also contain distributionally related antonyms. Since bilingual pivoting rarely extracts these non-paraphrases, we can use monolingual distributional similarity to re-rank paraphrases extracted from bitexts (following Chan et al. [2011]) or incorporate a set of distributional similarity scores as features in our log-linear model. For discontiguous paraphrase patterns we introduce a novel method that decomposes the pattern into smaller aligned elements and computes their distributional similarity [Ganitkevitch et al., 2012b].

Each similarity score relies on precomputed distributional signatures that describe the contexts that a phrase occurs in. To describe a phrase e, we gather counts for a

Language	Sentence Pairs	Foreign Words	English Words	Corpora
Arabic	9,542,054	205,508,319	204,862,233	GALE
Bulgarian	406,934	9,306,037	9,886,401	Europarl-v7
Chinese	11,097,351	229,364,807	244,690,254	GALE
Czech	596,189	12,285,430	14,277,300	Europarl-v7, News Commentary
Dutch	1,997,775	49,533,217	50,661,711	Europarl-v7
Estonian	651,746	11,214,489	15,685,939	Europarl-v7
Finnish	1,924,942	32,330,289	47,526,505	Europarl-v7
French	52,004,519	932,475,412	821,546,279	Europarl-v7, 10^9 word parallel corpus, JRC, News Commentary, UN, OpenSubtitles
German	1,720,573	39,301,114	41,212,173	Europarl-v7, News Commentary
Greek	1,235,976	32,031,068	31,939,677	Europarl-v7
Hungarian	624,934	12,422,462	15,096,547	Europarl-v7
Italian	1,909,115	48,011,261	49,732,033	Europarl-v7
Latvian	637,599	11,957,078	15,412,186	Europarl-v7
Lithuanian	635,146	11,394,858	15,342,163	Europarl-v7
Polish	632,565	12,815,795	15,269,016	Europarl-v7
Portugese	1,960,407	49,961,396	49,283,373	Europarl-v7
Romanian	399,375	9,628,356	9,710,439	Europarl-v7
Russian[7]	2,376,138	40,765,979	43,273,593	CommonCrawl, Yandex 1M corpus, News Commentary
Slovak	640,715	15,442,442	12,942,700	Europarl-v7
Slovenian	623,490	12,525,860	15,021,689	Europarl-v7
Spanish	15,074,434	339,957,943	297,440,364	Europarl-v7, News Commentary, TAUS, UN

set of contextual features for each occurrence of e in a corpus. Writing the context vector for the i-th occurrence of e as $\vec{s}_{e,i}$, we can aggregate over all occurrences of e, resulting in a *distributional* signature for e, $\vec{s}_e = \sum_i \vec{s}_{e,i}$. Following the intuition that phrases with similar meanings occur in similar contexts, we can then quantify the goodness of e' as a paraphrase of e by computing the cosine similarity between their distributional signatures:

$$sim(e, e') = \frac{\vec{s}_e \cdot \vec{s}_{e'}}{|\vec{s}_e||\vec{s}_{e'}|}. \tag{6.2}$$

A wide variety of features have been used to describe the distributional context of a phrase. Rich, linguistically informed feature-sets that rely on dependency and constituency parses, part-of-speech tags, or lemmatization have been proposed in work such as by Church and Hanks [1991] and Lin and Pantel [2001]. For instance, a phrase is described by the various syntactic relations such as: "what verbs have this phrase as the subject?", or "what adjectives modify this phrase?". Other work has used simpler n-gram features, e.g. "what words or bigrams have we seen to the left of this phrase?". A substantial body of work has focussed on using this type of feature-set for a variety of purposes in NLP [Lapata and Keller, 2005, Bhagat and Ravichandran, 2008, Lin et al., 2010, Van Durme and Lall, 2010].

For PPDB, we compute n-gram-based context signatures for the 200 million most frequent phrases in the Google n-gram corpus [Brants and Franz, 2006, Lin et al., 2010]. For each phrase we note the word to its immediate left and right. The extracted context vectors are illustrated in Figure 6.5. We generate 256-bit signatures for the

Google n-gram-based phrases.

We also extract richer linguistic signatures for 175 million phrases in the Annotated Gigaword corpus [**?**]. Our features extend beyond those previously used in the work by Ganitkevitch et al. [2012b]. They are:

- n-gram based features for words seen to the left and right of a phrase. Here, we consider position-aware lexical, lemma-based, part-of-speech, and named entity class unigram and bigram features, drawn from a three-word window to the right and left of the phrase.

- Incoming and outgoing (wrt. the phrase) dependency link features, labeled with the corresponding lexical item, lemmata and POS. These are generated for basic Stanford dependencies, as well as for the collapsed and collapsed-propagated variants.

- Syntactic features for any constituents governing the phrase, as well as for CCG-style slashed constituent labels for the phrase.

Figure 6.6 illustrates the feature extraction for an example phrase. For PPDB, all features collected on Annotated Gigaword are combined into a single 256-bit signature.

6.4 Engineering Underlying the Paraphrase Database

6.4.1 Efficient Paraphrase Extraction

Prior to the extraction of PPDB we have limited ourselves to fairly small source corpora. The French-English portion of Europarl [Koehn, 2005] used in the previous chapters, for instance, only counts 1.7 million sentence pairs. The initial paraphrase extraction experiments on Europarl therefore were feasible with a simple pipeline and ran on single machines. However, even with such a moderately proportioned data set strict pruning quickly becomes necessary to control the computational cost and intermediate output sizes.

In order to scale up to larger corpora, we need to more systematically exploit parallelism, and ideally implement the extraction in a flexibly scalable framework to adjust to increasing data size and fully exploit available computational resources. The Hadoop framework offers a widely-used solution for this. Based on the Map-Reduce paradigm [Dean and Ghemawat, 2008], Hadoop can be used for distributed batch processing of large data sets. It has been successfully used for natural language processing in both industry and academic environments [Lin and Dyer, 2010].

To implement pivot-based paraphrase extraction in Hadoop, we build on the Joshua decoder's Thrax toolkit [Weese et al., 2011]. Thrax implements the extrac-

tion of hierarchical and syntactic SCFGs for translation in Hadoop. We present an extended version of Thrax that implements distributed, Hadoop-based paraphrase extraction of syntactic SCFGs via the pivoting approach [Ganitkevitch et al., 2012a]. Our toolkit is capable of extracting syntactically informed paraphrase grammars at a large scale. We further present an optimized version of Thrax's paraphrase extraction that streamlines the entire toolkit [Post et al., 2013].

6.4.1.1 Pivot-Based Paraphrase Extraction in Map-Reduce

Pivot-based paraphrase extraction is a natural fit for the map-reduce paradigm. We can divide the paraphrase extraction process into two steps: *pivoting*, and *aggregation*. We outline the map-reduce implementation of our method on the example of extracting English paraphrases from a French-English bitext. Initially, we simply run the standard translation grammar extraction process that Thrax implements. From there, now given a collection of syntactically labeled French-English translation rules and their probability estimates, we begin the paraphrase extraction.

6.4.1.1.1 The Pivoting Step

This stage will start with the given translation SCFG, and seeks to look at all English expressions that share a given French translation. The map-reduce framework's *reduce* step straightforwardly implements this: we define a *key* for each datum (in this case: the translation rules), and Hadoop guarantees that all rules sharing

the same key will be processed *en bloc* on a single worker machine. By defining the reducer key to be the French source side of a translation rule plus the rule's left-hand side label, we achieve this effect. The group of rules is then processed by recombining the various English translations into paraphrase rules and estimating their associated paraphrase probabilities from the translation probabilities.

Due to the long tail of translations in unpruned translation grammars and the combinatorial effect of pivoting, paraphrase grammars, and especially the intermediate output of the pivoting step can easily grow very large. We implement a simple feature-level pruning approach that allows the user to specify upper or lower bounds for any pivoted feature. If a paraphrase rule is not within these bounds, it is discarded. Furthermore, knowing the paraphrase probability threshold we can pre-prune the translation rules before recombination. The multiplicative estimation we use for our paraphrase probabilities,

$$p_f(e_1|e_2) \approx p(e_1|f)p(f|e_2), \tag{6.3}$$

guarantees that $p_f(e_1|e_2) \leq p(f|e_2)$ and $p_f(e_1|e_2) \leq p(e_1|f)$. We can therefore drop weak translation rules early and achieve an additional reduction in processing load without impacting the outcome of our pruning heuristic.

The output of the pivoting step is a list of paraphrase rules estimated from single translation-pivot links. Since the same paraphrase rule can be obtained by pivoting through different foreign phrases f, this list is highly likely to contain duplicates.

6.4.1.1.2 The Aggregation Step

In the aggregation stage we combine all duplicate paraphrase rules that were generated in the pivoting step and aggregate their probabilities to compute the final paraphrase probability estimates. Here, the key is simply the entire paraphrase rule. For the probability-based features we typically will, following Equation 3.3, sum over the links. For other features we can define a different kind of aggregative behavior:

- For our estimate of the paraphrase rule's rarity penalty we choose the min of the individual links, and the max of the translation turtles.

As outlined in Chapter 3, the feature set we define for our paraphrase grammars mirrors that of translation SCFGs. For every supported translation feature, Thrax implements a corresponding *pivoted feature* for paraphrases. The pivoted features are set up to be aware of the prerequisite translation features they are derived from. This allows Thrax to automatically detect the needed translation features and spawn the corresponding map-reduce passes to compute conditional probabilities before the pivoting stage takes place. In addition to features useful for translation, Thrax also implements a number of features geared towards text-to-text generation tasks such as sentence compression or text simplification.

