Low-Dimensional Context-Dependent Translation Models

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Avneesh S. Saluja

B.S., Electrical Engineering, Stanford University
M.S., Electrical and Computer Engineering, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

September, 2015

To my parents, who have encouraged me to do nothing but follow my dreams.

Acknowledgments

There are many people who, directly or indirectly, contributed to my graduate school experience, the culmination of which is this thesis. First and foremost, I would like to thank the members of my thesis committee. My advisor Chris Dyer took me under his wing relatively late in my graduate school career, but through him I have learned that research is far more than proving theorems, implementing models, and running experiments; without a compelling and motivating story, everything else is just details. His enthusiasm and optimism for all things NLP (and indeed, on subjects well beyond) is infectious, and has often given me hope and belief in ideas that I would otherwise have prematurely dismissed. Joy Zhang, who was my advisor for the first three years of graduate school, also deserves special mention for gently introducing me to the ways of research and formulating problems that are compelling from a purely academic perspective but at the same time have significant practical impact as well. Ian Lane has often acted as my second advisor throughout graduate school, and has been extremely supportive of my independent streak in selecting research problems. Hal Daumé has been the most flexible and accommodating external member in the history of thesis committees, and his deep insights into the problems discussed in this thesis were invaluable.

What I truly enjoyed about graduate school is that I consider my "colleagues" as friends first. In no particular order, and for arbitrarily stated reasons (chosen amongst many), I would like to thank: my co-authors, from whom I have probably learned more than any class I have taken in graduate school (Ankur P., Shay C., Branislav K.); not to say classes were not useful – I am especially indebted to Larry W. for teaching statistics like a `bau5` and Noah S. for teaching me how to look at problems in language processing with a machine learning lens; my colleagues from my internships at IBM Research (Jiří N., Yasser Al-O., Salim R.), Microsoft Research (Hany H., Kristina T., Chris Q., Arul M.), and eBay Applied Science (Jean-David R., Derek B., Mahesh J., Hassan S.), for allowing me to work on problems of my choosing while knowing the right questions to ask about them; my Carnegie Mellon-Silicon Valley friends (Rahul R., Aveek P., Senaka B., Akshay C., David C., Eric C., Frank M., Malte I., Vishwa R.), for the conversations, beers, and perspectives; my clab and Language Technology Institute friends (Waleed A., Yulia T., Jeff F., Manaal F., Wang L., Austin M., Swabha S.) for the great research discussions and guidance, and being reasonable and responsible (for the

most part!) on `allegro`; friends from the Machine Learning department (Ahmed H., Avinava D., Sashank R.) for their company and assistance on problem sets and research, and friends from the far-flung reaches of campus (Sudeep B., Nazli T.) for the mind-expanding discussions on philosophy, decision-making, and the economics of altruism; special thanks to Chase K. for letting me stay at his place whenever I visited Pittsburgh; and friends from outside Carnegie Mellon who kept me sane and aligned to the real world while I was knee-deep in the trenches (Ashfaque S., Basilio B., Aditya S., Eytan D.). Last but not least, special thanks to my family: my parents, brother, and sister-in-law for their constant support through thick and thin, and my nephews for being a source of joy and entertainment.

This research was supported by the Defense Advanced Research Projects Agency under contract D11PC20022, Intelligence Advanced Research Projects Activity via Department of Defense U.S. Army Research Laboratory under contract W911NF-12-C-0015, eBay Inc., and the John and Claire Bertucci Fellowship. Any opinions, findings, conclusions or recommendations expressed in this work are those of the author and do not necessarily reflect the view of the funding sources.

Abstract

Context matters when modeling language translation, but state-of-the-art approaches predominantly model these dependencies via larger translation units. This decision results in problems related to computational efficiency (runtime and memory) and statistical efficiency (millions of sentences, but billions of translation rules), and as a result such methods stop short of conditioning on extreme amounts of local context or global context.

This thesis takes a step back from the current zeitgeist and posits another view: while context influences translation, its influence is inherently low-dimensional, and problems of computational and statistical tractability can be solved by using dimensionality reduction and representation learning techniques. The low-dimensional representations we recover intuitively capture this observation, that the phenomena that drive translation are controlled by context residing in a more compact space than the lexical-based (word or n-gram) "one-hot" or count-based spaces.

We consider low-dimensional representations of context, recovered via a multi-view canonical correlations analysis, as well as low-dimensional representations of translation units that are expressed (featurized) in terms of context, recovered by a rank-reduced SVD of a feature space defined over inside and outside trees in a synchronous grammar. Lastly, we test our low-dimensional hypothesis in the limit, by considering a semi-supervised learning scenario where contextual information is gleaned from large amounts of unlabeled data. All empirical setups show improvements by taking into account the low-dimensional hypothesis, indicating that this route is an effective way to boost performance while maintaining model parsimony.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

> *"Always design a thing by considering it in its larger context – a chair in a room, a room in a house, a house in an environment, an environment in a city plan."*
>
> — Eliel Saarinen

Context matters when modeling translation. We have seen empirical evidence of this assertion in the transition from word- to phrase-based models, which achieve start-of-the-art performance by relying on long multi-word units or phrases to incorporate local context (Koehn et al., 2003, Chiang, 2007). More recently, neural and factored translation models have started to condition on large amounts of source-language context and have reported sizable gains in translation performance (Feng and Cohn, 2013, Devlin et al., 2014). Like the current zeitgeist, this thesis takes seriously the premise that modeling contextual dependencies in machine translation (MT) is key to effective translation. However, state-of-the-art approaches predominantly model these dependencies via larger translation units. Larger units mean larger models, resulting in problems in computational efficiency (runtime and memory) and statistical efficiency (how can we reliably learn parameters for billions of rules from millions of sentences?). Furthermore, we contend that translation models should be sensitive to far more than "local" context: larger contexts are observed during evaluation (e.g., the entire sentence to be translated or even the document containing the sentence), and this information should be used when translating since it is computationally inexpensive to do so and produces better translations. For example, a subject could be separated by a long subordinate clause and if we consider limited local context, the subject's influence on a subsequent verb phrase may be ignored. But if we cannot reliably estimate context-insensitive models or effectively decode with them, how should we hope to develop models that are sensitive to the rich input context available during test time? Clearly, another approach is necessary.

The thesis statement and central hypothesis of this work is that *while context influences trans-*

lation, its influence is inherently low-dimensional, and problems of computational and statistical tractability can be solved by using dimensionality reduction and representation learning techniques. To make the right translation choices for ambiguous words like *bank* or *watch*, it is unnecessary to embed the word in a long phrase as a means of incorporating context (by naive memorization of sequences); a small amount of information, e.g., the presence of a determiner before *watch*, or the knowledge that *bank* occurs amongst context that is semantically related to finance, suffices. The low-dimensional representations we recover intuitively capture this observation, that the phenomena that drive translation are controlled by context residing in a more compact space than the lexical-based (word or n-gram) "one-hot" or count-based spaces. Furthermore, they allow more reliable parameter estimates from less data in a more computationally efficient manner (Kakade and Foster, 2007).

In most previous approaches, global context adaptation is carried out on top of a massive phrase-based model. Due to the sheer number of phrases to consider, this decision often limits the extent to which context can be considered and for tractability reasons, context disambiguation often boils down to a lexical selection problem whereas ideally we should consider translation rules with multi-word units. Despite empirical evidence showing models with composed rules (ones that can be formed out of smaller rules) outperform their minimal counterparts by weakening independence assumptions in the translation model (Koehn et al., 2003, Galley et al., 2006), we propose to work with minimal translation rules as our basic units. The reasoning is that if we have better translation models that take into account context more effectively than including limited amounts within translation rules, we can eliminate the gigantic grammars that phrase-based translation has relied on and work with smaller, simpler models that generalize better to new corpora. Thus, a secondary objective of this thesis is to explore the performance of minimal grammars in conjunction with low-dimensional context-dependent models, and compare them to composed grammars which incorporate local context dependence within rules.

An outline of the structure of the thesis is now provided.

1.1 Dissertation Outline

In this thesis, we consider low-dimensional representations of context, as well as low-dimensional representations of translation units that are expressed (featurized) in terms of context. In Chapter 2, we introduce the fundamental concepts behind the thesis. Specifically, we discuss two assumptions that our low-dimensional hypothesis hinges on, the multi-view and manifold assumptions, and describe how they are utilized in this thesis when applied to linguistic context. Furthermore, we also provide a brief primer on machine translation, with an emphasis on the relevant parts for this thesis. In particular, the distinction between minimal and composed grammars is stressed, motivating the need to incorporate larger forms of context

through auxiliary models. The ideas presented in Ch. 2 are used throughout the remainder of the thesis, and we will refer back to them as necessary. Then, Chapters 3 ,4, and 5 explore three instantiations of the low-dimensional assumption:

1. **Low-dimensional embeddings of translation units** (Ch. 3): we propose a latent-variable model for synchronous context-free grammars (SCFGs) and apply it to hierarchical phrase-based translation (HPBT). The non-terminals in each rule are augmented with latent states in a context-dependent manner, which we learn from a parallel corpus. Specifically, non-terminals are refined by expressing them in terms of low-dimensional projections of the context in which they occur. In this case, we project a high-dimensional representation of a translation rule, represented with the empirical covariances of inside and outside tree features in synchronous trees where the rule occurs as a non-terminal, into a low-rank space.

2. **Low-dimensional embeddings of context** (Ch. 4): we adopt the multi-view assumption (Foster et al., 2008, Dhillon et al., 2011), which states that we can use two complementary views of the data (e.g., the left and right contexts a source phrase occurs in) to recover a low-dimensional basis via canonical correlation analysis (CCA). Supervised learning can then proceed in this low-rank space but with reduced sample complexity. The intuition is to utilize information in both views which correlate well with each other. In this case, we project a high-dimensional representation of context defined in terms of lexical and syntactic features into a low-dimensional space using CCA. We model phrasal choice and translation sense disambiguation by conditioning on low-dimensional representations of source context in this manner.

3. **Low-dimensional embeddings and semi-supervised learning** (Ch. 5): by looking at a setting where large amounts of context can be gleaned in an unsupervised manner and combined with translation information from parallel corpora, we test our low-dimensional hypothesis in semi-supervised scenarios. Specifically, we consider a nearest-neighbor approach to translation where a phrase pair is embedded in a graph, the source side a node with edges to its most similar phrases, and the target side as a "label" for the node. The presence and strengths of these edges is determined by the context in which the phrases occur, and is computed on monolingual corpora. Translation information can propagate through the graph either enabling the discovery of new phrases and their translations, or in a domain adaptation setting where we use in-domain monolingual data to embed translation units. The framework allows an interesting empirical evaluation of context dimensionality in an MT setting, since the graphs can be constructed in a number of different ways, either in the raw high-dimensional space or in a recovered low-dimensional one.

Figure 1.1 is a visual representation of the three instantiations, and emphasizes the common elements among them; both structured and unstructured inputs are considered through the lenses of the multi-view and manifold assumptions for representation learning purposes, and

this information is integrated into a standard MT setup through a number of ways. Lastly, Chapter 6 discusses several proposals for extending the work presented in this thesis and draws general conclusions on the topic, including potential downsides with our framework.

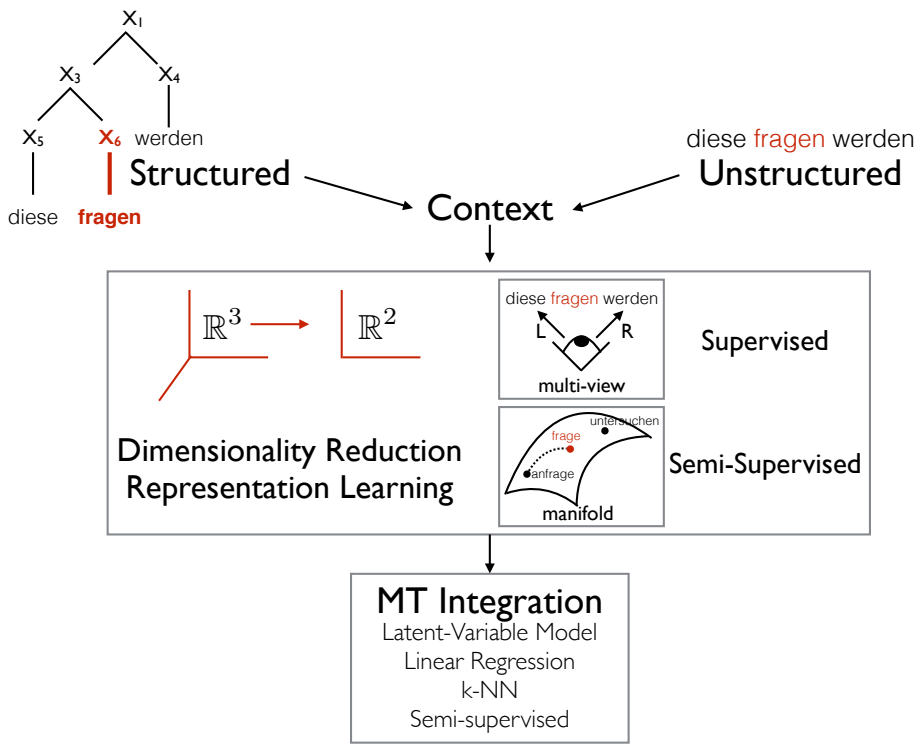


Figure 1.1: Pictorial representation of the three instantiations of the low-dimensional hypothesis that we explore.

1.2 Research Contributions

This thesis makes several novel contributions, which are broadly divided into two categories: Foundations and Applications.

1.2.1 Foundations

- A generalization of the latent PCFGs formalization (Matsuzaki et al., 2005) to latent SCFGs. Latent SCFGs are the bilingual generalization of latent PCFGs, just as SCFGs

are the bilingual generalization of PCFGs. The formalization is quite general, and a number of inference and parameter estimation algorithms can be applied.

- A novel tensor-based version of the inside-outside algorithm that operates on a hypergraph representation of the parse forest for an input sentence. The algorithm is a straightforward generalization of the regular inside-outside algorithm, except scalar multiplications are replaced by tensor dot products.
- Two algorithms that learn these latent categories (equivalently, the latent space) from the data without any externally imposed syntactic labels. The first is a likelihood maximization approach that is a variant of EM, and the second is a novel spectral estimation algorithm based on the method of moments, which is computationally faster than the EM-based algorithm, and under certain assumptions yields the globally optimal solution.
- Two classes of methods to jointly estimate low-dimensional context and translation rule representations. The first is a linear estimation technique that computes a CCA between different views of the context (a generalization of the spectral estimation algorithm for latent SCFGs), after which supervised learning is conducted in the low-dimensional context space. The second is a non-linear method that is a generalization of the skip-gram model used to estimate word representations, but for bilingual phrase pairs.
- A graph-based SSL method to expand translation models with information contained in monolingual data via contextual similarity. There are two forms of context dimensionality reduction here; the first is the graph embedding itself, which leverages the low-dimensional manifold structure of the data, and the second is a dimensionality reduction in the ambient space, which theoretically should make manifold estimation easier.

1.2.2 Applications

- An empirical demonstration that adding marginal rule probabilities from a latent SCFG model as features in the traditional linear translation model (Och and Ney, 2004) improves translation quality, and release of the entire source code as the `spectral-scfg` package.
- A thorough experimental comparison of phrase sense disambiguation models, both high-dimensional and low-dimensional, in isolated and end-to-end MT settings, and release of the entire source code as the `cca-mt` package.
- An exploration of the transition from minimal to composed grammars, not through the conventional methods of using larger translation rules, but instead through auxiliary low-rank models.
- Experimental evidence in different language pairs that expanding translation models

using graph embeddings results in significant improvements in translation quality, and release of the entire source code as the `graphMT` package.

Chapter 2

Background

"He who would learn to fly one day must first learn to stand and walk and run and climb and dance; one cannot fly into flying."

— Friedrich Nietzsche

The contents of this chapter are broadly divided into two sections: translation models (§2.1), which reviews the necessary statistical machine translation (SMT)[1] background and contextualizes the problem of low-dimensional context-dependence in translation models; and the parts on the two primary assumptions we utilize in our low-dimensional hypothesis: the multi-view assumption (§2.2) and the manifold assumption (§2.3). We avoid entering into a full-blown MT primer, for which there are excellent resources (Lopez, 2008, Koehn, 2010) that are recommended if the reader is more interested in the subject.

First, a brief discussion on notation. For strings, x refers to a single lexeme or unit (depending on the level of granularity, x could be a character, morpheme, word, or phrase), and $\boldsymbol{x}$ refers to an entire sentence (consisting of at least one unit). In the context of linear algebra, symbolically we only distinguish between scalars x, vectors $\mathbf{x}$ (both row and column), and general-order tensors X, which include both matrices (second-order tensors) and higher-order tensors. Dot products between two vectors $\mathbf{x}$ and $\mathbf{y}$ can be explicitly represented using $\mathbf{x} \cdot \mathbf{y}$, otherwise matrix-matrix and matrix-vector products are represented with adjacent symbols i.e., XY or $X\mathbf{y}$. For products between higher-order tensors and vectors, we use the notation in Kolda and Bader (2009), where the mode of the tensor is specified; see §3.2.2 for additional details, when tensor notation becomes relevant.

[1] We will often alternate between the acronyms SMT and MT to refer to machine translation; almost exclusively, the predominant approach is statistical.

2.1 Translation Models

The goal of MT is to translate a sentence in a source language, denoted $\boldsymbol{f}$, to a sentence in a target language, denoted $\boldsymbol{e}$.[2] The sequences of translation decisions used to go from source to target sentence, which we refer to as a **derivation**, is denoted with $\boldsymbol{d}$. Note that there can be many derivations that transform the same source sentence $\boldsymbol{f}$ into the same target sentence $\boldsymbol{e}$. As such, the distribution we are truly interested in modeling is $P(\boldsymbol{e}|\boldsymbol{f})$, which can be computed by summing over all derivations:

$$P(\boldsymbol{e}|\boldsymbol{f}) = \sum_{\{\boldsymbol{d}|(\boldsymbol{d},\boldsymbol{f})\rightarrow\boldsymbol{e}\}} P(\boldsymbol{e},\boldsymbol{d}|\boldsymbol{f})$$

where $\{\boldsymbol{d}|(\boldsymbol{d},\boldsymbol{f}) \rightarrow \boldsymbol{e}\}$ is the set of all derivations that transform a source sentence $\boldsymbol{f}$ into the target sentence $\boldsymbol{e}$. Thus, the derivation is treated as a latent variable that must be marginalized. Unfortunately, computing this sum involves exponential complexity (in the source sentence length) for the prevalent translation model formalisms (Brown et al., 1993, Melamed, 2004) and is in fact NP-complete, so most approaches sidestep the issue by treating derivations as translations.[3] The best translation (derivation) conditional on the source sentence is thus:

$$\arg\max_{\boldsymbol{e},\boldsymbol{d}} P(\boldsymbol{e},\boldsymbol{d}|\boldsymbol{f})$$

One way to parameterize the learning procedure of $P(\boldsymbol{e}|\boldsymbol{f})$ is to adopt a generative model. A generative model learns the joint probability distribution $P(\boldsymbol{f},\boldsymbol{d},\boldsymbol{e})$ and then uses Bayes' theorem:

$$\begin{aligned} P(\boldsymbol{e}|\boldsymbol{f}) &= \frac{P(\boldsymbol{f},\boldsymbol{d},\boldsymbol{e})}{P(\boldsymbol{f})} \\ &= \frac{P(\boldsymbol{f},\boldsymbol{d}|\boldsymbol{e})P(\boldsymbol{e})}{P(\boldsymbol{f})} \end{aligned} \tag{2.1}$$

During decoding, which is the application of parameters learned in training to translate new sentences, the denominator is irrelevant i.e., computing an arg max only depends on the numerator, which consists of two terms: the **translation model** ($P(\boldsymbol{f},\boldsymbol{d}|\boldsymbol{e})$, TM) and the **language**

[2] The conventional use of $\boldsymbol{f}$ for the source sentence and $\boldsymbol{e}$ for the target sentence (or in general, strings in the source and target languages) stems from the original experiments that were applied on the Hansard **F**rench-**E**nglish Canadian parliamentary proceedings.

[3] Blunsom and colleagues presented a noticeable exception to this choice, by actually summing over derivations in the process of learning $P(\boldsymbol{e}|\boldsymbol{f})$ directly (i.e., a discriminative model). However, in Blunsom et al. (2008b), the authors crucially do not incorporate language model information, and although it is considered in Blunsom and Osborne (2008), the authors have to resort to an approximate sampling procedure to make the approach tractable.

model ($P(\boldsymbol{e})$, LM).[4] The TM has several responsibilities: re-order words in the source sentence into an appropriate target order (if necessary), translate the words or word sequences from the source to the target language, and assign a weight or a score to each translation hypothesis in the output. Thus, the TM consists of a phrasal inventory or a model of translational equivalence (Lopez, 2008), and a model to score the numerous translation hypotheses. The LM is a model of the target language, and re-ranks hypotheses produced by the TM by scoring on the basis of target language fluency. Since our focus in this section is translation models, we do not discuss LM learning and inference here.

The decomposition in Eq. 2.1 is also called the noisy channel model[5], and was the primary basis behind the introduction of *statistical* MT (Brown et al., 1990), after successes in its application in the speech recognition space (Bahl et al., 1983). Generative models have the advantage of applying two independent models when translating, which is beneficial if the errors in each model cancel each other out or if we have a lot more monolingual data (which the LM uses) than parallel data. Because of the strong conditional independence assumptions, empirically and theoretically generative models perform well in a "low data" regime (Ng and Jordan, 2002). In practice however, the classic noisy-channel model is not strictly followed; for example, it was found that using $P(\boldsymbol{e}|\boldsymbol{f})$ instead of $P(\boldsymbol{f}|\boldsymbol{e})$, while theoretically unfounded, worked empirically as well if not better (Och and Ney, 2002). This observation and others motivated the combination of previous generative model components (i.e., TM and LM) with several other information sources e.g., the forward probability $P(\boldsymbol{e}|\boldsymbol{f})$ to disambiguate the vast number of translation options available. The relative weights $\mathbf{w}$ of these components are learned in a discriminative fashion as a linear (or log-linear, the only difference being that a softmax function is applied to the arg max argument in Eq. 2.2) structured prediction problem:

$$\arg\max_{\boldsymbol{e},\boldsymbol{d}} \mathbf{w} \cdot \boldsymbol{\phi}(\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{d}) \tag{2.2}$$

where $\boldsymbol{\phi}(\cdot)$ is a feature vector defined over the source, target, and derivation structure. Here, we have rephrased the translation decision problem in a more general manner by removing the probabilistic interpretation (although variants, like the log-linear model, retain their probabilistic nature). The discriminative framework allows us to define numerous sparse or dense overlapping features in the form of the vector $\boldsymbol{\phi}(\cdot)$, unlike in the generative case where features had to abide by strict independence assumptions. A number of different objectives can be used to learn the weights in Eq. 2.2 e.g., minimum error rate training (MERT, Och, 2003) or large-margin criteria (Liang et al., 2006, Chiang et al., 2009, *inter alia*). For a more detailed overview of these algorithms, we refer the reader to Koehn (2010) for MERT and Chiang (2012) for various large-margin algorithms. In light of the shifting semantics with the move to discriminative models, we broaden our definition of a translation model beyond just the

[4] Note that in the generative framework, we model the reverse probability $P(\boldsymbol{f}|\boldsymbol{e})$.
[5] A term borrowed from information theory.

reverse probability model $P(\boldsymbol{f}|\boldsymbol{e})$ to now contain a general translational equivalence model that is able to score its translation options when conditioned on an input source sentence. In other words, it is the part of an MT system that learns parameters from a parallel corpus using a formalism that can generate strings in two different languages simultaneously.