The Hadoop-based implementation allows us to easily scale up paraphrase grammar extraction to larger corpora. Table 6.8 gives a few examples of large paraphrase grammars extracted from WMT training data. With appropriate pruning settings,

Source Bitext	Sentences	Words	Pruning	Rules
Fr – En	1.6M	45M	$p(e_1\|e_2), p(e_2\|e_1) > 0.001$	49M
{Da + Sv + Cs + De + Es + Fr} – En	9.5M	100M	$p(e_1\|e_2), p(e_2\|e_1) > 0.02$	31M
			$p(e_1\|e_2), p(e_2\|e_1) > 0.001$	91M

Table 6.8: Large paraphrase grammars extracted from EuroParl data using Thrax. The sentence and word counts refer to the English side of the bitexts used.

we are able to obtain paraphrase grammars estimated over bitexts with more than 100 million words. An additional benefit is that Hadoop-based toolkits like Thrax can be easily deployed on arbitrarily large on-demand clusters provided by platforms like Amazon Web Services.

6.4.1.2 Compact Map-Reduce Representations for Grammar Extraction

The Hadoop-based implementation of paraphrase and translation grammar extraction in Thrax enables us to process bilingual corpora on the order of 100 million words. Beyond that, however, we find that the extractions generate a volume of intermediate data that quickly grows to outpace our computing cluster's resources. This problem is mainly due to the toolkit's verbose, naively string-based data structures.

We therefore implemented a full overhaul of Thrax for its 2.0 release [Post et al., 2013]. The core of Thrax 2.0 is a newly optimized data representation. In a first distributed map-reduce pass over the bitext, we collect a vocabulary that then enables us

Rules	Cs-En 112M		Fr-En 357M		De-En 202M		Es-En 380M	
	Space	Time	Space	Time	Space	Time	Space	Time
Thrax 1.0	120GB	112 min	364GB	369 min	211GB	203 min	413GB	397 min
Thrax 2.0	31GB	25 min	101GB	81 min	56GB	44 min	108GB	84 min
Difference	-74.1%	-77.7%	-72.3%	-78.0%	-73.5%	-78.3%	-73.8%	-78.8%

Table 6.9: Comparing Hadoop's intermediate disk space use and extraction time on a selection of Europarl v.7 Hiero grammar extractions. Disk space was measured at its maximum, at the input of Thrax's final grammar aggregation stage. Runtime was measured on our Hadoop cluster with a capacity of 52 mappers and 26 reducers. On average Thrax 2.0, bundled with Joshua 5.0, is up to 300% faster and more compact.

to encode the extracted rules and features in compact integer arrays. Replacing the data structure underpinnings of Thrax's map-reduce framework resulted in greater compactness and speed, yielding a 300% increase in extraction speed and an equivalent reduction in disk I/O. Table 6.9 shows time and memory use for translation grammar extractions with Thrax 2.0 at differing pruning levels.

The gains achieved with this optimization enable us to extract a syntactically labeled German-English SAMT-style translation grammar from a bitext of over 4 million sentence pairs in just over three hours. Furthermore, Thrax 2.0 is capable of scaling to very large data sets, like the composite bitext used in the extraction of our paraphrase collection PPDB [Ganitkevitch et al., 2013], which counts 106 million sentence pairs and over 2 billion words on the English side.

Further additions to Thrax 2.0 include a module focused on the extraction of compact distributional signatures over large datasets. This *distributional* mode collects contextual features for n-gram phrases, such as words occurring in a window around the phrase, as well as dependency-based and syntactic features. We then compute a bit signature from the resulting feature vector via a randomized locality-sensitive hashing projection. This yields a compact representation of a phrase's typical context. To perform this projection Thrax relies on the Jerboa toolkit [Van Durme, 2012]. As part of the PPDB effort, Thrax has been used to extract rich distributional signatures for 175 million 1-to-4-gram phrases from the Annotated Gigaword corpus [**?**], a parsed and processed version of the English Gigaword, Fifth Edition [Parker et al., 2011].

Thrax is distributed with Joshua and is also available as a separate download.[8]

6.4.2 A Memory-Efficient SCFG Representation Using Packed Tries

Both statistical machine translation systems, and paraphrasers tend to perform better when the system is trained on larger amounts of bilingual parallel data. Using tools such as Thrax, synchronous context-free grammars (SCFGs) are extracted and their parameters estimated from the data.

[8] `github.com/joshua-decoder/thrax`

To recapitulate, an SCFG is a collection of rules $\{\mathbf{r}_i\}$ that take the form:

$$\mathbf{r}_i = C_i \rightarrow \langle \alpha_i, \gamma_i, \sim_i, \vec{\varphi}_i \rangle, \tag{6.4}$$

where *left-hand side* C_i is a nonterminal symbol, the *source side* α_i and the *target side* γ_i are sequences of both nonterminal and terminal symbols. Further, $\sim_i$ is a one-to-one correspondence between the nonterminal symbols of α_i and γ_i, and $\vec{\varphi}_i$ is a vector of features quantifying the probability of α_i translating to γ_i, as well as other characteristics of the rule [Weese et al., 2011]. At decoding time, typical decoders like Joshua load the grammar rules into memory in their entirety, and store them in a trie data structure indexed by the rules' source side. This allows the decoder to efficiently look up rules that are applicable to a particular span of the (partially translated) input.

As the size of the training corpus grows, so does the resulting translation grammar. Using more diverse sets of nonterminal labels – which can significantly improve translation performance – further aggravates this problem. As a consequence, the space requirements for storing the grammar in memory during decoding quickly grow impractical. In some cases grammars may become too large to fit into the memory on a single machine.

As an alternative to the commonly used trie structures based on hash maps, we propose a packed trie representation for SCFGs. The approach we take is similar to work on efficiently storing large phrase tables by Zens and Ney [2007], and language models by Heafield [2011] as well as Pauls and Klein [2011].

6.4.2.1 Packed Synchronous Tries

For our grammar representation, we break the SCFG up into three distinct structures. As Figure 6.7 indicates, we store the grammar rules' source sides $\{\alpha_i\}$, target sides $\{\gamma_i\}$, and feature data $\{\vec{\varphi}_i\}$ in separate formats of their own. Each of the structures is packed into a flat array, and can thus be quickly read into memory. All terminal and nonterminal symbols in the grammar are mapped to integer symbol identifiers using a globally accessible vocabulary map. We will now describe the implementation details for each representation and their interactions in turn.

6.4.2.1.1 Source-Side Trie

The source-side trie (or source trie) is designed to facilitate efficient lookup of grammar rules by source side, and to allow us to completely specify a matching set of rule with a single integer index into the trie. We store the source sides $\{\alpha_i\}$ of a grammar in a downward-linking trie, i.e. each trie node maintains a record of its children. The trie is packed into an array of 32-bit integers. Figure 6.7 illustrates the composition of a node in the source-side trie. All information regarding the node is stored in a contiguous block of integers, and decomposes into two parts: a *linking block* and a *rule block*.

The linking block stores the links to the child trie nodes. It consists of an integer n, the number of children, and n blocks of two integers each, containing the symbol id a_j leading to the child and the child node's address s_j (as an index into the source-side

array). The children in the link block are sorted by symbol id, allowing for a lookup via binary or interpolation search.

The rule block stores all information necessary to reconstruct the rules that share the source side that led to the current source trie node. It stores the number of rules, m, and then a tuple of three integers for each of the m rules: we store the symbol id of the left-hand side, an index into the target-side trie and a *data block id*. The rules in the data block are initially in an arbitrary order, but are sorted by application cost upon loading.

6.4.2.1.2 Target-Side Trie

The target-side trie (or target trie) is designed to enable us to uniquely identify a target side γ_i with a single pointer into the trie, as well as to exploit redundancies in the target side string. Like the source trie, it is stored as an array of integers. However, the target trie is a *reversed*, or upward-linking trie: a trie node retains a link to its parent, as well as the symbol id labeling said link.

As illustrated in Figure 6.7, the target trie is accessed by reading an array index from the source trie, pointing to a trie node at depth d. We then follow the parent links to the trie root, accumulating target side symbols g_j into a target side string g_1^d as we go along. In order to match this traversal, the target strings are entered into the trie in reverse order, i.e. last word first. In order to determine d from a pointer into the target trie, we maintain an offset table in which we keep track of where each

new trie level begins in the array. By first searching the offset table, we can determine d, and thus know how much space to allocate for the complete target side string.

To further benefit from the overlap there may be among the target sides in the grammar, we drop the nonterminal labels from the target string prior to inserting them into the trie. For richly labeled grammars, this collapses all lexically identical target sides that share the same nonterminal reordering behavior, but vary in nonterminal labels into a single path in the trie. Since the nonterminal labels are retained in the rules' source sides, we do not lose any information by doing this.

6.4.2.1.3 Features and Other Data

We designed the data format for the grammar rules' feature values to be easily extended to include other information that we may want to attach to a rule, such as word alignments, or locations of occurrences in the training data. In order to that, each rule $\mathbf{r}_i$ has a unique block id b_i associated with it. This block id identifies the information associated with the rule in every attached data store. All data stores are implemented as memory-mapped byte buffers that are only loaded into memory when actually requested by the decoder. The format for the feature data is detailed in the following.

The rules' feature values are stored as sparse features in contiguous blocks of variable length in a byte buffer. As shown in Figure 6.7, a lookup table is used to map the b_i to the index of the block in the buffer. Each block is structured as follows:

a single integer, n, for the number of features, followed by n feature entries. Each feature entry is led by an integer for the feature id f_j, and followed by a field of variable length for the feature value v_j. The size of the value is determined by the type of the feature. Joshua maintains a quantization configuration which maps each feature id to a type handler or *quantizer*. After reading a feature id from the byte buffer, we retrieve the responsible quantizer and use it to read the value from the byte buffer.