The discussion above operated at the level of sentences. It would be infeasible to actually learn parameters at this level: a back-of-the-napkin calculation with realistic numbers suffices to show this case.[6] Hence, the actual TMs operate at a much finer granularity, usually at the level of words or small, multi-word sequences known as phrases.[7] The derivation structure $\boldsymbol{d}$ is thus the collection of translation decisions that yield the target sentence given the source, and the overall sentence probability is the product of the scores of the individual translation rules that make up $\boldsymbol{d}$. We first provide a brief overview of lexical models (§2.1.1), emphasizing the lack of translational context considered by these models, which provided the main motivation to move towards phrasal models (§2.1.2). However, phrasal models introduce their own host of issues, which segues into our discussion of minimal grammars (§2.1.3). The minimal grammars form the starting point for translation models in Chapters 3 and 4.

2.1.1 Lexical Models

In the beginning, researchers at IBM created a series of lexical translation models (Brown et al., 1990, 1993), which are collectively dubbed the **IBM models**. The first two of these models make use of the following general form[8]:

$$P(\boldsymbol{e}|\boldsymbol{f}) = P(m|\boldsymbol{f}) \sum_{\boldsymbol{a}\in[0,l]^m} P(\boldsymbol{a}|\boldsymbol{f}, m) \prod_{i=1}^{m} P(e_i|f_{a_i}) \tag{2.3}$$

Here, m is the length of the target sentence, l is the length of the source sentence, and $\boldsymbol{a}$ is the set of **alignment** variables, which in these models function as the derivation structures $\boldsymbol{d}$; just like $\boldsymbol{d}$, these variables are latent, in that we do not observe their values at all during training and they must be inferred. The alignments are random variables that, in Eq. 2.3, are attached to each target word $e_i, 1 < i < m$. Each alignment variable can take any value in the set $[0, l]$, specifying an alignment for each target word to some word in the source sentence, where the 0th position corresponds to the null alignment (i.e., a target word aligned to null is an unaligned word).

[6] Assuming an average sentence length of 30 words and a vocabulary size on the order of 10^5, sentence-level models would result in model sizes on the order of 10^{150} parameters (sentences), with only a tiny fraction of such sentences observed in corpora.

[7] This definition is at odds with the parsing community's, where the term is strictly entwined with phrase structure grammars. In MT, it simply means any contiguous (usually short) sequence of words.

[8] The traditional exposition is for $P(\boldsymbol{f}|\boldsymbol{e})$ from the noisy channel model, but we use $P(\boldsymbol{e}|\boldsymbol{f})$ instead.

Eq. 2.3 is an exact decomposition, but is not the only way we can decompose $P(\boldsymbol{e}|\boldsymbol{f})$. Rather, the choice of decomposition displays the generative process underlying our lexical model. In this case, first a length m is chosen for the target sentence, conditional on the source; then, we decide which positions in $\boldsymbol{f}$ are linked to every position $1 < i < m$ in the target. Subsequently, translation decisions $P(e_i|f_{a_i})$ are conditionally independent of each other given these alignments, and each word in $\boldsymbol{e}$ is generated from exactly one word in $\boldsymbol{f}$, as indicated by the alignment a_i. Utilizing these latent alignment variables (or more generally, latent derivation structures) allows us to make far more reasonable conditional assumptions than otherwise. Instead of assuming that translation decisions are unconditionally independent, we assume that, *conditioned on* the alignment information, these decisions are independent. To deal with the target length m, we assume that $P(m|\boldsymbol{f})$ is independent of both m and $\boldsymbol{f}$; hence, we can replace it with a small constant $\epsilon > 0$. With these assumptions, most kinds of alignments are allowed: word dropping, insertion (by aligning a target word to the null token), and one-to-many links, but many-to-one alignments are not.

IBM Model 1 (M1) makes additional assumptions to Eq. 2.3 in order to to make parameter estimation more tractable. Specifically, the m alignment decisions are independent of each other, and the alignment distribution for each alignment link is uniform over all source words and the null token. Mathematically:

$$P(\boldsymbol{a}|\boldsymbol{f}, m) = \frac{1}{(l+1)^m} \Rightarrow P(\boldsymbol{e}|\boldsymbol{f}) = \frac{\epsilon}{(l+1)^m} \sum_{\boldsymbol{a}\in[0,l]^m} \prod_{i=1}^{m} P(e_i|f_{a_i}) \tag{2.4}$$

While the expectation in Eq. 2.4 is intractable in general as there are $(l+1)^m$ possible alignments, the summation is over terms each of which is a monomial that contains m translation probabilities. Brown et al. (1993) show that we can write the result as:

$$\frac{\epsilon}{(l+1)^m} \sum_{\boldsymbol{a}\in[0,l]^m} \prod_{i=1}^{m} P(e_i|f_{a_i}) = \frac{\epsilon}{(l+1)^m} \prod_{i=1}^{m} \sum_{j=0}^{l} P(e_i|f_j) \tag{2.5}$$

As is standard with latent variable models (recall that the alignments are unobserved), expectation maximization (EM, Dempster et al., 1977) for maximizing empirical likelihood is used to learn the parameters. The basic idea is to use current (or initial) parameter estimates for the translation probabilities $P(e|f)$ to compute posterior alignment probabilities $P(a_i|\boldsymbol{e}, \boldsymbol{f})$, and then fix these posterior alignments in order to re-estimate the translation probabilities. Because of the assumptions made, under mild conditions the EM algorithm converges to the global optimum of the log-likelihood function of Eq. 2.4. IBM Model 2 (M2) is similar to M1, except the uniform alignment probability assumption is relaxed. Specifically, we assume that the probability of an individual alignment a_i is not uniform and depends on the position i and the target length m as well as the source length l as before. While we can use the same trick as in Eq. 2.5, EM no longer recovers the global optimum.

Subsequent IBM models are based on a different form of Eq. 2.3, where phenomena like word **fertility** (target words can be aligned to multiple source words at the same time, thereby allowing many-to-one alignments) and **distortion** (to handle re-ordering effects) are incorporated into the generative process. These elaborations come at a cost however: learning via EM is rife with local minima (although initializing parameters with simpler models like M1 and M2 help), and the trick in Eq. 2.5 no longer applies, which means sampling (over high probability alignments) is utilized in lieu of exhaustive computation. More importantly, none of these additions masks the fact that these models are fundamentally limited in their use of context. The translation distributions $P(e|f)$ do not take into account the words surrounding f at all, and this severe limitation was the main motivation for the move to phrasal models (§2.1.2). Nonetheless, alignment information (specifically, the Viterbi i.e., most likely alignment for a sentence pair) is very important, and most phrase extraction approaches for building phrasal inventories bootstrap from the word alignments. The lexical probabilities $P(e|f)$ and $P(f|e)$ computed over lexical items in translation rules are also fundamental features in practically all modern MT systems.

2.1.2 Phrasal Models

Phrasal models overcome the deficiencies of lexical models by using phrases as the basic unit in the model. By considering longer translation units, these models incorporate limited local context dependencies. For example, the German word *fragen* can be translated in both noun ("question") or verb ("to ask") form. If we instead considered two-word phrases as the basic translation unit, we may find instances of *diese fragen* ("these questions"). In this case, the presence of a determiner disambiguates the noun and verbal forms.

For a sentence $\boldsymbol{f}$ with length l, the number of possible phrasal segmentations is exponential in l, so how do we extract the "right" translation units from this sentence? Unfortunately, direct phrasal alignment is a very difficult task. Marcu and Wong (2002) first proposed a phrasal alignment model that was a generalization of the lexical models in §2.1.1, but because of the trade-offs and approximations involved, it was not practical. In his thesis (DeNero, 2010), DeNero showed a number of drawbacks with direct phrasal models. First, learning via EM tends to the degenerate solution, because of the existence of two latent variables: the segmentation of source and target sentences into phrases, and the alignment of these phrases across the two languages. While competing alignments cannot all be correct, competing segmentations can, which leads EM to prefer longer phrases during segmentation in an effort to maximize likelihood. Furthermore, inference (finding the most likely alignment) is NP-hard. Naturally, there are ways to bias EM away from preferring longer phrases (by imposing suitable priors) and restricted classes of phrase alignment models where polynomial-time inference procedures do exist, but the most practical and empirically successful method of extracting phrases has been to bootstrap from word alignments. Phrasal inventories are bootstrapped from word alignments in different ways depending on the formalization.

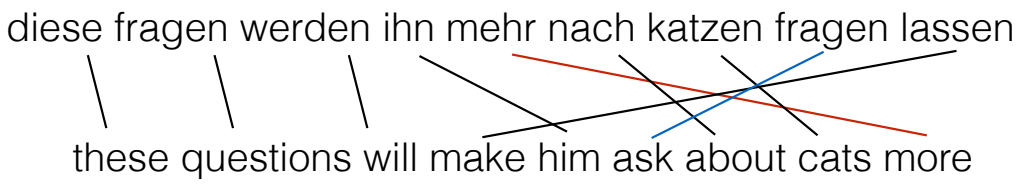


Figure 2.1: German-English sentence pair and word alignments. Overlapping alignments are colored differently for demonstration purposes.

In the following, we briefly review the three primary types of translation models and the phrase extraction process each type entails. The discussion on syntax-based translation is primarily to provide background on the distinction between minimal and composed grammars, but in this thesis, we utilize the phrase-based and hierarchical phrase-based translation formalisms only. The examples we give will be based on the German-English sentence pair and word alignments in Fig. 2.1.

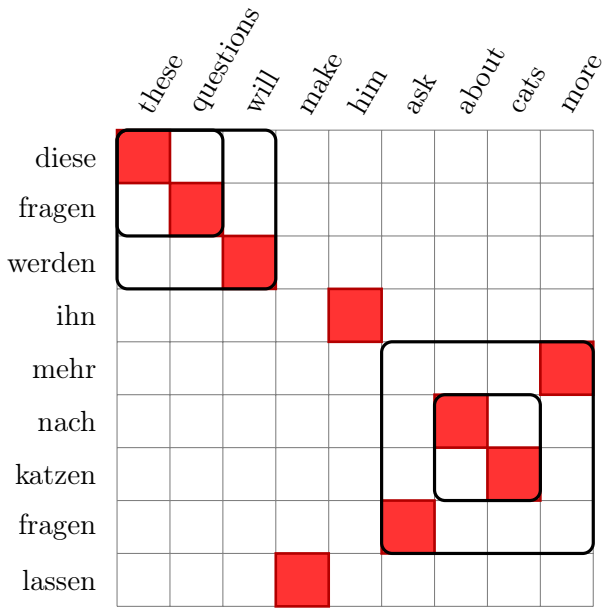


Figure 2.2: An alignment matrix for the sentence pair in Fig. 2.1. Examples of consistent phrase pairs are outlined in black.

Phrase-based Translation

Phrase-based translation (PBT, Koehn et al., 2003, Och and Ney, 2004) models are instantiations of weighted finite-state transducers (Lopez, 2008). Translational equivalence between two languages is modeled using pairs of phrases in the source and target language that have the same meaning. Using a heuristic, phrases are extracted from a sentence pair by maintaining *consistency* with the word alignments. Fig. 2.2 presents example word alignments for the sentence pair in Fig. 2.1, represented as an alignment matrix. Consistent phrase pairs are those that do not have alignment links directly adjacent to the box surrounding a phrase pair in the alignment matrix (diagonal adjacencies are fine). The black boxes that delineate several example phrase pairs in Fig. 2.2 are consistent phrase pairs. An example phrase pair is "*diese fragen* ||| these questions". Note that consistent phrase pairs can be nested (as the example phrase pair is), and during phrase extraction all nested pairs are extracted too. This decision results in an enormous number of phrase pairs extracted, and so limits are set on their length. Phrase-based translation also does not directly model reordering phenomena, and often a separate reordering model is employed for this purpose.