Joshua's packed grammar format supports Java's standard primitive types, as well as an 8-bit quantizer. We chose 8 bit as a compromise between compression, value decoding speed and translation performance [Federico and Bertoldi, 2006]. Our quantization approach follows Federico and Bertoldi [2006] and Heafield [2011] in partitioning the value histogram into 256 equal-sized buckets. We quantize by mapping each feature value onto the weighted average of its bucket. Joshua allows for an easily per-feature specification of type. Quantizers can be share statistics across multiple features with similar value distributions.

6.4.2.2 Decoding Performance

We assess the packed grammar representation's memory efficiency and impact on the decoding speed on the WMT12 French-English task. Table 6.10 shows a comparison of the memory needed to store our WMT12 French-English grammars at runtime. We can observe a substantial decrease in memory consumption for both

Grammar	Format	Memory
Hiero (43M rules)	Baseline	13.6G
	Packed	1.8G
Syntax (200M rules)	Baseline	99.5G
	Packed	9.8G
	Packed 8-bit	5.8G

Table 6.10: Decoding-time memory use for the packed grammar versus the standard grammar format. Even without lossy quantization the packed grammar representation yields significant savings in memory consumption. Adding 8-bit quantization for the real-valued features in the grammar reduces even large syntactic grammars to a manageable size.

Hiero-style grammars and the much larger syntactically annotated grammars. Even without any feature value quantization, the packed format achieves an 80% reduction in space requirements. Adding 8-bit quantization for the log-probability features yields even smaller grammar sizes, in this case a reduction of over 94%.

In order to avoid costly repeated retrievals of individual feature values of rules, we compute and cache the stateless application cost for each grammar rule at grammar loading time. This, alongside with a lazy approach to rule lookup allows us to largely avoid losses in decoding speed.

Figure shows a translation progress graph for the WMT12 French-English development set. We graph number of sentences translated over time – the first portion of the graph represents the time it takes to load the translation and language models.

Grammar	Peak Memory (GB)	Loading Time (s)	Decoding Time (s)	Total Time
PPDB Baseline	295.9	10,920	2081	13001
PPDB Packed	18.8	8	2170	2442
Difference	-93.6%	-99.9%	+4.3%	-81.2%

Table 6.11: Using PPDB:Eng to decode our sentence compression test set (1000 sentences). The packed representation provides a staggering improvement over the unpacked, in both memory use and time taken for the decoding run.

Both systems load a Hiero-style grammar with 43 million rules, and use 16 threads for parallel decoding. The initial loading time for the packed grammar representation is dramatically shorter than that for the baseline setup (a total of 176 seconds for loading and sorting the grammar, versus 1897 for the standard format). Even though decoding speed is slightly slower with the packed grammars (an average of 5.3 seconds per sentence versus 4.2 for the baseline), the effective translation speed is more than twice that of the baseline (1004 seconds to complete decoding the 2489 sentences, versus 2551 seconds with the standard setup).

We repeat a similar experiment with PPDB:Eng, using the entire, full-sized XXXL version to paraphrase a set of 1000 sentences. The PPDB grammar has over 169 million paraphrase rules. We decode with 4 concurrent threads, using the Joshua Decoder. Key figures of the comparison are summarized in Table 6.11. When used with in the standard text-based storage format and read into the hash-based trie, the model peaks at over 295GB of memory. This figure encompasses the memory used

for grammar storage, and up to 4 decoding charts. The loading of the grammar from local hard drive storage takes just over three hours. The packed version lazily loads near instantly at 8 seconds, and at peak usage takes up just 18.8 GB of RAM. This corresponds to a 93.6% reduction in memory use, and a 99.9% reduction in loading time.

The substantial differences are due to a number of factors that go beyond the more compact, memory-friendly nature of the packed trie structure. For one, the packed representation does not in fact load the entirety of the grammar into memory upon startup. Only the source and target tries are loaded fully, the feature data on the other hand is accessed only on-demand. In addition to that, Joshua 5.0 uses an amortized sorting strategy. Previously the decoder would traverse the entire grammar upon loading to compute rule cost and sort the rules at each trie node. Joshua 5.0 adopts a lazy approach to grammar sorting. Upon accessing a trie node, we check if it has been visited before. Only if it is untouched do we compute the rule cost and perform a sort. These two optimizations combined allow Joshua 5.0 to jump into text generation near instantaneously, when compared to the standard approach. Figure 6.10 illustrates this in a progress graph, with the packed PPDB:Eng curve rising almost immediately. Figure 6.9 visualizes the memory use over time for the baseline representation and the packed PPDB. Here, we can see the memory use slowly growing and plateauing as the decoder on-demand loads the relevant portions of the grammar.

6.5 The Multilingual Paraphrase Database

There have been several prior efforts to extract non-English paraphrases for use in natural language processing applications. For example, paraphrase tables across five different languages were extracted as a part of METEOR-NEXT, a multilingual extension of the METEOR metric for machine translation evaluation [Denkowski and Lavie, 2010]. Similarly, automatically extracted paraphrases in Arabic and Chinese have been used to improve English-Arabic [Denkowski et al., 2010] and Chinese-Japanese [Zhang and Yamamoto, 2002, 2005] machine translation systems. Other individualized efforts have sought to create paraphrase resources for single languages, like Mizukami et al. [2014]'s efforts to create a Japanese version of PPDB. While achieving good results, many of the paraphrase collections used in these efforts have remained unavailable to the community.

While most work in natural language processing is still focussed on English, efforts to extend NLP systems' coverage and functionality to other languages are quickly becoming ubiquitous. As with English-language systems, we strongly believe that the availability of paraphrase resources can significantly help with the development of multilingual NLP solutions. Paraphrase collections can easily provide a boost in coverage for the input and output of an NLP system. Beyond that, paraphrases can form a basis for more complex work on tasks like entailment recognition or natural language understanding. The multilingual release of PPDB aims to provide a first iteration of such a paraphrase resource.

6.5.1 Pivoting over English

Pivoting over foreign language phrases for paraphrase extraction is language-independent. We can apply the exact same procedure to extract German, French, or Arabic paraphrases. In fact, as we have seen on the example of PPDB:Eng, paraphrases need not be extracted from a single pivot language. They can be obtained from multiple bitexts where the language of interest is contained on one side of the parallel corpus. Thus, instead of extracting German paraphrases just by pivoting over English, we could extract additional paraphrases from a German-French or a German-Spanish bitext. Although it is easy to construct parallel corpora for all pairs of languages in the European Union using existing resources like the Europarl parallel corpus [Koehn, 2005] or the JRC corpus [Steinberger et al., 2006], we only pivot over English for this release of the multilingual PPDB.

The reason that we limit ourselves to pivoting over English, is that we extend the bilingual pivoting method to incorporate syntactic information. Abundant NLP resources, such as statistical parsers, are available for English. By using annotations from the English side of the bitext, we are able to create syntactic paraphrases for languages for which we do not have syntactic parsers.

Syntactic information can be incorporated into the paraphrase process in a variety of ways. Callison-Burch [2008] showed that constraining paraphrases to be the same syntactic type as the original phrase significantly improved their quality. Ganitkevitch et al. [2011] showed how paraphrases could be represented using synchronous context

free grammars (SCFGs).

We project the English syntax onto the foreign sentence via the automatic word alignments. The notion of projecting syntax across aligned bitexts has been explored for bootstrapping parsers [Hwa et al., 2005]. The method that we use to find the syntactic labels for the foreign phrases is described in Zollmann and Venugopal [2006] and Weese et al. [2011]. Only the English side of each parallel corpus needs to be parsed, which we do with the Berkeley Parser [Petrov et al., 2006].

As with the English PPDB, each of the paraphrase databases in our multilingual set comes in the form of a syntactically labeled SCFG. Again, we distinguish three types of paraphrases:

- **Lexical paraphrases** – single word paraphrases or synonyms.
- **Phrasal paraphrases** – multi-word paraphrases, including cases where a single word maps onto a multi-word paraphrase and many-to-many paraphrases.
- **Syntactic paraphrases** – paraphrase rules that contain a placeholder symbol. These allow any paraphrase that matches that syntactic types of placeholder symbol to be substituted into that site.

The labels for the left-hand sides of our paraphrase rules and for the nonterminals in the syntactic paraphrase pairs are obtained by projecting the English parses across the bilingual word alignments. Figure 6.11 illustrates this projection on the example of a phrasal paraphrase being generalized into a syntactic paraphrases. This is done

by replacing words and phrases that are themselves paraphrases with appropriate nonterminal symbols.

The syntactic paraphrases can be used in conjunction with our Joshua decoder [Post et al., 2013] for monolingual text-to-text (T2T) generation applications, like sentence compression [Ganitkevitch et al., 2011, Napoles et al., 2011b]. This opens up the possibilities of developing new natural language generation (NLG) applications for the languages in our PPDB release.

6.5.2 Resource Size

We extract significantly different numbers of paraphrases for each the 23 non-English languages. The number of paraphrases is roughly proportional to the size of the bitext that was used to extract the paraphrases for that language. Figure 6.12 sorts the languages in order of how many paraphrases we extract for them. Unsurprisingly, we observe a large difference in size between the French, Arabic, and Chinese paraphrase sets, and the others. This is due to the comparatively large bilingual corpora that we used for these three languages, versus the smaller bitexts that were available for the other languages. Table 6.12 gives a detailed breakdown of the number of each kind of paraphrases (lexical, phrasal, syntactic) that we have extracted for each language.

Language	Code	Number of Paraphrases			
		Lexical	Phrasal	Syntactic	Total
Arabic	Ara	119.7M	45.1M	20.1M	185.7M
Bulgarian	Bul	1.3M	1.4M	1.2M	3.9M
Czech	Ces	7.3M	2.7M	2.6	12.1M
German	Deu	7.9M	15.4M	4.9M	28.3M
Greek	Ell	5.4M	9.4M	7.4M	22.3M
Estonian	Est	7.9M	1.0M	0.4M	9.2M
English	Eng	7.6M	68.4M	93.6M	169.6M
Finnish	Fin	41.4M	4.9M	2.3M	48.6M
French	Fra	78.8M	254.2M	170.5M	503.5M
Hungarian	Hun	3.8M	1.3M	0.2M	5.3M
Italian	Ita	8.2M	17.9M	9.7M	35.8M
Lithuanian	Lit	8.7M	1.5M	0.8M	11.0M
Latvian	Lav	5.5M	1.4M	1.0M	7.9M
Dutch	Nld	6.1M	15.3M	4.5M	25.9M
Polish	Pol	6.5M	2.2M	1.4M	10.1M
Portuguese	Por	7.0M	17.0M	9.0M	33.0M
Romanian	Ron	1.5M	1.8M	1.1M	4.5M
Russian	Rus	81M	46M	16M	144.4M
Slovak	Slk	4.8M	1.8M	1.7M	8.2M
Slovenian	Slv	3.6M	1.6M	1.4M	6.7M
Spanish	Spa	33.1M	73.2M	55.3M	161.6M
Swedish	Swe	6.2M	10.3M	10.3M	26.8M
Urdu	Urd	32.6k	22.4k	1k	56.0k
Chinese	Zho	52.5M	46.0M	8.9M	107.4M

Table 6.12: An overview over the sizes of the multilingual PPDB. The number of extracted paraphrases varies by language, depending on the amount of data available as well as the languages morphological richness. The language names are coded following ISO 639-2.

6.5.3 Morphological Variants as Paraphrases

Many of the languages covered by our resource are more morphologically complex than English. Since we are using English pivot phrases and English syntactic labels, the pivoting approach tends to group a variety of morphological variants of a foreign word into the same paraphrase cluster. For example, French adjectives inflect for gender and number, but English adjectives do not. Therefore, the French words *grand*, *grande*, *grands* and *grandes* would all share the English translation *tall*, and would therefore all be grouped together as paraphrases of each other. It is unclear whether this grouping is desirable or not, and the answer may depend on the downstream task. It is clear that there are distinctions that are made in the French language that our paraphrasing method currently does not make. All of the morphological variants of *grand* are given the label *JJ*, since it is projected from the English parse. If we had a French parser that produced distinct labels for each of the morphological variants, they would be separated from each other.

This effect is also observable in verbs. Other languages often have more inflectional variation than English does. Whereas English verbs only distinguish between past versus present tense and 3rd person singular versus non-3rd person singular, other languages exhibit more forms. For instance, the English verb *go*, aligns to a variety of present forms of the French irregular verb *aller*. The high-ranking paraphrases of *vais*, the first person singular form of *aller*, are all other forms of the verb. These are shown in Table 6.13. Similar effects can be observed across other verb paraphrases,

Tag	Phrase	Paraphrases
VB	vais	va, vas, irai, vont, allons, ira, allez, irons
	vas	va, vont, allez, vais, allons, aller
	vont	vas, va, allons, allez, vais, aller
VBD	allais	allait, alliez, allaient, allions
VB	denke	denken, denkt

Table 6.13: Top paraphrases extracted for forms of the French *aller* and the German *denken*. The English part-of-speech label we use preserves the unifying morphological characteristic quite well: present tense forms of *aller* dominate the ranking for the VB label (which best corresponds with present tense usage in English). Similarly, imperfect forms are reliably captured for the past tense VBD tag.

both in French and other languages. The minimal distinction in the Penn Treebank tags between past tense verbs (VBD), base form verbs (VB), and present tense verbs (VBN/VBP) partitions the foreign verbs to some extent. But clearly there is a semantic distinction between verb forms that are marked for person and number which pivoting constrained by English syntactic labels fails to make.

In fact, the interaction between our bilingual pivoting method and the impoverished morphologic system of English opens up avenues for improving the quality of the multilingual paraphrases. Our method makes distinctions between paraphrases when they have different syntactic labels. This does a good job of separating out things that make a sense distinction based on part of speech (like *squash* which paraphrases as *racquetball* as a noun and *crush* as a verb). It also limits different paraphrases

based on which form the original phrase takes. For instance, *divide* can paraphrase as *fracture* or *split* in both noun and verb forms, but it can only paraphrase as *gap* when the original phrase is a noun. Currently we use Penn Treebank tags, which are rather English-centric. This tag set could be replaced or refined to make finer-grained distinctions that are present in the foreign language. Refined, language-specific tag sets would do a better job at partitioning paraphrase sets that should be distinct.

Overall, the multilingual PPDB is a first-effort resource that breaks new ground in NLP. For most of the languages covered, PPDB is the first freely available paraphrase resource, providing a boost to NLP efforts in less common languages. Simultaneously, it shows up a number of interesting avenues for improvement in coverage, syntactic granularity, and semantic precision.

6.6 Conclusion

In this chapter we presented the 1.0 release of the paraphrase database PPDB, a large-scale collection of syntactically labeled paraphrases in English and 23 more languages. We have illustrated the resource's utility with an analysis of its coverage of Propbank predicates and relations. Our results suggest that PPDB will be useful in a variety of NLP applications.

We also elaborated on the engineering-heavy improvements to our extraction and decoding toolkit that were necessary to create and handle a resource the size of

PPDB. Both in extraction and in decoding we presented multi-fold gains in memory use and computation time, significantly improving over contemporary state-of-the-art techniques in the field.

One focal point for future releases of PPDB may be expanding the paraphrase collection's coverage with regard to both data size and languages supported. For one, larger parallel corpora can be included in the extraction, taking advantage of improved bitext mining techniques [Smith et al., 2013]. Especially for the multilingual release incorporating other pivot languages in addition to English could yield improvements in coverage. These expansions have the potential to significantly improve the coverage and quality of our paraphrase sets, especially for the lower-resource languages.

Furthermore, as we have shown in Chapter 5, paraphrase scoring can be improved by incorporating additional sources of information. Since large amounts of monolingual data are readily available, we expect a significant improvement in paraphrase quality by re-ranking our non-English paraphrases, especially for languages for which we only have small amounts of bitexts, such as Bulgarian, Romanian, or Urdu. PPDB could also be improved by incorporating second-degree information present in the data, like domain or topic signals. Additional points of refinement may be a better handling of phrase ambiguity, and accounting for the reliability of, and effects specific to, individual pivot languages.

Finally, PPDB offers a starting point for the exploration and extension towards aspects of related large-scale resources such as lexical-semantic hierarchies [Snow et al.,

2006], textual inference rules [Berant et al., 2011], relational patterns [Nakashole et al., 2012], and (lexical) conceptual networks [Navigli and Ponzetto, 2012]. Expanding from a plain collection of paraphrases to a more precisely semantically annotated resource would further increase the usefulness and importance of PPDB for natural language processing. Our aim is for PPDB to be a continuously updated and improving resource.

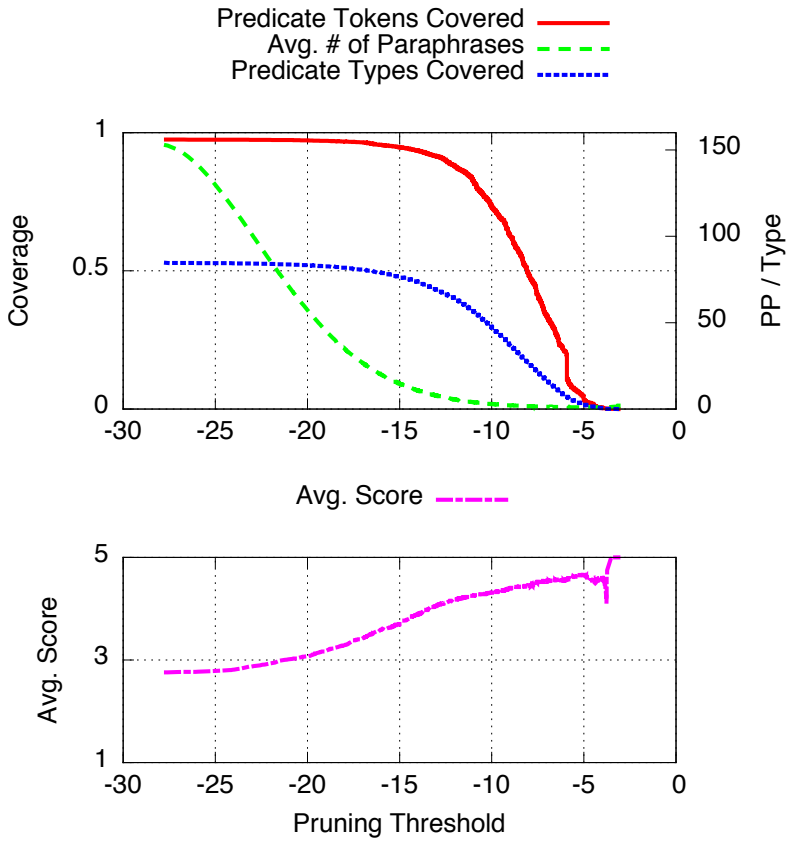

Figure 6.3: An illustration of PPDB:Eng's coverage of the manually annotated Propbank predicate phrases (top), and the average human judgment score (bottom) for varying pruning thresholds. The solid curve indicates the coverage on tokens, i.e. it shows what proportion of the predicates occurring in the Propbank corpus we can paraphrase. We also show coverage on types (dotted line), reflecting the proportion of the distinct predicates paraphrased, regardless of number of occurrences. The dashed line shows the average number of paraphrases per covered type at the given pruning level.