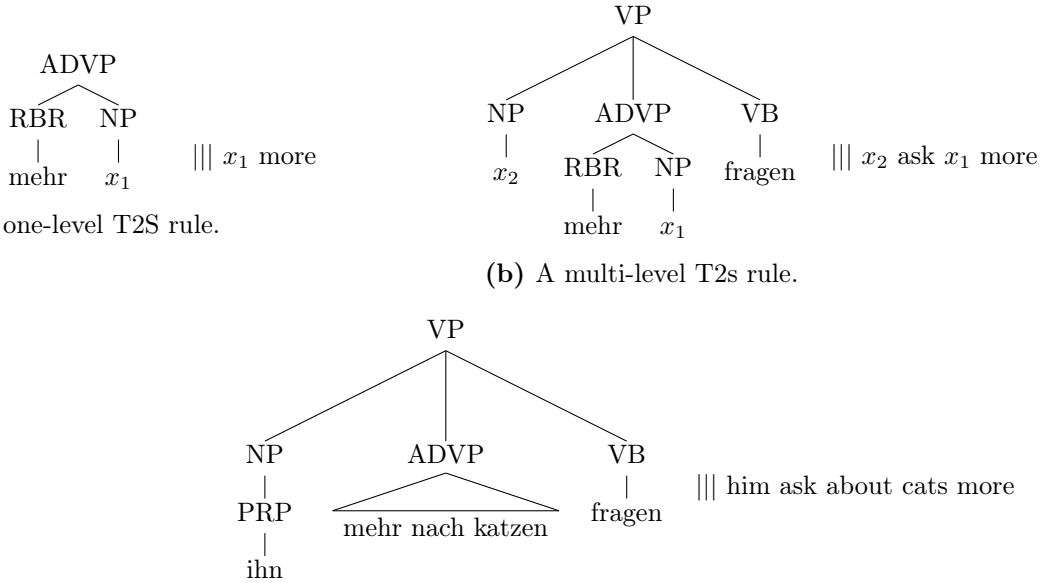
Syntax-based Translation

Syntax-based translation[9] (SBT, Galley et al., 2004, Zollmann and Venugopal, 2006) models are usually instantiations of weighted synchronous context-free grammars. Translational equivalence between two languages is modeled using a synchronous grammar that simultaneously generates source and target sentences and the correspondence between them. Local translation decisions are made and aggregated in a syntactically consistent way. There are a number of different approaches to SBT, depending on whether syntax is used only on the source side (tree-to-string, or T2S translation), only on the target side (string-to-tree, or S2T translation), or one both sides (tree-to-tree, or T2T translation). These approaches vary in terms of the basic translation units considered and the phrase extraction process. For this discussion, we restrict ourselves to T2S translation, specifically extended tree-to-string transducers (Graehl et al., 2008), where we have translation rules with tree fragments on the source side, and strings on the target side. Figure 2.3 gives several examples, which we explain below.

Galley et al. (2004) presented an elegant heuristic that extracts these kinds of syntax-based translation rules from word alignments and a parallel corpus. The basic idea is to attach the word-aligned target side as additional nodes in the source parse tree, by introducing edges between the leaf nodes (source words) in the tree and the target words. The result is an alignment graph. From the alignment graph, a set of frontier nodes are identified; these are nodes with the property that the span on the target side in the alignment graph is exclusive

[9] Some of this discussion presupposes a basic knowledge of syntactic parsing.

i.e., it does not intersect with spans from nodes other than parent and child nodes. From these frontier nodes, a set of minimal rules are extracted that have lexical (pre-terminal) and/or internal (non-terminal) nodes on the target side, and a tree fragment on the source-side. These rules are **minimal** in the sense that they cannot be decomposed into simpler rules induced by the same alignment graph. Both Galley et al. (2004) and Huang et al. (2006) stress the importance of multi-level rules i.e., tree fragments on the source side that also include tree fragments of parent nodes (see Figures 2.3a and 2.3b), to provide a narrow form of context dependence.



(a) A one-level T2S rule.

(b) A multi-level T2s rule.

(c) A composed rule, created by composing the one level-rule with its parent rule.

Figure 2.3: Examples of tree-to-string rules of the type used in Galley et al. (2004, 2006) and Huang et al. (2006). The part-of-speech tags may not be what a German parser would output, but are the English equivalents.

Follow-up work (Galley et al., 2006) found that in practice, "context-rich" syntactic models do better. The starting point of these models is a minimal explanation for a sentence pair, as in Galley et al. (2004); however, they subsequently "acquire larger rules that *crucially condition on more syntactic context*" (emphasis added). These larger rules are called **composed** rules, and they result from the composition of two or more minimal rules (Fig. 2.3c). Composed rules are different from multi-level rules in that we replace tree fragments on the source side that are denoted with **non-terminals** (NTs, variables that describe gaps in which other translation rules can fit) with their actual realizations from the training corpus. With these rules, there

are multiple synchronous trees consistent with the alignments for a given sentence pair, and thus the total number of applicable rules can be combinatorially larger than if we just consider the set of rules that cannot be formed from other rules, namely the minimal rules. The same reasoning also applies to the PBT models that consider nested, consistent phrase pairs. In SBT, the addition of composed rules in the grammar results in a dramatic improvement in translation quality, but comes at a cost: the grammar also increases more than 60 times in size!

While syntax-based models are appealing to the computational linguist in all of us and can model relatively long-range dependencies and reordering phenomena, there are numerous downsides which have precluded the widespread adoption of such models in MT. Firstly, a good parser is often a necessary prerequisite, lest parsing errors propagate into translation errors; good parsers tend to require lots of Treebank-style training data[10], which is hard to obtain for many of the world's languages. Furthermore, the introduction of non-terminal categories from syntax like NP and VP results in additional data sparseness issues, and more importantly, these category constraints may not be consequential or even make sense for the purposes of translation.

Hierarchical Phrase-based Translation

Hierarchical phrase-based translation (HPBT, Chiang, 2007), also known as "Hiero", occupies a middle-ground between the PBT and SBT schemes. While formally a context-free grammar, HPBT borrows the hierarchical structure from this formalism but does not necessarily adhere to any syntactic constraints. Most noticeably, the NT categories that were so prevalent in SBT have been discarded and a single NT category X is used (in addition to S, the root symbol). The hierarchical nature of the model addresses the shortcomings of PBT models by allowing the modeling of discontinuous spans and longer-range reordering patterns, and the translation units only have a single NT category into which we can substitute any other rule, avoiding the data sparseness issues with SBT. Unlike SBT, HPBT does not rely on any external syntactic resources like a parser; rules are extracted directly from word alignments using the consistent phrase pairs from PBT. Rule generation can be understood as a 'subtraction' of a consistent phrase pair from a larger phrase pair (i.e., removing nested consistent phrase pairs from larger ones). As an example, from the top-left box in Fig. 2.2 we can extract the following Hiero rules[11] (among others): "*diese fragen werden* ||| these questions will", "*diese* X^1 ||| these X^1", and "X^1 *fragen* X^2 ||| X^1 questions X^2". In HPBT, NTs are aligned across source and target sides, and this is indicated by the superscript notation. Lastly, two glue rules where the left-hand side NT is the root symbol S are added, allowing the grammar to derive sentence pairs by left-to-right concatenation of translation rules.

[10] Sentences annotated with their phrase structure trees; see Marcus et al. (1993).
[11] In these rules, we omit the left-hand side NT, which is X.

2.1.3 Minimal Grammars

The discussion of the prevalent formalizations points to a common issue: in the process of building a translation inventory from word alignments, the translation model sizes grow significantly. In PBT, we extract all phrase pairs, minimal or composed (i.e., nested consistent phrase pairs) consistent with the alignments. In SBT, an algorithm to extract minimal translation rules was initially proposed, but it was found that composing rules does better, albeit by drastically increasing grammar size. In HPBT the phrase extraction process is similar to PBT and we often need to resort to thresholds and other restrictions to avoid a model blow-up. Basically, all of these models incorporate a limited form of context dependence by including larger rules in their inventories; larger rules lead to larger models, which result in computational (more translation options to consider during decoding) and statistical (more parameters to estimate) tractability concerns. Furthermore, the way these rules are scored is still relatively simplistic: apart from the lexical features for a phrase pair (§2.1.1), the convention is to include the log relative frequencies of rule occurrences in a parallel corpus (in both directions) as a feature. The relative frequency estimates are aggregated, corpus-level statistics and are absolutely invariant to the context in which a phrase is actually being translated. Perhaps a better avenue of exploration is to revert to minimal grammars, and instead incorporate context through scoring methods that are context-dependent.

Utilizing minimal grammars instead of the traditional, composed ones allows us to explore the transition from minimal to composed grammars by using auxiliary models that score translation options using context. Rules that are more context-sensitive are created without increasing the overall size of the phrasal inventory, but instead by holding this information in the auxiliary model. For each sentence in the training data we extract the minimal set of synchronous rules consistent with the word alignments; then, rule types across all sentence pairs are combined to form a minimal grammar. To extract a set of minimal rules, we use the linear-time extraction algorithm of Zhang et al. (2008). We give a rough description of their method below, and refer the reader to the original paper for additional details.

The algorithm returns a complete minimal derivation tree for each word-aligned sentence pair, and generalizes an approach for finding all common intervals (pairs of phrases such that no word pair in the alignment links a word inside the phrase to a word outside the phrase) between two permutations (Uno and Yagiura, 2000) to sequences with many-to-many alignment links between the two sides, as in word alignment. The key idea is to encode all phrase pairs of a sentence alignment in a tree of size proportional to the source sentence length, which they call the normalized decomposition tree. Each node corresponds to a phrase pair, with larger phrase spans represented by higher nodes in the tree. Constructing the tree is analogous to finding common intervals in two permutations, a property that they leverage to propose a linear-time algorithm for tree extraction. Converting the tree to a set of minimal rules for the sentence pair is straightforward, by replacing nodes corresponding to spans with lexical items

or NTs in a bottom-up manner.[12]

2.1.4 Related Work

There has been a scattering of work that addresses the limitations of phrasal models by concentrating on incorporating context to improve upon the context-independent relative frequency features. The most significant line of work in this regard is n-gram translation (Mariño et al., 2006, Durrani et al., 2011), which uses Markov models to model context in the form of a history. Thus, while no new rules are added to the translation inventory, additional parameters in the form of an auxiliary Markov model dictate the dynamics of translation. The n-gram framework allows the use of heuristic smoothing techniques from language modeling (Chen and Goodman, 1999) to indirectly capture context low-dimensionally, and while more principled approaches based on Pitman-Yor priors achieve good performance (Feng and Cohn, 2013), the n-gram methods are still limited by their unidirectional (i.e, left-to-right) notion of context and their reliance on smoothing as a proxy to reasoning about the effect of context dimensionality on translation. Except for Vaswani et al. (2011), who leverage the n-gram translation framework to explore this transition from minimal to composed grammars in the setting of T2S translation, this line of work has broadly ignored the fact that context is being doubly modeled in some sense, both in large, composed rules as well as through a Markov model. This decision often limits the extent to which context can be considered for tractability reasons. Naturally, one option is to combine the explicit low-dimensional hypothesis with the n-gram translation model, and learn transition parameters in the latent, recovered space. This combination introduces the idea of structured prediction to our low-dimensional hypothesis, and other variants can also be explored.

The usage of word-sense disambiguation techniques to compute context-dependent translation scores has also been explored, and we review this literature in §4.3.

2.2 Multi-view Assumption

What assumptions will our auxiliary models for context take? The standard MT models described in §2.1.2 make use of larger rules to incorporate context; we argue that this decision treats context **high-dimensionally**. When including surrounding context to make a larger rule, only the word identity information is used, and there is no notion of how similar two contexts or translation rules are. For example, the translation rules "*diese fragen* ||| these questions" and "*diese anfragen* ||| these requests" are as similar (or distant) to each other

[12] We filtered rules with arity 3 and above (i.e., containing more than 3 NTs on the RHS). While synchronous grammars are perfectly capable of handling such cases, computation quickly becomes intractable with higher arity rules.

according to a phrasal model as the pairs "*diese fragen* ||| these questions" and "*diese antworten* ||| these answers". Even if a rule is a substring of another rule e.g., "*fragen* |||| questions", it is not any more similar to "*diese fragen* ||| these questions" than other translation rules. In other words, if we were to embed our translation rules in $\mathbb{R}^P$, where P is the number of rules in our inventory, then all rules would be equally distant from each other as they form vertices of a standard $P-1$-dimensional simplex. This kind of encoding is also known as a **one-hot encoding**, because each translation rule is represented as a P-dimensional vector with exactly one entry in the vector equal to 1 (at a position indicating the identity of the translation rule). The dimensionality of the space is equal to the number of translation rules in our inventory, which is problematic if we want to incorporate some notion of rule similarity in our model. As is well documented in the machine learning literature, the curse of dimensionality (Bishop, 2006) implies that models trained in such high-dimensional spaces are prone to overfitting, not to mention the computational issues (although sparse representations of high-dimensional spaces help considerably in this regard).

One of the central points of this thesis is that the effect of context on translation is **low-dimensional**. The question then shifts to how we can recover appropriate low-dimensional spaces from the high-dimensional, one-hot space. One possible way to achieve this goal is to use the **multi-view assumption**, which originates from the semi-supervised learning literature (Yarowsky, 1995, Blum and Mitchell, 1998). Informally, the assumption states that we can partition the representation of an example (represented as a vector of real values) into two (or potentially more) different views, and that either view of the example is sufficient to make accurate predictions.[13] Then, a multi-view learning algorithm forces agreement between the two predictors that have been trained on separate splits of the data, with the intuition being that the complexity of the learning problem should be reduced by eliminating hypotheses from each view that do not agree with each other. A formal treatment of the assumption depends on the analysis used, but here we state the regret-based formulation of Kakade and Foster (2007):

$$L(f^{(1)}) - L(f) \leq \epsilon$$
$$L(f^{(2)}) - L(f) \leq \epsilon$$

where $L(\cdot)$ is the (expected) squared loss function which takes a predictor as an argument, $f^{(\nu)}$ is the best linear predictor based on view $\nu \in \{1, 2\}$, and f is the best linear predictor based on both both views. The assumption implies that (only on average) the predictors must agree.

Kakade and Foster (2007) showed that, assuming second order information (moments) are

[13] This form of the assumption is known as the redundancy assumption, and is used in Kakade and Foster (2007). Ando and Zhang (2007) introduce a conditional independence-based assumption, and Foster et al. (2008) showed that a weaker form of this assumption and the redundancy assumption are, for the purposes of dimensionality reduction and prediction, equivalent.

known, a technique known as **canonical correlations analysis** (CCA, Hotelling, 1936) can be applied to the two views to recover a subspace for supervised learning. In a nutshell, CCA recovers a pair of projection matrices (i.e., two sets of basis vectors) that project each view into a shared, latent space, such that the correlations between the projected views is mutually maximized. Uncorrelated noise in each view is removed, and the procedure recovers directions in which the two views agree. In the regression case, the norm of the linear predictor f should be computed in this basis in order to regularize the regression; doing so allows us to identify linear predictors that have large weights in directions that are less correlated with the other view, as these predictors have larger norms. Foster et al. (2008) followed up and showed that under several different assumptions, dimensionality reduction via truncated (rank-reduced) CCA reduces sample complexity for supervised learning problems if the predictor is learned in the recovered latent space. Very little predictive information is lost by operating in this space: a small amount of bias is introduced at the expense of significant variance reduction. They also show that, under certain conditions, the best linear predictor that utilizes both views relies on the concatenation of the projected views in the hidden subspace; this optimal dimensionality reduction cannot be improved upon without additional assumptions. CCA is thus an important technique that we will use to recover the low-dimensional subspaces for context and translation rules, and it is reviewed in more detail below.

2.2.1 Canonical Correlations Analysis

We are given n instances or training examples. Each instance $\mathbf{x}_i$ is represented as a d-dimensional vector and belongs to a class $h \in [1, P]$, where P is the total number of classes (responses). In our case, the total number of classes is the number of translation rules in the model, and each instance is a source phrase occurrence in the corpus. Each source phrase is represented by d features extracted from surrounding source words in the sentence. Hence, the original data matrix is $Z \in \mathbb{R}^{n \times d}$. The multi-view assumption says that we can split the n d-dimensional vectors into two views of length d_1 and d_2 respectively (i.e., $d_1 + d_2 = d$), resulting in $X \in \mathbb{R}^{n \times d_1}$ and $Y \in \mathbb{R}^{n \times d_2}$. For a desired rank $k < \min\{d_1, d_2\}$, we compute the CCA between X and Y, resulting in a pair of projection matrices $A \in \mathbb{R}^{d_1 \times k}$ and $B \in \mathbb{R}^{d_2 \times k}$.

The objective of CCA is to find basis vectors (which form the columns of matrices A and B) such that the correlation ρ between projections of variables or instances onto these vectors is mutually maximized. The first column in the matrices A and B are the directions that maximize correlation across all datapoints. Subsequent columns in A (resp. B) have the implicit constraint that their projections are in orthogonal subspaces to the column space of the partially-constructed A (B) matrix, otherwise the previous directions would not be the ones that maximize correlation. Expressing this objective mathematically, we solve the

following optimization problem:

$$
\begin{aligned}
\max \rho &= \max_{A,B} \ \operatorname{corr}(XA, YB) \\
&= \max_{A,B} \frac{(XA)^T(YB)}{\|XA\|\|YB\|} \\
&= \max_{A,B} \frac{A^T X^T Y B}{\sqrt{A^T X^T X A B^T Y^T Y B}} && (2.6) \\
&= \max_{A,B} \frac{A^T \operatorname{Cov}(X,Y) B}{\sqrt{A^T \operatorname{Var}(X) A B^T \operatorname{Var}(Y) B}} && (2.7) \\
&= \max A^T \operatorname{Cov}(X,Y) B && (2.8)
\end{aligned}
$$

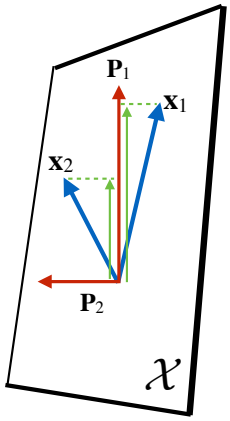
where in lines 2.6 and 2.7 we assume the data matrices are mean-centered. Note that by the arguments above, the columns in the matrix XA are orthogonal, and similarly with YB. In line 2.8, we make use of the property that the objective function is scale invariant, which can be easily verified.[14] Thus, $A^T\operatorname{Var}(X)A$ and $B^T\operatorname{Var}(Y)B$ can be set to the identity matrix since XA and YB consist of orthogonal columns, obviating the need for the denominator. Subsequently for shorthand, we use C_{XX} for $\operatorname{Var}(X)$, C_{YY} for $\operatorname{Var}(Y)$, and C_{XY} for $\operatorname{Cov}(X,Y)$.

Intuition

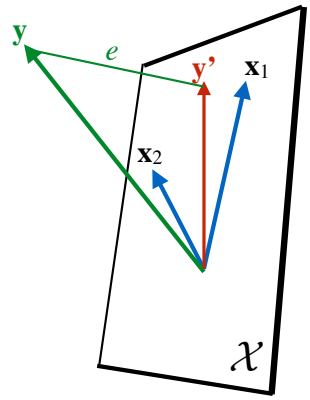
Before discussing the various ways to solve this optimization problem, an intuition of the CCA objective and what exactly it computes, especially in relation to principal components analysis (PCA, Pearson, 1901) and least-squares regression, is presented (Fig. 2.4). In PCA, the aim is to compute a dimension-reduced version of the data matrix X, which is originally of dimension d. A reasonable way to achieve this goal is to find the vectors or directions in which the data maximally varies i.e., the variance of the data is highest, and truncate below a certain number of directions (dimensions) because they do not contain much information of interest. These directions, which turn out to be orthogonal to each other, are known as the principal components.[15] Using the complete set of d principal components will perfectly reconstruct X, and using $k < d$ principal components reconstructs the data while minimizing the total squared reconstruction error (out of all sets of k linearly independent vectors). Figure 2.4a presents a two-dimensional example with two examples $\mathbf{x}_1$ and $\mathbf{x}_2$, and principal components $\mathbf{P}_1$ and $\mathbf{P}_2$. Note that the principal components lie in the same space as the data X; the data is used to compute the principal components, which in turn is used to provide a representation for the data.

In linear regression (Fig. 2.4b), the dependent variable $\mathbf{y}$ (in green) is not in the data plane,

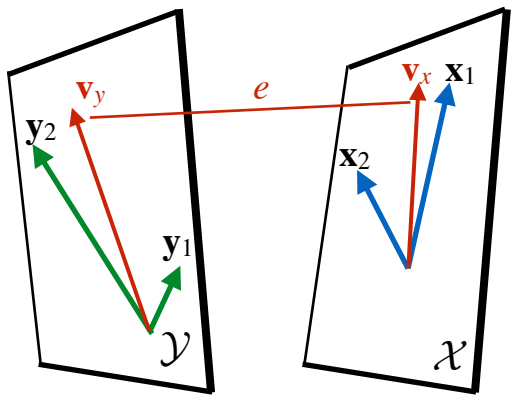
[14] Indeed, scaling should not affect the angle between two vectors, and the cosine of the angle is the correlation.
[15] The principal components correspond to the eigenvectors of the data covariance matrix $X^T X$.



(a) Principal Components Analysis.



(b) Linear Regression.



(c) Canonical Correlations Analysis.

Figure 2.4: A pictorial comparison between PCA, linear regression, and CCA. Intuitively, CCA can be thought of as PCA where each space has a dependent variable from the other space, as in regression. The correlation between vectors $\mathbf{v}_x$ and $\mathbf{v}_y$ is maximized.

and unless it is in the plane there will always be an error e when using the space of predictors X to construct $\mathbf{y}$. $\mathbf{y}$ is perpendicularly projected onto the data space, and a weighted linear combination of the data is used to reconstruct the projection. In CCA (Fig. 2.4c), the dependent variable resides outside the data plane as in regression, except this time there are two sets of variables that predict each other simultaneously. Specifically, $\mathbf{v}_x$ and $\mathbf{v}_y$ are a pair of **canonical correlations**; each canonical correlation is a linear combination of the variables in the respective data space, as in the case of linear regression. So for the data space X, $\mathbf{v}_y$ is

akin to $\mathbf{y}$ in linear regression, and $\mathbf{v}_x$ is equivalent to $\mathbf{y}$' (and similarly for the Y space). Note that $\mathbf{v}_x$ and $\mathbf{v}_y$ are not orthogonal, but are extracted so as to minimize the angle between them (i.e., maximize the correlation). After finding the first pair of canonical correlations, the next pair $\mathbf{v}_{x,2}$ and $\mathbf{v}_{y,2}$ is found such that $\mathbf{v}_{x,2}$ is orthogonal to $\mathbf{v}_x$ and $\mathbf{v}_{y,2}$ is orthogonal to $\mathbf{v}_y$. We repeat the process to find the k canonical correlations.

Optimization

There are a number of ways to solve the objective in Eq. 2.8. The first and most direct method is to re-formulate the objective function by adding constraints using the method of Lagrange multipliers, with the constraints dictating the scaling of the vectors (since the original objective function is scale invariant). Hardoon et al. (2004) present a detailed derivation of the solution using this approach, where it is shown to be equivalent to the solution of the generalized eigenvalue problem.[16] Luckily, the generalized eigenvalue problem can be converted into a symmetric eigenvalue problem in this case, and a standard algorithm for computing eigendecompositions of symmetric matrices can be used to recover the canonical directions.

In general, an interesting trick is that we never need to explicitly optimize Eq. 2.8 (Press, 2011). First, note that in addition to the columns of XA and YB being orthogonal to each other, they are also cross-orthogonal i.e., $A^T X^T Y B \in \mathbb{R}^{k \times k}$ is a diagonal matrix. This property is also known as **bi-orthogonality**. In other words, the correlation between the ith column in XA and any other column $j, j \neq i$ in YB is zero. Using proof by contradiction, if the correlation were not zero then an appropriately weighted linear combination of the ith and j^{th} columns in YB would have a higher correlation with the ith column of XA. Next, by counting arguments we can show that *any* construction of the projection matrices A and B that satisfy the bi-orthogonality property will produce the canonical correlations. For the simple case where $d_1 = d_2$ and with no column degeneracies in the data (full rank), there are $2k^2$ parameters in A and B. The conditions $A^T C_{XX} A = I$ and $B^T C_{YY} B = I$ impose $k(k+1)/2$ constraints each, since they are symmetric matrices. The cross-orthogonal constraint that $A^T X^T Y B$ be diagonal imposes $k^2 - k$ constraints, since the other elements of this matrix are unconstrained. Hence, the sum of the constraints equals the number of parameters (degrees of freedom), meaning there can only be countable, isolated solutions.