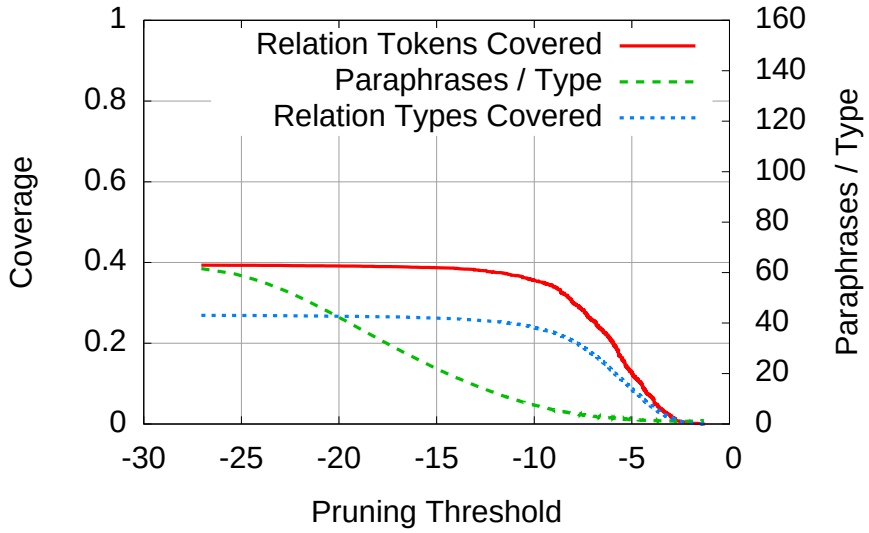

Figure 6.4: PPDB:Eng's coverage of full Propbank relations with up to two arguments. Here we consider rules that paraphrase the entire predicate-argument expression, matching the syntactic labels for both the entire predicate span, as well as those of each argument. The solid curve indicates the coverage on tokens, i.e. it shows what proportion of the relations occurring in the Propbank corpus we can paraphrase. We also show coverage on relation types (dotted line), reflecting the proportion of the distinct predicate-argument expressions paraphrased, regardless of its number of occurrences. The dashed line shows the average number of paraphrases per covered relation type at the given pruning level.

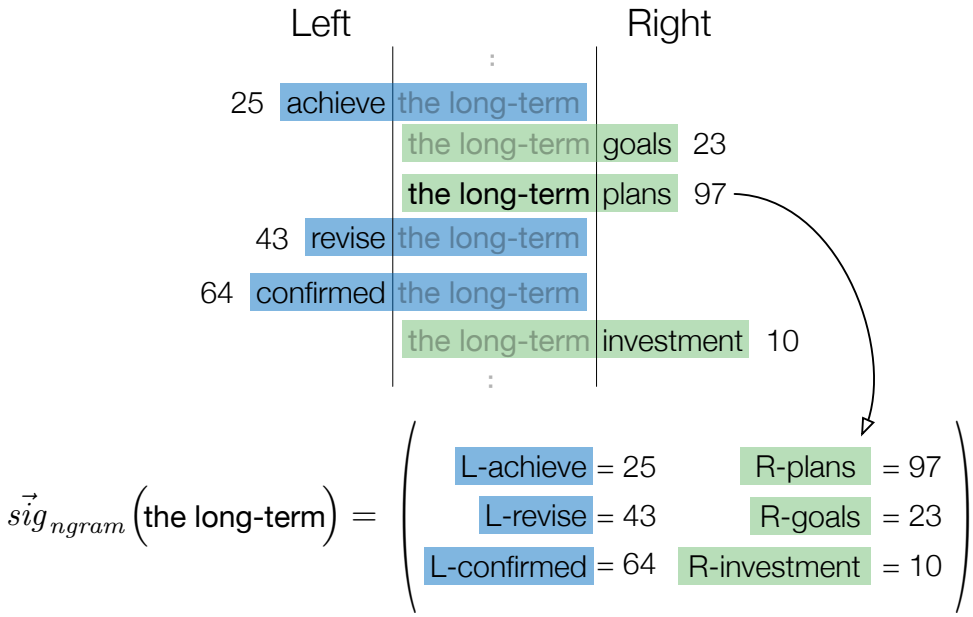

Figure 6.5: Features extracted for the phrase *the long term* from the n-gram corpus. The n-gram corpus records *the long-term* as preceded by *revise* (43 times), and followed by *plans* (97 times). We add corresponding features to the phrase's distributional signature retaining the counts of the original n-grams.

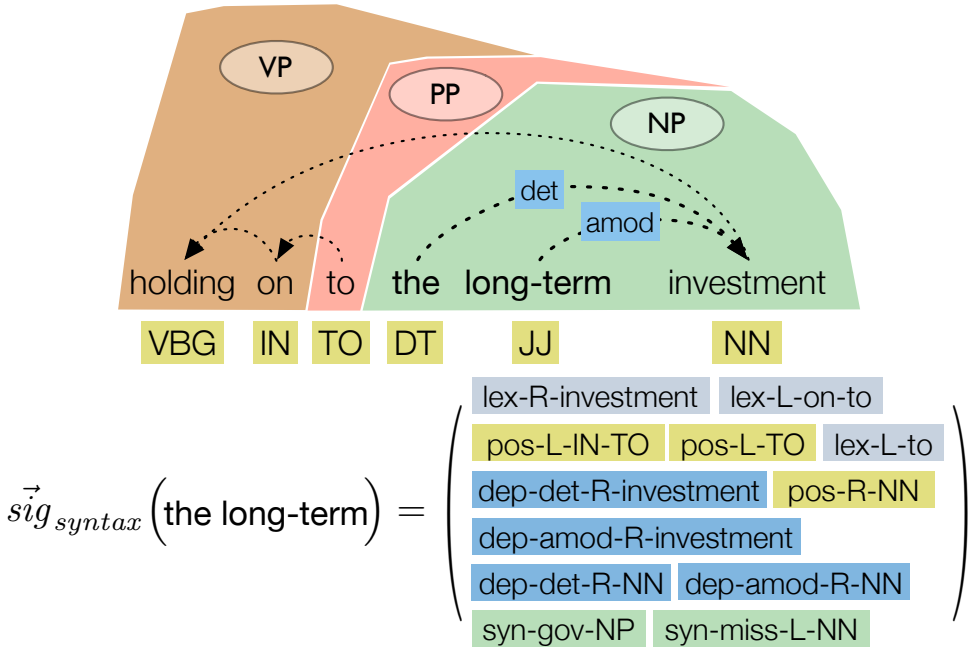

Figure 6.6: Features extracted for the phrase *the long term* from Annotated Gigaword. Here, position-aware lexical and part-of-speech n-gram features, labeled dependency links , and features reflecting the phrase's CCG-style label NP/NN are included in the context vector.

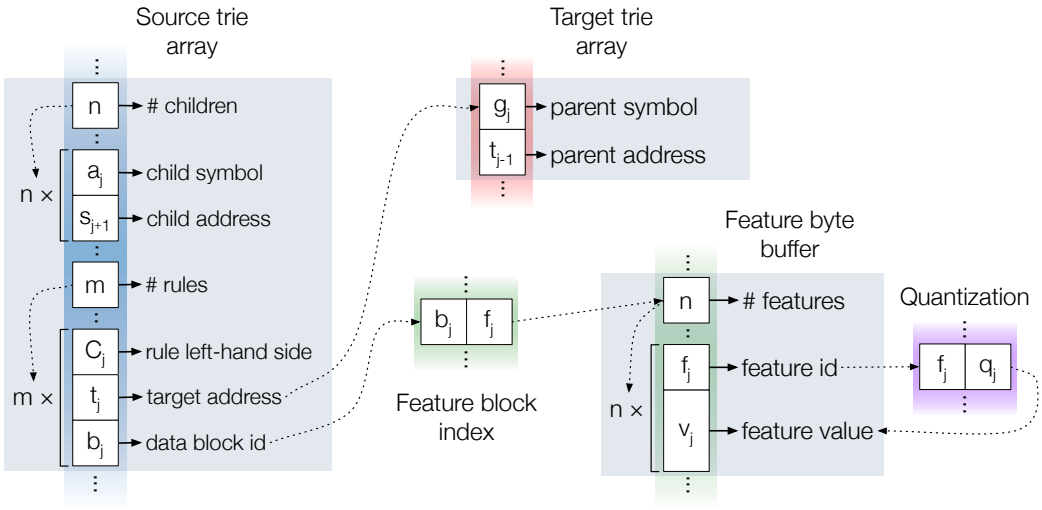

Figure 6.7: An illustration of our packed grammar data structures. The source sides of the grammar rules are stored in a packed trie. Each node may contain n children and the symbols linking to them, and m entries for rules that share the same source side. Each rule entry links to a node in the target-side trie, where the full target string can be retrieved by walking up the trie until the root is reached. The rule entries also contain a data block id, which identifies feature data attached to the rule. The features are encoded according to a type/quantization specification and stored as variable-length blocks of data in a byte buffer.

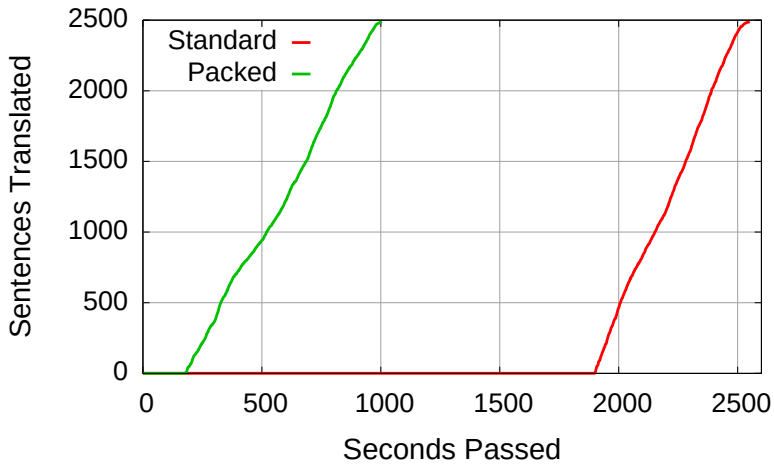

Figure 6.8: A visualization of the loading and decoding speed on the WMT12 French-English development set contrasting the packed grammar representation with the standard format. Grammar loading for the packed grammar representation is substantially faster than that for the baseline setup. Even with a slightly slower decoding speed (note the difference in the slopes) the packed grammar finishes in less than half the time, compared to the standard format.