In the more general case, the constraints alone may not uniquely specify a solution but the largest correlations can be selected using a singular value decomposition (SVD, Golub and Van Loan, 1996). A rank-k SVD is a factorization of a matrix $W \in \mathbb{R}^{n \times m}$ into three component matrices: $W \approx U \Sigma V^T$, where $U \in \mathbb{R}^{n \times k}$ and $U \in \mathbb{R}^{m \times k}$ are orthogonal projection matrices

[16] The generalized eigenvalue problem is the problem of finding vectors $\mathbf{v}$ that obey $A\mathbf{v} = \lambda B\mathbf{v}$ for matrices A and B and λ is some scalar and must obey $\det(A - \lambda B) = 0$. Compare this formulation to the standard eigenvalue problem, where vectors $\mathbf{v}$ obey $A\mathbf{v} = \lambda\mathbf{v}$.

that contain the left and right singular vectors, and $\Sigma \in \mathbb{R}^{k \times k}$ is a diagonal matrix that contains the singular values. The SVD is used for an immense variety of applications (from computing the pseudoinverse of a matrix to solving homogeneous linear equations), but in our case the rank-k SVD minimizes the Frobenius norm between the original matrix and the low-rank reconstruction[17], and is thus used to recover low-rank matrices from full-rank ones.

Since any pair of matrices A, B that satisfy the bi-orthogonality property recovers the canonical correlations, there are a number of options at our disposal:

- Björck and Golub (1973) suggest first using a QR decomposition on the data matrices: $X = Q_x R_x,\ Y = Q_y R_y$. The Q matrices are orthogonal and thus provide an orthonormal basis, while the R matrices are upper-triangular. Then, SVD is used to compute an orthonormal basis for the cross-product space: $Q_x^T Q_y = U \Sigma V^T$. We can recover A and B as $A = R_x^{-1} U,\ B = R_y^{-1} V$, which is easily computable through back-substitution since the R matrices are upper triangular.

- The main purpose of the QR step is to obtain orthogonal bases, with a secondary purpose being that the transform into this basis is easily invertible (to recover A and B). SVD also satisfies these criteria, so we can replace the QR decomposition step with an SVD-based one: $X = U_x \Sigma_x V_x^T,\ Y = U_y \Sigma_y V_y^T$. The second SVD is computed on the cross-product space, as before: $U_x^T U_y = U \Sigma V^T$, and A and B can be easily recovered: $A = V_x \Sigma_x^{-1} U,\ B = V_y \Sigma_y^{-1} V$. With this approach, we also get an idea of the allocation of variance as in PCA[18] (Press, 2011).

- The QR and SVD steps de-correlate or "whiten" the data, which can also be achieved by defining a change of basis using the square root inverse (SRI) operation. Since the aim of any whitening transformation is to transform the data covariance matrix into the identity matrix, multiplying the data matrices X and Y by the SRI of the covariance matrix[19] will achieve this goal. In this approach, an SVD of the cross-product space is computed after a change in basis has been applied to the space:

$$C_{XX}^{-\frac{1}{2}} C_{XY} C_{YY}^{-\frac{1}{2}} = U \Sigma V^T \tag{2.9}$$

  following which we apply the change of basis to the result to yield matrices A and B: $A = C_{XX}^{-\frac{1}{2}} U,\ B = C_{YY}^{-\frac{1}{2}} V$. The approach was first used in Mardia et al. (1979), and it is straightforward to show that these matrices satisfy the bi-orthogonality property.

[17] This result is known as the Eckart-Young theorem.

[18] The notion of allocation of variance is not exactly the same as PCA, but since the canonical correlations are orthogonal, they also partition the variance like the principal components do.

[19] The SRI of the covariance matrix is well-defined: the covariance matrix is positive-definite, therefore its square root consists of real values and is invertible.

While Chs. 3 and 4 both make use of the multi-view assumption and CCA, in Ch. 3 we simply the computation by assuming $C_{XX} = I$ and $C_{YY} = I$, which means the CCA computation boils down to SVD of the cross-product space. In Ch. 4, we investigate several approximations to C_{XX} and C_{YY} (§4.1.1).

Related Approaches

Because of the bi-orthogonality property, CCA is an expensive computational operation. The QR or SVD steps on the data matrix X (or Y) can be intractable, especially if these matrices have very large dimension d_1 (resp. d_2).[20] Similarly, whitening the data matrix X (or Y) requires the inversion of a $d_1 \times d_1$ (or $d_2 \times d_2$) matrix. As a result, a number of large-scale approaches towards CCA have been introduced. Lu and Foster (2014) presented an iterative approach to computing CCA that is specifically optimized for large, sparse datasets, where the computation reduces to a sequence of fast least squares solutions on relatively small matrices. For extremely large problems, an approximate least squares solver that works in a manner similar to principal component regression is used in place of the exact one. There has also been a line of work that applies random matrix methods (Halko et al., 2011) to the CCA problem. Avron et al. (2013) apply a transformation known as a Walsh-Hadamard Transform to the data matrices prior to computing CCA. Informally, the transform "spreads" the information in the data equally among the input rows, which allows these matrices to be amenable to uniform row sampling. Walsh-Hadamard matrices are cheaper to store and applying them is faster than multiplying by dense, Gaussian random matrices. This transform was also used by Lopez-Paz et al. (2014) for fast, scalable *non-linear* (kernelized) CCA. Mineiro and Karampatziakis (2014) present another randomized algorithm, which uses a randomized range finder to iteratively compute the (orthogonal) column space or range of the data matrices X and Y, after which a Cholesky decomposition and SVD is used to find the canonical correlations.

Partial Least Squares (PLS, Rosipal and Krämer, 2006) is a technique used to model sets of multivariate data, and in particular, orthonormalized PLS is a variant which explicitly considers variance in one of the views. In other words, in Eq. 2.8 we constrain only one view's covariance to be the identity matrix, and leave the other view unconstrained. The relationship between orthonormalized PLS and CCA was explored in Sun et al. (2008), where they build on top of the relationship between CCA and Fisher Linear Discriminant Analysis to show that the projection directions (i.e., canonical correlations) learned by CCA and orthonormalized PLS differ only by a rotation, which is not problematic since rotation operations are isometric. The advantage of using the least squares formulation and correcting for the rotation afterwards is that sparsity-inducing, ℓ_1-regularized algorithms can be applied to learn the canonical correlations, which is not trivial to do in the standard formulation of CCA.

[20] QR decomposition on an $n \times d_1$ matrix is $O(nd_1^2)$, and SVD is $O(\min(nd_1^2, N^2 d_1))$ (Golub and Van Loan, 1996).

Moving away from linear algebraic approaches for revealing latent subspaces, Ghahramani (1996) draws an interesting connection between single-layer linear neural networks and CCA. In general, neural networks are a powerful model family that, due to the non-linearities involved, can accurately model a large number of phenomena given adequate data; for a brief primer, see Bishop (2006). When modeling the relationship between input and output variables, a single layer network inserts a **hidden layer** between the variables. The hidden layer acts as a bottleneck, in that all inputs must go through this layer and then to the output layer in a two-step process. Consider such a network which takes $\mathbf{x}_i \in \mathbb{R}^{d_1}$ as input and predicts $\mathbf{y}_i \in \mathbb{R}^{d_2}$ as the output. Let the hidden layer contain k hidden units such that $k < \min\{d_1, d_2\}$. The relationship between input and output variables can then be written as $\hat{\mathbf{y}}_i = BA^T\mathbf{x}_i$, where $A \in \mathbb{R}^{d_1 \times k}$ and $B \in \mathbb{R}^{d_2 \times k}$ are the projection matrices as in CCA. Ghahramani showed that if we train the network to minimize the Mahalanobis distance (using the covariance in Y as the covariance matrix) between the prediction $\hat{\mathbf{y}}_i$ and the true output $\mathbf{y}_i$, then the objective is minimized when using the left and right singular vectors of Eq. 2.9 with the largest singular value, which is exactly a solution to CCA. Note that the analysis is specific to the case where no non-linear activation functions are used, and these activation functions are one of the ways in which neural networks shine, as it allows them to approximate complex functions reasonably accurately. Thus, adding these non-linearities and training networks to minimize Mahalanobis distance is equivalent to a non-linear variant of CCA.

Ghahramani (1996) viewed CCA in a similar manner to Sun et al. (2008) by treating the second data space Y as a dependent variable to be predicted, and suitably modifying other aspects of the problem. A more direct approach is to actually design a deep (multi-layer) neural network that mimics CCA (Andrew et al., 2013), where representations of the two views are computed by passing the inputs through multiple non-linear layers. The objective is to maximize the correlation of the outputs from each of the stacked layers, which is optimized using backpropagation. In general, neural networks should be understood as an alternative technique for learning low-dimensional representations of high-dimensional spaces, but without the consistency guarantees provided by multi-view learning-oriented approaches. Interestingly, neural network optimization is inherently online in that it is amenable to gradient descent-based procedures; these procedures can also be more scalable in large-scale data scenarios. Thus, optimization techniques from neural networks can also be used for large-scale CCA, which is the idea that several of the discussed works use (Sun et al., 2008, Lu and Foster, 2014).

2.3 Manifold Assumption

In §2.2, we discussed the multi-view assumption and showed how it can be used to recover a low-dimensional subspace of interest. In order to evaluate our low-dimensional context-dependent translation hypothesis in the limit, we need to resort to another assumption that

enables a shift to the semi-supervised learning domain, where a large number of examples do not have label information: the **manifold assumption**.

Informally, the manifold assumption states that the data of interest lie approximately on a manifold $\mathcal{M}^k$ of lower dimension k than the ambient space in which the data nominally reside, $\mathbb{R}^d$.[21] A manifold is a topological space that is locally Euclidean near each point, but globally may be highly non-Euclidean. A classic example is the surface of a sphere, which is non-Euclidean but in small localities may be approximated as Euclidean. More formally, the assumption states that the support of the marginal probability distribution of the data (computed by summing over the labels in the joint probability distribution[22] over data and labels) is on some low-dimensional manifold (compared to the ambient Euclidean space).

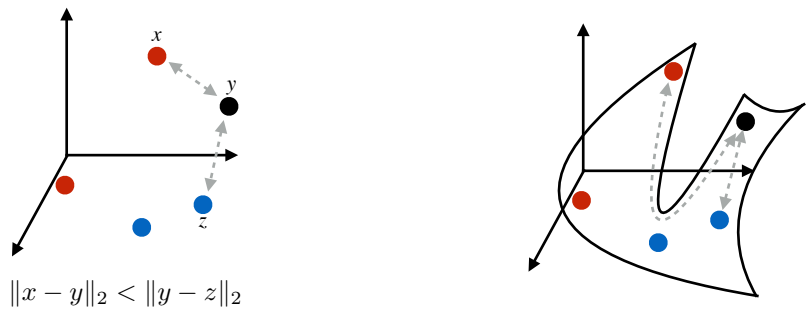
The motivation behind this assumption comes from the intuition that although natural data in its surface form resides in a high-dimensional space, the data is often generated by systems with much fewer underlying degrees of freedom and therefore have lower intrinsic dimensionality (Niyogi, 2013). In fact, PCA also makes use of a special case of the manifold assumption, namely that the data manifold consists of a single ellipsoid. Thus, the manifold assumption can also be seen as relaxing some of the stricter conditions imposed when we use the multi-view assumption and CCA. A related assumption that applies more to the distribution over labels is the smoothness assumption: the underlying target function over the manifold, which in this case is the distribution over labels conditional on the data, is smooth with respect to the underlying manifold. In other words, points on the data manifold that are connected via a path (or paths) through high density regions of the manifold are likely to have the same label (or very similar labels). The two assumptions work together and allow algorithms to take advantage of the structure of the data when inferring labels for unlabeled points, and in order to learn functions that respect the manifold assumption we impose the notion of smoothness over the manifold.