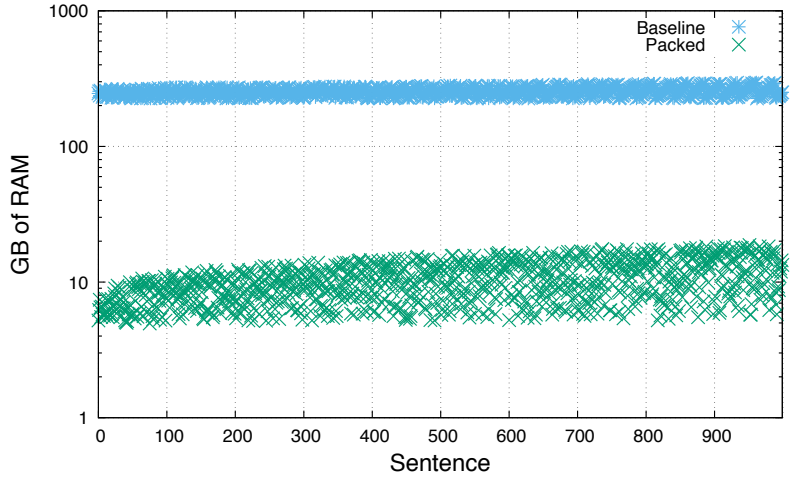

Figure 6.9: Memory use when decoding with the PPDB:Eng XXXL, on four threads. The packed version uses over an order of magnitude less memory.

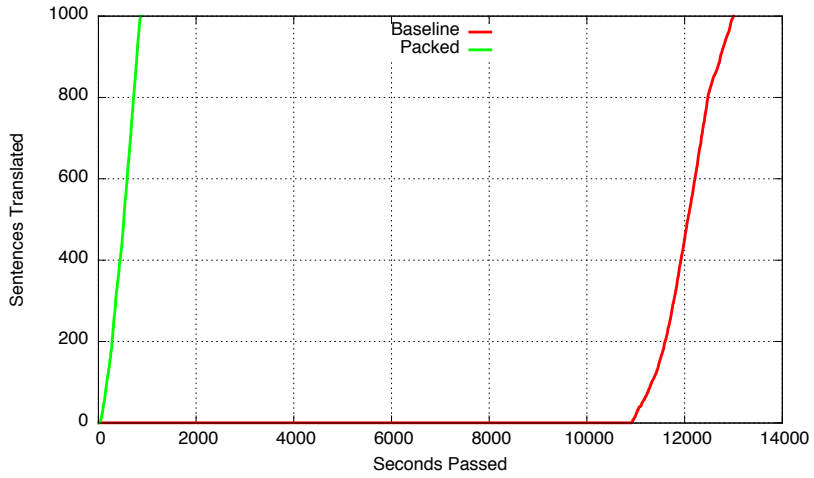

Figure 6.10: Decoding progress when decoding with the PPDB:Eng XXXL, on four threads. The baseline grammar takes a long time to load, while the packed version of PPDB is available near-instantly.

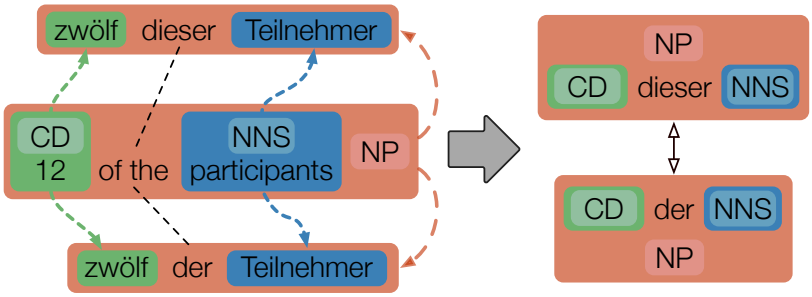

Figure 6.11: In addition to extracting lexical and phrasal paraphrases, we also extract syntactic paraphrases. These have nonterminal symbols that act as slots that can be filled by other paraphrases that match that syntactic type. The syntactic labels are drawn from parse trees of the English sentences in our bitexts.

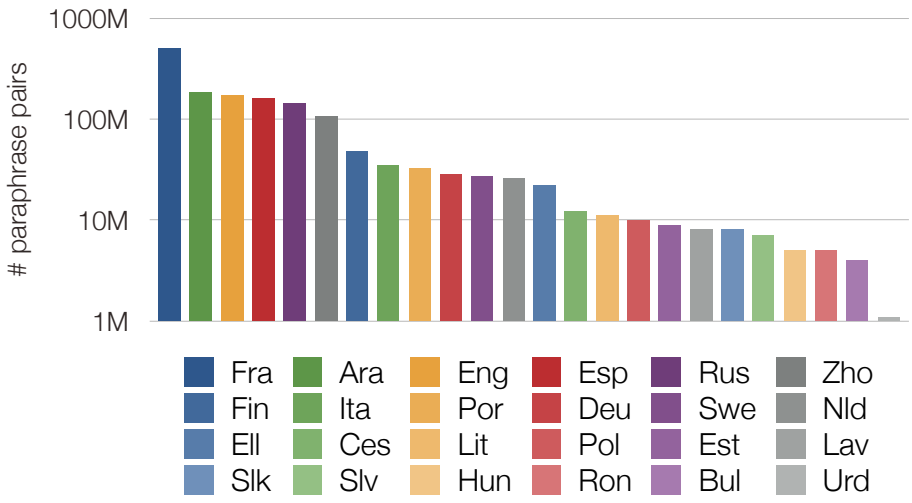

Figure 6.12: A visualization of the paraphrase collection size per language, measured in millions of paraphrase pairs. Languages are in order of PPDB size.

Chapter 7

Conclusion

A key thesis of this work is that high-quality paraphrases can be derived, at scale, from commodity data when combined with scalable automatic annotation. We furthermore stated that, when made readily available, as well as easy and computationally cheap to use, paraphrase resources can enable and drive progress in NLP work.

7.1 Summary of Contributions

In Chapter 3 we have shown that using widely available bilingual parallel corpora and open-source toolkits for parsing and alignment we can extract rich paraphrases. We build on the notion that two English phrases that translate to a shard foreign phrase likely mean the same thing. We expand that notion by incorporating linguistic

generalization our method uses word alignment techniques from machine translation to project syntactic annotations from English into a foreign language. By pivoting over foreign language phrases with matching projected syntax, we can extract syntactically annotated English paraphrases from bilingual data that have better precision and generalization capabilities than previous methods. We show that the paraphrase expressions our method yields can capture a variety of linguistically well-motivated paraphrastic rewrites. Our results indicate that it is possible to extract high-quality monolingual resources from relatively noisy commodity data sources like bilingual parallel texts and automated annotation.

Chapter 4 demonstrated that our syntactic paraphrases can be successfully applied to text-to-text rewriting tasks, in our case text compression. We use a light-weight adaptation scheme that extends statistical machine translation machinery to learn to produce task-targeted sentential paraphrase rewrites. To this end, we derive a small sample of development data reflecting the goal of our task, enrich our paraphrases with simple task-appropriate features, and adjust the objective function in our parameter tuning step. Our results show that the resulting system produces results comparable to custom-built contemporary text compression systems. This suggests that targeted sentential rewriting build on high-quality paraphrase resources can provide a viable approach to a wide variety of NLP problems.

In Chapter 5 we expanded our paraphrase quality estimation approach to include a new signal source: monolingual text. We investigate the incorporation of monolingual

distributional similarity signal into our pivot-based paraphrase extraction scheme. Experiments with paraphrastic text-to-text generation tasks show that the added distributional features do indeed carry orthogonal signal and improve our system's output quality. Comparing different distributional signals we find that even when extracted over a (relatively) smaller corpus, similarity signals help more when they are built on richer, linguistically motivated distributional feature sets based on automated text annotation. We can conclude that combining different resource types and large-scale text annotation can further improve the paraphrase quality of our approach.

We condensed these insights in the efforts presented in Chapter 6, scaling up our paraphrase extraction approach to generate and make available the paraphrase database PPDB. PPDB is the largest English paraphrase resource to date, counting over 170 million paraphrase expressions. It was published to be used as a general-purposed paraphrase resource, a basis for paraphrastic text rewriting work, as well as a foundation for next-generation work in paraphrase extraction. Our work also included a scaled-up toolkit for the extraction of paraphrase collections of this scale, and their application to text-to-text generation. The existence of this machinery made it easy to scale and adjust our paraphrase extraction scheme to other languages by pivoting over English. This enabled the extension of of PPDB to 23 different languages.

7.1.1 Subsequent Work

As we compile this thesis several years after the publication of its research contributions, we have an opportunity to gauge the work's impact on and contribution to the field of NLP. The paraphrase database PPDB, has remained quite widely cited in NLP, even as the field at large has moved away from the statistical methods and model we used to generate and apply PPDB, and on to neural models relying more on embeddings and network architecture.

For instance, a number of contributions have focussed on using PPDB to improve word embeddings trained for large corpora, sharpening them towards representations better suited for tasks like synonymy detection [Wieting et al., 2015, Faruqui et al., 2014, Yu and Dredze, 2014]. Other efforts have leveraged PPDB as a black-box source of features for semantic parsing, entailment recognition, or monolingual alignment [Wang et al., 2015, Bjerva et al., 2014, Yao et al., 2013].

7.2 Discussion and Future Directions

In this thesis we laid out a study at-scale paraphrase extraction from a variety of data sources, and made the case for the use of paraphrases as a general-purpose resource in NLP. Since the publication of the work underlying this thesis, the core models driving current NLP research have shifted towards neural approaches. Yet, our focus on data-centric, scalable paraphrase extraction, as well as the notion that

it is helpful to encapsulate the variety and nuance of natural language in paraphrase collections remain highly relevant.

In fact, the direction we took in applying our paraphrases to text-to-text generation tasks bears strong similarities to state-of-the-art neural NLP. Oftentimes, a neural NLP system will structurally separate the language grounding (pre-existing embeddings) from the task-specific architectures and learning. This closely mirrors our adaptive approach to use general-purpose paraphrases for a specific task.

The prevalence of embeddings-based approaches to NLP opens up a swath of interesting research directions. Encoding the semantics underlying an expression as a point in a high-dimensional space instead of as a set of pair-wise relations to other expressions allows for a vastly more scalable and flexible representation of paraphrase resources. There exists a substantial body of work on word embeddings, both mono- and multi-lingual. Similarly, an increasing amount of interest is devoted to the extraction of meaningful phrasal embeddings. We believe that this work can be extended to reflect the insights garnered from this thesis: that using commodity linguistic annotations over large datasets can improve the quality of extracted embeddings, especially in the case of finding meaningful encoding multiword-phrases.

Of further interest to us is the representation of syntactic paraphrase rules, multiword expressions with slots that are syntactically (or otherwise meaningfully) labeled. Frequently these expressions act not just as meaning-bearing portions of a text, but also modulate the meaning of their arguments. We believe that it would be worthwhile

to investigate a representation that takes advantage of the structured, grammatical nature of language while retaining the expressive power of high-dimensional tensor representations.

Generally, we believe that rebuilding a dataset like the paraphrase database as an embeddings corpus would both an interesting, ongoing research problem and of tremendous use for the NLP community.

Bibliography

Alfred V. Aho and Jeffrey D. Ullman. *The Theory of Parsing, Translation, and Compiling*. Prentice Hall, 1972.

Peter G. Anick and Suresh Tipirneni. The paraphrase search assistant: terminological feedback for iterative information seeking. In *Proceedings of SIGIR*, 1999.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.

Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *Proceedings of ACL*, 2005.

Regina Barzilay. *Information Fusion for Multidocument Summarization: Paraphrasing and Generation*. PhD thesis, Columbia University, New York, 2003.

Regina Barzilay and Lillian Lee. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *Proceedings of HLT/NAACL*, 2003.

BIBLIOGRAPHY

Regina Barzilay and Kathleen McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of ACL*, 2001.

Regina Barzilay, Kathleen R. McKeown, and Michael Elhadad. Information fusion in the context of multi-document summarization. In *Proceedings of ACL*, 1999.

Jonathan Berant. *Global Learning of Textual Entailment Graphs*. PhD thesis, Tel Aviv University, 2012.

Jonathan Berant, Jacob Goldberger, and Ido Dagan. Global learning of typed entailment rules. In *Proceedings of ACL*, 2011.

Jonathan Berant, Ido Dagan, Meni Adler, and Jacob Goldberger. Efficient tree-based approximation for entailment graph learning. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 117–125. Association for Computational Linguistics, 2012.

Rahul Bhagat and Eduard Hovy. What is a paraphrase? *Computational Linguistics*, 39(3):463–472, March 2013.

Rahul Bhagat and Deepak Ravichandran. Large scale acquisition of paraphrases for learning surface patterns. In *Proceedings of ACL/HLT*, 2008.

Steven Bird and Edward Loper. Nltk: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 31. Association for Computational Linguistics, 2004.

BIBLIOGRAPHY

Johannes Bjerva, Johan Bos, Rob Van der Goot, and Malvina Nissim. The meaning factory: Formal semantics for recognizing textual entailment and determining semantic similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 642–646, 2014.

Thorsten Brants and Alex Franz. The google web 1t 5-gram corpus version 1.1. ldc2006t13, 2006.

Peter Brown, John Cocke, Stephen Della Pietra, Vincent Della Pietra, Frederick Jelinek, Robert Mercer, and Paul Poossin. A statistical approach to language translation. *Computational Linguistics*, 16(2), June 1990.

Peter Brown, Stephen Della Pietra, Vincent Della Pietra, and Robert Mercer. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, June 1993.

Chris Callison-Burch. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of EMNLP*, 2008.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of WMT*, pages 1–28, Athens, Greece, March 2009.

Tsz Ping Chan, Chris Callison-Burch, and Benjamin Van Durme. Reranking bilingually extracted paraphrases using monolingual distributional similarity. In *Pro-*

ceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics, pages 33–42, Edinburgh, UK, July 2011. Association for Computational Linguistics.

Moses Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of STOC.*, 2002.

David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*, 2005.

David Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33 (2):201–228, 2007.

Kenneth Church and Patrick Hanks. Word association norms, mutual information and lexicography. *Computational Linguistics*, 6(1):22–29, 1991.

James Clarke and Mirella Lapata. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:273–381, 2008.

Trevor Cohn and Mirella Lapata. Large margin synchronous generation and its application to sentence compression. In *Proceedings of EMNLP-CoLing*, 2007.

Trevor Cohn and Mirella Lapata. Sentence compression beyond word deletion. In *Proceedings of the COLING*, 2008.

BIBLIOGRAPHY

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Michael Denkowski and Alon Lavie. METEOR-NEXT and the METEOR paraphrase tables: Improved evaluation support for five target languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, WMT '10, pages 339–342, 2010. ISBN 978-1-932432-71-8.

Michael Denkowski, Hassan Al-Haj, and Alon Lavie. Turker-assisted paraphrasing for english-arabic machine translation. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, CSLDAMT '10, pages 66–70, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Mona Talat Diab. *Word Sense Disambiguation Within a Multilingual Framework*. PhD thesis, College Park, MD, USA, 2003. AAI3115805.

Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the COLING*, 2004.

BIBLIOGRAPHY

Mark Dras. Reluctant paraphrase: Textual restructuring under an optimization model. In *PACLING-97*, pages 184–191, 1997.

Mark Dras. *Tree Adjoining Grammar and the Reluctant Paraphrasing of Text*. PhD thesis, Macquarie University, Australia, 1999.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of ACL*, 2010.

Andreas Eisele and Yu Chen. MultiUN: A multilingual corpus from united nation documents. In *Proceedings of LREC*, Valletta, Malta, 2010.

Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1535–1545. Association for Computational Linguistics, 2011.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*, 2014.

Marcello Federico and Nicola Bertoldi. How many bits are needed to store proba-

bilities for phrase-based translation? In *Proceedings of WMT06*, pages 94–101. Association for Computational Linguistics, 2006.

Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

William Gale and Kenneth Church. A program for aligning sentences in bilingual corpora. *Compuatational Linguistics*, 19(1):75–90, 1993.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *Proceedings of HLT/NAACL*, 2004.

Juri Ganitkevitch and Chris Callison-Burch. The multilingual paraphrase database. In *Proceedings of LREC*, Reykjavik, May 2014. Association for Computational Linguistics.

Juri Ganitkevitch, Chris Callison-Burch, Courtney Napoles, and Benjamin Van Durme. Learning sentential paraphrases from bilingual parallel corpora for text-to-text generation. In *Proceedings of EMNLP*, 2011.

Juri Ganitkevitch, Yuan Cao, Jonathan Weese, Matt Post, and Chris Callison-Burch. Joshua 4.0: Packing, PRO, and paraphrases. In *Proceedings of WMT12*, 2012a.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Monolingual distributional similarity for text-to-text generation. In *Proceedings of *SEM*. Association for Computational Linguistics, 2012b.

BIBLIOGRAPHY

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

Dmitriy Genzel, Jakob Uszkoreit, and Franz Och. Poetic statistical machine translation: rhyme and meter. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 158–166. Association for Computational Linguistics, 2010.

Erica Greene, Tugba Bodrumlu, and Kevin Knight. Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of EMNLP*, 2010.

Kenneth Heafield. KenLM: Faster and smaller language model queries. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 187–197. Association for Computational Linguistics, 2011.

Mark Hopkins and Jonathan May. Tuning as ranking. In *Proceedings of EMNLP*, 2011.

Liang Huang and David Chiang. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of ACL*, 2007.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. Boot-

strapping parsers via syntactic projection across parallel texts. *Natural language engineering*, 11(3):311–325, 2005.

Ali Ibrahim, Boris Katz, and Jimmy Lin. Extracting structural paraphrases from aligned monolingual corpora. In *Proceedings of the Second International Workshop on Paraphrasing (ACL 2003)*, 2003.

Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of STOC*, 1998.

David Kauchak and Regina Barzilay. Paraphrasing for automatic evaluation. In *Proceedings of EMNLP*, 2006.

Paul Kingsbury and Martha Palmer. From treebank to propbank. In *Proceedings of LREC*, 2002.

Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139:91–107, 2002.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit*, volume 5, 2005.

Philipp Koehn. *Statistical Machine Translation*. Cambridge University Press, 2010.

Philipp Koehn and Josh Schroeder. Experiments in domain adaptation for statistical

machine translation. In *Proceedings of WMT*, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of HLT/NAACL*, 2003.

Raymond Kozlowski, Kathleen McCoy, and K. Vijay-Shanker. Generation of single-sentence paraphrases from predicate/argument structure using lexico-grammatical resources. In *Workshop On Paraphrasing*, 2003.

I. Langkilde and K. Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 704–710. Association for Computational Linguistics, 1998.

Mirella Lapata and Frank Keller. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1), 2005.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of WMT09*, 2009.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Ann Irvine, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Ziyuan Wang, Jonathan Weese, and

BIBLIOGRAPHY

Omar Zaidan. Joshua 2.0: A toolkit for parsing-based machine translation with syntax, semirings, discriminative training and other goodies. In *Proceedings of WMT10*, 2010.