Fig. 2.5 provides a tangible example with two classes, underlining why the manifold assumption matters. The red points belong to one class, the blue points belong to the other, and the aim is to infer the most likely class of the black point (y). In Fig. 2.5a, we consider only the ambient dimension, and using the Euclidean distance, y is labeled red. In the context of manifolds, this distance is also known as the chordal distance. However by considering the manifold in Fig. 2.5b, we find that the red point is further away to y than the blue point when restricting ourselves to paths on the manifold. With the manifold assumption the geodesic distance i.e., the distance between two points along a path *on* the manifold, is the metric we use to determine similarity between points, and not the chordal distance.

It should be emphasized that there are two major forms of dimensionality reduction in the graph-based semi-supervised setting: the first is the embedding of each high-dimensional ex-

[21] This construction is technically known as an embedded manifold.

[22] To make this joint distribution concrete, in the case of MT the data consists of source phrases, and the labels are their corresponding translations.



(a) Chordal distance in the ambient space. **(b)** Geodesic distance on the manifold.

Figure 2.5: A tangible example showing that, assuming the manifold assumption is true, taking into account the geodesic distance and the structure of the manifold (Fig. 2.5b vs. Fig. 2.5a) is crucial.

ample in a graph, where the example is described purely in terms of its relations with its nearest neighbors. Inference is performed solely by taking into account the graph structure, and the actual position of the example in the ambient, Euclidean space is irrelevant.[23] The second is a dimensionality reduction step that can be applied to the ambient, Euclidean space prior to estimation of the manifold; a variety of standard dimensionality reduction techniques that operate in Euclidean space can be used, and we present results for one approach in §5.2.5.

In the following sections, we first review the basics of Riemannian manifolds i.e., the specific class of manifolds we consider (§2.3.1). We then introduce an important quantity defined for continuous manifolds, the Laplacian (§2.3.2), as well as its discrete counterpart, the graph Laplacian (§2.3.3). Intuitively, the Laplacian dictates how functions evolve over the graph by taking into account its structure. We conclude by discussing how inference over the graph respects the structure presented by the Laplacian (§2.3.4).

2.3.1 Riemannian Manifolds

Manifolds are very general mathematical objects, but for the purposes of this thesis we restrict ourselves to a specific subclass of **differentiable manifolds** called **Riemannian manifolds**. For a more detailed look at these concepts, the reader is encouraged to access Do Carmo (1992). A differentiable manifold is simply a manifold that, locally speaking, is linear enough to enable the machinery of calculus as if it were in some Euclidean space. Fig. 2.6 presents

[23] §2.3.3 will also review dimensionality reduction techniques that make use of the spectral properties of the Laplacian, specifically where inconsequential manifold directions are eliminated prior to graph propagation. We do not explicitly make use of such approaches in this thesis, however.

an example of the surface of a sphere, which is a 2-dimensional manifold in 3-dimensional ambient space. The plane tangent to the point p is the tangent space $T_p\mathcal{M}^k \in \mathbb{R}^k$, a real vector space that allow us to apply operators associated with differentiation to the point p on the manifold.

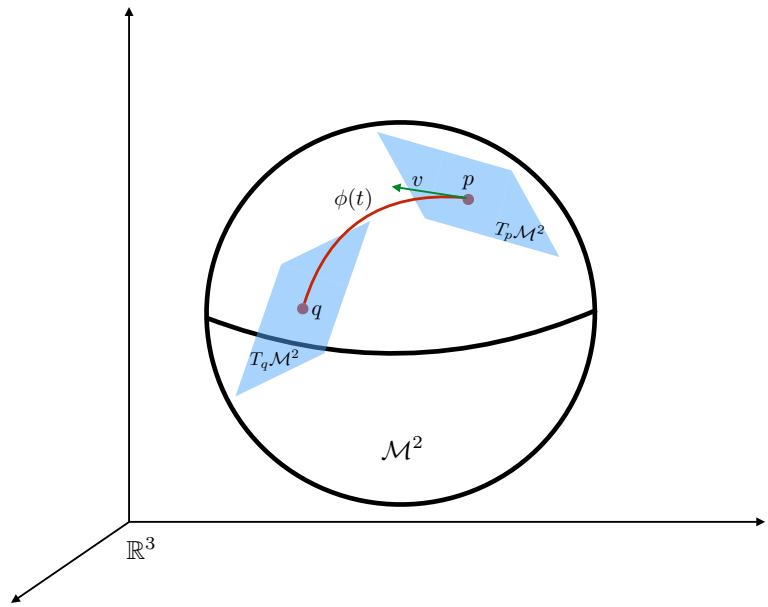


Figure 2.6: A two-dimensional manifold $\mathcal{M}^2$ (the surface of a sphere) embedded in $\mathbb{R}^3$.

A Riemannian manifold is a differentiable manifold such that the tangent space $T_p\mathcal{M}^k$ is an inner product space[24], which crucially allows the concept of a norm and thus a measurement of the length of the tangent vectors that reside in the tangent space. These tools allow us to define the geodesic distance on the manifold.[25] Referring to Fig. 2.6, we can introduce a smooth curve $\phi(t) : \mathbb{R} \to \mathcal{M}^k$ between points p and q parameterized by $t \in [0, 1]$; the derivative of this curve $\phi'(t_0)$ with respect to t evaluated at a specific point $t = t_0$ is the tangent vector v that resides in the tangent space $T_{t_0}\mathcal{M}^k$. Because the tangent space is an inner product space, we can compute the length of the tangent vector using the norm, and thus the length l of the curve ϕ is:

$$l(\phi) = \int_0^1 \|\phi'(t)\| dt$$

[24] Also known as a Hilbert space.

[25] An important point is that the collection of inner products as defined by the tangent spaces at various points on the manifold should vary smoothly as we traverse the manifold, otherwise the tangent space is not doing a good job of approximating manifold characteristics. Bengio et al. (2006) discuss the contribution of manifold curvature to this phenomenon.

the minimum of which, over all curves between p and q, is precisely the geodesic distance.

The geodesic allows us to define the notion of an exponential map $\exp_p : T_p\mathcal{M}^k \to \mathcal{M}^k$, which is a mapping from tangent vectors residing in $T_p\mathcal{M}^k$ to the manifold itself. Intuitively, the exponential map defines the mechanics of traversing the manifold: at a point p we pick a tangent vector with a certain norm, and traverse the manifold for a distance equal to this norm, bringing us to another point q on the manifold. The exponential map thus provides a natural coordinate system for navigating the manifold, since it allows the manifold to be analyzed completely in terms of tangent spaces. Note that these tangent spaces are of the same dimension as the manifold k, and in practical problems $k << d$, the ambient dimension.

2.3.2 The Manifold Laplacian

We now introduce a twice-differentiable function $f : \mathcal{M}^k \to \mathbb{R}$ that maps points on the manifold to the real line. With additional restrictions on the range, this function could be e.g., the marginal distribution of the data, but the issue is that the domain of this function is the manifold. Composing this function with the exponential map $\exp_p$ however, allows us to describe f as a function of just k variables: $\mathbb{R}^k \to \mathbb{R}$. Now, f is a standard function defined in Euclidean space, for which the Laplace operator or Laplacian is well-defined:[26]

$$\Delta_{\mathcal{M}} f(p) \equiv \sum_{i=1}^{k} \frac{\partial^2 f(\exp_p(x))}{\partial x_i^2}$$

The Laplacian is an operator that has a number of different interpretations in various fields, and is especially useful for the modeling of physical phenomena like heat flow. One way to look at it is the trace of the Hessian matrix (the matrix of second-order partial derivatives), which in turn contains information about the curvature of the function on the manifold. In other words, the Laplacian tells us how much the value of the function f differs from its average value taken over the surrounding points on the manifold, an average rate of change of sorts.

Why do we care about this quantity? Ideally, we want a smooth map f from $\mathcal{M}^k$ to $\mathbb{R}$: if two points are close to each other on the manifold, then their images according to f should be close to each other as well, requiring us to respect the manifold structure. This desideratum can be expressed as:

$$\arg\min_f \int_{\mathcal{M}} \|\nabla_{\mathcal{M}} f\|^2 = \arg\min_f \int_{\mathcal{M}} f \cdot \Delta_{\mathcal{M}} f \tag{2.10}$$

where $\nabla_{\mathcal{M}}$ is the gradient with respect to the manifold, and the equality is due to Stokes'

[26] When applied to the specific case of Riemannian manifolds, it is known as the Laplace-Beltrami operator.

theorem. Eq. 2.10 essentially states that the Laplacian is the fundamental quantity that controls how smooth a function is over the manifold. The equation is in quadratic form, and it is well-known from linear algebra and functional analysis (Kreyszig, 1989) that minimizing the quadratic form of a linear operator is equivalent to computing the eigenfunctions and eigenvalues of that operator: $\Delta_{\mathcal{M}} f = \lambda f$, where λ represents the eigenvalues. Specifically, the ith eigenfunction ϕ_i is the minimizer to Eq. 2.10 with the minimal value λ_i. Furthermore, it can be shown that the Laplace operator is bounded and self-adjoint; these properties provide amenable characteristics to the eigenspace: eigenvalues are real, positive, and discrete, and most importantly the eigenfunctions form an orthonormal basis for the space of square-integrable functions on the manifold. Essentially, we can write any function defined on the manifold as the infinite sum of weighted basis functions, which turn out to be the eigenfunctions of the Laplace operator.

The Laplacian provides a good idea of the kinds of functions to use such that the geometry of the data is respected, since the eigenfunctions provide a set of basis functions that are specifically adapted to the geometry of the manifold. In the continuous case e.g., the surface of a sphere, the Laplace operator can be explicitly computed.[27] However, in our instance we are exposed to examples that have been sampled from the low-dimensional data manifold; thus, we need to approximate the characteristics of the manifold using discrete samples from it. How do we estimate a natural coordinate system for the unknown manifold, when only given samples?

2.3.3 The Graph Laplacian

Luckily, there is a well-defined discrete version of the Laplace operator that is called the graph Laplacian.[28] Given a graph $G = (V, E)$ consisting of vertices V and edges E, we define a function g that is analogous to f, except the domain is no longer the manifold $\mathcal{M}^k$ but rather the vertices of the graph: $g : V \to \mathbb{R}$. From n random samples $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, we can construct the graph Laplacian as a random matrix $L \equiv D - W \in \mathbb{R}^{n \times n}$, where W is a similarity matrix such that $w_{ij} \in [0, 1]$ denotes the similarity between points i and j, and D is a diagonal matrix such that $D = \text{diag}(\sum_k w_{ik})$. Note that theoretical analysis which links the manifold and graph Laplacian relies on the graph being a neighborhood graph: instead of a fully-connected graph, we assume a sparsely-connected one where only a subset of all possible edge connections are greater than zero. These can either be ϵ-neighborhood graphs, where an edge weight w_{ij} between nodes i and j is set to zero if $w_{ij} < \epsilon$, or m-neighborhood

[27] In many instances, computing the Laplacian introduces a deep connection between harmonic analysis and Riemannian geometry. For example, the classical Fourier series are simply the eigenfunctions of the Laplacian when the manifold is a circle, which is intuitive since the set of functions defined on the circle are the periodic functions.

[28] In fact, there is a large body of work known as spectral graph theory that specifically analyzes the spectral properties of graphs and objects derived from graphs, like the Laplacian; see Chung (1997) for more details.

graphs, where the nearest m neighbors by similarity are active $w_{ij} > 0$, with the rest of the connections set to zero.

Using the definition of the graph Laplacian, it is easy to show that:

$$\sum_{i,j} W_{ij}(g_i - g_j)^2 = g^T L g \tag{2.11}$$

which can be seen as the finite-sample version of Eq. 2.10. Hence, minimizing Eq. 2.11 by computing the eigenvectors[29] of the Laplacian is equivalent to learning functions where the functional values between adjacent points i and j are minimized proportional to how similar i and j are.