Percy Liang, Ben Taskar, and Dan Klein. Alignment by agreement. In *Proceedings of HLT/NAACL*, 2006.

Dekang Lin and Patrick Pantel. Discovery of inference rules from text. *Natural Language Engineering*, 2001.

Dekang Lin, Kenneth Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, and Sushant Narsale. New tools for web-scale n-grams. In *Proceedings of LREC*, 2010.

Jimmy Lin and Chris Dyer. Data-intensive text processing with mapreduce. *Synthesis Lectures on Human Language Technologies*, 3(1):1–177, 2010.

Linguistic Data Consortium. Assessment of fluency and adequacy in arabic-english and chinese-english translations. Specification, 2002.

Yang Liu, Qun Liu, and Shouxun Lin. Tree-to-string alignment templates for statistical machine translation. In *Proceedings of the ACL/Coling*, 2006.

Bill MacCartney. *Natural language inference*. Stanford University, 2009.

BIBLIOGRAPHY

Nitin Madnani and Bonnie Dorr. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–388, 2010.

Nitin Madnani, Necip Fazil Ayan, Philip Resnik, and Bonnie Dorr. Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of WMT07*, 2007.

Prodromos Malakasiotis and Ion Androutsopoulos. A generate and rank approach to sentence paraphrasing. In *Proceedings of EMNLP*, pages 96–106, 2011. URL `http://www.aclweb.org/anthology/D11-1009`.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: the Penn Treebank. *Computational Linguistics*, 19(2), 1993.

Ryan McDonald. Discriminative sentence compression with soft syntactic evidence. In *Proceedings of EACL*, 2006.

Kathleen R. McKeown. Paraphrasing using given and new information in a question-answer system. In *Proceedings of ACL*, 1979.

Dan Melamed. Statistical machine translation by parsing. In *Proceedings of ACL*, 2004.

Marie W. Meteer and Varda Shaked. Strategies for effective paraphrasing. In *Proceedings of COLING*, 1988.

BIBLIOGRAPHY

Masahiro Mizukami, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Creation of a Japanese paraphrase database and its application to linguistic individuality transformation. In *Proceedings of the 20th Annual Meeting of the Association for Natural Language Processing (NLP2014)*, pages 773–776, Hokkaido, Japan, March 2014. This paper is written in Japanese.

Kazunori Muraki. On a semantic model for multi-lingual paraphrasing. In *Proceedings of COLING*, 1982.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. PATTY: a taxonomy of relational patterns with semantic types. In *Proceedings of EMNLP*, 2012.

Courtney Napoles, Chris Callison-Burch, and Benjamin Van Durme. Evaluating sentence compression: Pitfalls and suggested remedies. In *Workshop on Monolingual Text-To-Text Generation*, 2011a.

Courtney Napoles, Chris Callison-Burch, Juri Ganitkevitch, and Benjamin Van Durme. Paraphrastic sentence compression with a character-based metric: Tightening without deletion. *Workshop on Monolingual Text-To-Text Generation*, 2011b.

Courtney Napoles, Matt Gormley, and Benjamin Van Durme. Annotated gigaword. In *Proceedings of AKBC-WEKEX 2012*, 2012.

Roberto Navigli and Simone Paolo Ponzetto. BabelNet: The automatic construc-

tion, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193, 2012.

Franz Josef Och. Minimum error rate training for statistical machine translation. In *Proceedings of ACL*, 2003.

Karolina Owczarzak, Declan Groves, Josef Van Genabith, and Andy Way. Contextual bitext-derived paraphrases in automatic MT evaluation. In *Proceedings of WMT06*, 2006.

Bo Pang, Kevin Knight, and Daniel Marcu. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of HLT/NAACL*, 2003.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, 2002.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword fifth edition. Number LDC2011T07. Linguistic Data Consortium, 2011.

Adam Pauls and Dan Klein. Faster and smaller n-gram language models. In *Proceedings of ACL*, pages 258–267, Portland, Oregon, USA, June 2011.

Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Benjamin Van Durme, and Chris Callison-Burch. Adding semantics to data-driven paraphrasing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*

and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), volume 1, pages 1512–1522, 2015a.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 425–430, 2015b.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July 2006. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P06/P06-1055`.

Matt Post, Juri Ganitkevitch, Luke Orland, Jonathan Weese, Yuan Cao, and Chris Callison-Burch. Joshua 5.0: Sparser, better, faster, server. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 206–212, Sofia, Bulgaria, August 2013. URL `http://www.aclweb.org/anthology/W13-2226`.

Chris Quirk, Chris Brockett, and William Dolan. Monolingual machine translation for paraphrase generation. In *Proceedings of EMNLP*, 2004.

BIBLIOGRAPHY

Chris Quirk, Arul Menezes, and Colin Cherry. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of ACL*, 2005.

Deepak Ravichandran and Eduard Hovy. Learning sufrace text patterns for a question answering system. In *Proceedings of ACL*, 2002.

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. Randomized Algorithms and NLP: Using Locality Sensitive Hash Functions for High Speed Noun Clustering. In *Proceedings of ACL*, 2005.

Stefan Riezler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal, and Yi Liu. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, 2007.

Hadar Shemtov. Generation of paraphrases from ambiguous logical forms. In *Proceedings of COLING*, 1996.

Stuart Shieber and Yves Schabes. Generation and synchronous tree-adjoining grammars. In *Workshop On Natural Language Generation*, 1990.

Jason Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. Dirt cheap web-scale parallel text from the Common Crawl. In *Proceedings of the 2013 Conference of the Association for Computational Linguistics (ACL 2013)*, Sofia, Bulgaria, July 2013. Association for

Computational Linguistics. URL `http://cis.upenn.edu/~ccb/publications/bitexts-from-common-crawl.pdf`.

Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. Ter-plus: paraphrase, semantic, and alignment enhancements to translation edit rate. *Machine Translation*, 23(2-3):117–127, 2010.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the ACL/Coling*, 2006.

Mark Steedman. Alternating quantifier scope in CCG. In *Proceedings of ACL*, 1999.

Mark Steedman and Jason Baldridge. Combinatory categorial grammar. In *Non-Transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell, 2011.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Dániel Varga. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of LREC*, Genoa, Italy, 2006.

Andreas Stolcke. SRILM - an extensible language modeling toolkit. In *Proceeding of the International Conference on Spoken Language Processing*, 2002.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

BIBLIOGRAPHY

Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP*, Proceedings of EMNLP, 2004.

Jörg Tiedemann. News from OPUS: A collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing*, volume 5, 2009.

Benjamin Van Durme. Jerboa: A toolkit for randomized and streaming algorithms. Technical Report 7, Human Language Technology Center of Excellence, Johns Hopkins University, 2012.

Benjamin Van Durme and Ashwin Lall. Online generation of locality sensitive hash signatures. In *Proceedings of ACL, Short Papers*, 2010.

Ashish Venugopal and Andreas Zollmann. Grammar based statistical MT on Hadoop: An end-to-end toolkit for large scale PSCFG based MT. *Prague Bulletin of Mathematical Linguistics*, 91, 2009.

Yushi Wang, Jonathan Berant, and Percy Liang. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1332–1342, 2015.

Jonathan Weese, Juri Ganitkevitch, Chris Callison-Burch, Matt Post, and Adam

Lopez. Joshua 3.0: Syntax-based machine translation with the Thrax grammar extractor. In *Proceedings of WMT11*, 2011.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*, 2015.

Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3), 1997.

Wei Xu, Alan Ritter, Chris Callison-Burch, William B. Dolan, and Yangfeng Ji. Extracting lexically divergent paraphrases from twitter. *Transactions of the Association for Computational Linguistics*, 2:435–448, 2014. ISSN 2307-387X. URL `https://transacl.org/ojs/index.php/tacl/article/view/498`.

Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proceedings of ACL*, 2001.

Kazuhide Yamamoto. Machine translation by interaction between paraphraser and transfer. In *Proceedings of COLING*, 2002.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. Semi-markov phrase-based monolingual alignment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 590–600, 2013.

Mo Yu and Mark Dredze. Improving lexical embeddings with semantic knowledge.

In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 545–550, 2014.

Omar F. Zaidan. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88, 2009.

Richard Zens and Hermann Ney. Efficient phrase-table representation for machine translation with applications to online MT and speech translation. In *Proceedings of HLT/NAACL*, pages 492–499, Rochester, New York, April 2007. Association for Computational Linguistics.

Congle Zhang and Daniel S Weld. Harvesting parallel news streams to generate paraphrases of event relations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1776–1786, 2013.

Congle Zhang, Stephen Soderland, and Daniel S Weld. Exploiting parallel news streams for unsupervised event extraction. *Transactions of the Association of Computational Linguistics*, 3(1):117–129, 2015.

Yujie Zhang and Kazuhide Yamamoto. Paraphrasing of chinese utterances. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

BIBLIOGRAPHY

Yujie Zhang and Kazuhide Yamamoto. Paraphrasing spoken chinese using a paraphrase corpus. *Nat. Lang. Eng.*, 11(4):417–434, December 2005. ISSN 1351-3249.

Shiqi Zhao, Cheng Niu, Ming Zhou, Ting Liu, and Sheng Li. Combining multiple resources to improve SMT-based paraphrasing model. In *Proceedings of ACL/HLT*, 2008a.

Shiqi Zhao, Haifeng Wang, Ting Liu, and Sheng Li. Pivot approach for extracting paraphrase patterns from bilingual corpora. In *Proceedings of ACL/HLT*, 2008b.

Liang Zhou, Chin-Yew Lin, Dragos Stefan Munteanu, and Eduard Hovy. ParaEval: Using paraphrases to evaluate summaries automatically. In *Proceedings of HLT/NAACL*, 2006.

Andreas Zollmann and Ashish Venugopal. Syntax augmented machine translation via chart parsing. In *Proceedings of WMT06*, 2006.