Building on top of a number of results (Belkin, 2003, Lafon, 2004, *inter alia*), Belkin and Niyogi (2008) presented uniform convergence results (over the class of differentiable functions on the manifold) showing the graph Laplacian converges to the Laplace operator on the manifold as $n \to \infty$, for uniformly sampled points on the manifold as well as arbitrary probability distributions.[30] Thus, not only do we have a way to control for smoothness when learning a predictor by respecting the manifold structure (the Laplacian), but we can also empirically estimate this quantity reliably from data. Most importantly, the asymptotic rate of convergence depends not on d, the ambient dimension but rather k, the intrinsic dimension of the manifold.

The Laplacian Eigenmaps algorithm (Belkin and Niyogi, 2003) is a dimensionality reduction approach that directly operates on the Laplacian. The Laplacian eigenspace is computed; the first eigenvector corresponds to the constant function over the graph, with the corresponding eigenvalue equal to zero. The subsequent k eigenvectors are used to provide a basis that respects the manifold structure of the data, where k is a user-defined hyperparameter. Each eigenvector is in $\mathbb{R}^n$, so assembling the eigenvectors we obtain an $n \times k$ embedding matrix, where the embedding of the ith example is given by the ith row. Since the eigenspace of the Laplacian of a graph provides a natural coordinate system for the graph, and the graph is a finite-sample representation of a k-dimensional manifold, this approach should provide an optimal embedding (from a squared error minimization perspective, Eq. 2.11). Other well-known approaches (Roweis and Saul, 2000, Tenenbaum et al., 2000) compute eigenspaces for matrices that are similar to the Laplacian, but without the theoretical guarantees of convergence to the manifold Laplacian.

[29] We are dealing with a finite-dimensional operator on matrices i.e., the graph Laplacian and not an infinite-dimensional operator on functions, thus eigenvectors and not eigenfunctions.

[30] Technically for this particular theorem W should take the form of a Gaussian kernel, but other works (Hein et al., 2005, 2007) have generalized this notion to isotropic data-dependent weights.

2.3.4 Graph Propagation

Instead of taking a dimensionality reduction approach where the eigenspace is truncated, in this thesis we adopt a more direct viewpoint that is conceptually simple and scalable. In particular, variants of the label propagation algorithm (Zhu and Ghahramani, 2002) are used for inference over the graph in §5.1.4. By only considering each example's similarities with its m nearest neighbors during inference, the graph embedding is itself a type of low-dimensional representation. In this section, we establish the connection between label propagation and the graph Laplacian; for further details, see Bengio et al. (2006).

In a semi-supervised learning setup, label propagation is an iterative algorithm that transfers label information from labeled nodes to unlabeled nodes by following the graph's structure. The dynamics of the propagation is dictated by the **random walk matrix** $P = D^{-1}W$.[31] The row normalization is done to make the similarity matrix stochastic. P has the same eigenvalues as $I - \mathcal{L}$, where $\mathcal{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$, since $D^{-1}W = D^{-\frac{1}{2}}(I - \mathcal{L})D^{\frac{1}{2}}$. $\mathcal{L}$ is also referred to as the normalized Laplacian (Chung, 1997). Thus, label propagation performs inference by enforcing smoothness with respect to the graph structure via the Laplacian. Furthermore, the label propagation update equation (see Eq. 5.2 for a version specific to translation distributions) can be seen as an iterative Jacobi update to a linear system of equations that minimizes a weighted quadratic cost criterion i.e., Eq. 2.11 (Bengio et al., 2006). The matrices of the linear system dictate the dynamics of information or label "flow" over the manifold, and the eigenspace of this linear system provides a natural way to explain these dynamics. This eigenspace could potentially be truncated similar to Belkin and Niyogi (2003), which would entail a modified random walk matrix P, but we did not investigate this particular form of dimensionality reduction in §5.2 and leave it for future work.

[31] Label propagation can be seen as a random walk of labels over the graph; see Zhu et al. (2003) for additional details.

Chapter 3

Low-Dimensional Embeddings of Translation Units

"Hidden nature is secret God."

— Sri Aurobindo

In this chapter, we consider learning low-dimensional representations of translation units that are expressed (or featurized) in terms of the context which they occur in. Specifically, a latent-variable model for synchronous context-free grammars (SCFGs, Lewis and Stearns, 1968), as applied to hierarchical phrase-based translation (HPBT, Chiang, 2007), is learned. The non-terminals in each rule are augmented with latent states in a context-dependent manner, which we learn from a parallel corpus. In this case, we project a high-dimensional representation of a translation rule, represented with the empirical covariances of inside and outside tree features in synchronous trees where the rule occurs as a non-terminal, into a low-rank space.[1]

Translation models based on SCFGs treat the translation problem as a context-free parsing problem. A parser constructs trees over the input sentence by parsing with the source language projection of an SCFG, and each derivation induces translations in the target language (Chiang, 2007). However, in contrast to syntactic parsing, where linguistic intuitions can help elucidate the "right" tree structure for a grammatical sentence, no such intuitions are available for synchronous derivations, and so learning the "right" grammars is a central challenge.

Of course, learning synchronous grammars from parallel data is a widely studied problem (Wu, 1997, Blunsom et al., 2008a, Levenberg et al., 2012, *inter alia*). However, there has been less exploration of learning rich non-terminal categories, largely because previous efforts to learn such categories have been coupled with efforts to learn derivation structures—a com-

[1] This chapter is based on material published originally in Saluja et al. (2014a).

putationally formidable challenge. One popular approach has been to derive categories from source and/or target monolingual grammars (Galley et al., 2004, Zollmann and Venugopal, 2006, Hanneman and Lavie, 2013). While often successful, accurate parsers are not available in many languages: a more appealing approach is therefore to learn the category structure from the data itself.

In this chapter, we take a different approach to previous work in synchronous grammar induction by assuming that reasonable tree structures for a parallel corpus can be chosen heuristically, and then, fixing the trees (thereby enabling us to sidestep the worst of the computational issues), we learn non-terminal categories as latent variables to explain the distribution of these synchronous trees. This technique has a long history in monolingual parsing (Petrov et al., 2006, Liang et al., 2007, Cohen et al., 2014), where it reliably yields state-of-the-art phrase structure parsers based on generative models, but we are the first to apply it to translation.

We first generalize the concept of latent PCFGs to latent-variable SCFGs (§3.1). We then follow by a presentation of the tensor-based formulation for our parameters, a representation that makes it convenient to marginalize over latent states. Subsequently, two methods for parameter estimation are presented (§3.3): a spectral approach based on the method of moments, and an EM-based likelihood maximization. Results on a Chinese–English evaluation set (§3.4) indicate significant gains over baselines and point to the promise of using latent-variable synchronous grammars in conjunction with a smaller, simpler set of rules instead of unwieldy and bloated grammars extracted via existing heuristics, where a large number of context-independent but un-generalizable rules are utilized. Hence, the hope is that this work promotes the move towards translation models that directly model the conditional likelihood of translation rules via (potentially feature-rich) latent-variable models which leverage information contained in the synchronous tree structure, instead of relying on a heuristic set of features based on empirical relative frequencies (Koehn et al., 2003) from non-hierarchical phrase-based translation.

3.1 Latent-Variable SCFGs

Before discussing parameter learning, we introduce latent-variable synchronous context-free grammars, by extending the definition of L-PCFGs (Matsuzaki et al., 2005, Petrov et al., 2006) to synchronous grammars as used in machine translation (Chiang, 2007). The key difference in comparison to L-PCFGs is that the right-hand side (RHS) non-terminals of synchronous rules are aligned pairs across the source and target languages. A latent-variable SCFG (L-SCFG) is a 6-tuple $(\mathcal{N}, m, n_s, n_t, \pi, t)$ where:

- $\mathcal{N}$ is a set of non-terminal (NT) symbols in the grammar.

- $[m]$ is the set of possible hidden states associated with NTs. Aligned pairs of NTs across the source and target languages share the same hidden state. We assume that the states associated with NTs on the RHS are *not* conditionally independent given the latent state of the left-hand side (LHS).
- $[n_s]$ is the set of source side words, i.e., the source-side vocabulary, with $[n_s] \cap \mathcal{N} = \emptyset$.
- $[n_t]$ is the set of target side words, i.e., the target-side vocabulary, with $[n_t] \cap \mathcal{N} = \emptyset$.
- The synchronous production rules compose a set $\mathcal{R} = \mathcal{R}_0 \cup \mathcal{R}_1 \cup \mathcal{R}_2$:
  - Arity 2 (binary) rules ($\mathcal{R}_2$):

    $$a(h_1) \rightarrow \langle \alpha_1 b(h_2) \alpha_2 c(h_3) \alpha_3, \beta_1 b(h_2) \beta_2 c(h_3) \beta_3 \rangle$$

    or

    $$a(h_1) \rightarrow \langle \alpha_1 b(h_2) \alpha_2 c(h_3) \alpha_3, \beta_1 c(h_2) \beta_2 b(h_3) \beta_3 \rangle$$

    where $a, b, c \in \mathcal{N}$, $h_1, h_2, h_3 \in [m]$, $\alpha_1, \alpha_2, \alpha_3 \in [n_s]^*$ and $\beta_1, \beta_2, \beta_3 \in [n_t]^*$.
  - Arity 1 (unary) rules ($\mathcal{R}_1$):

    $$a(h_1) \rightarrow \langle \alpha_1 b(h_2) \alpha_2, \beta_1 b(h_2) \beta_2 \rangle$$

    where $a, b \in \mathcal{N}$, $h_1, h_2 \in [m]$, $\alpha_1, \alpha_2 \in [n_s]^*$ and $\beta, \beta_2 \in [n_t]^*$.
  - Pre-terminal rules ($\mathcal{R}_0$): $a(h_1) \rightarrow \langle \alpha, \beta \rangle$ where $a \in \mathcal{N}$, $\alpha \in [n_t]^*$ and $\beta \in [n_s]^*$.

  Each of these rules is associated with a probability $t(a(h_1) \rightarrow \gamma | a, h_1)$ where γ is the right-hand side (RHS) of the rule, and each rule probability is in the form of a conditional distribution, conditioned on the LHS (NT category and hidden state).
- For $a \in \mathcal{N}$, $h \in [m]$, $\pi(a, h)$ is a parameter specifying the root probability of $a(h)$.

A skeletal tree (s-tree) for a sentence is the set of rules $r_1, \ldots, r_N$ in the synchronous derivation of that sentence, where each node $i \in [1, N]$ in the tree has an NT category and represents a rule production without any additional latent state information or decoration. A full tree consists of an s-tree $r_1, \ldots, r_N$ together with values $h_1, \ldots, h_N$ for every NT or node in the tree. We can compute the probability mass function (PMF) over full trees:

$$p(r_1, \ldots, r_N, h_1, \ldots, h_N) = \pi(a_1, h_1) \times \prod_{i=2}^{N} t(a_i(h_i) \rightarrow \gamma | a_i, h_i) \tag{3.1}$$

where $i = 1$ references the root node, and the RHS γ is either a function of two hidden states ($r_i \in \mathcal{R}_2$), one hidden state ($r_i \in \mathcal{R}_1$), or no hidden states ($r_i \in \mathcal{R}_0$). Correspondingly, the PMF over s-trees is $p(r_1, \ldots, r_N) = \sum_{h_1, \ldots, h_N} p(r_1, \ldots, r_N, h_1, \ldots, h_N)$.

In our instantiation of this model, we refine the single-category "Hiero" grammar introduced by Chiang (2007) for HPBT in order to learn additional latent NT categories. In HPBT, the set $\mathcal{N}$ consists of only two symbols: X, and a goal symbol S for the LHS of the root node. Thus, the following discussion is restricted to these kinds of grammars, although the method is equally applicable in other scenarios, e.g., the extended tree-to-string transducer (**xRs**) formalism (Huang et al., 2006, Graehl et al., 2008) commonly used in syntax-directed translation, and phrase-based MT (Koehn et al., 2003). The formalism can also naturally handle rules with more than two NTs in the RHS, but for tractability reasons we make use of at most binary rules $\mathcal{R}_2$. This decision also simplifies the subsequent exposition.

3.2 Marginal Inference with L-SCFGs.

Inference with L-SCFGs involves two steps: the first, which is identical to the algorithm for standard SCFGs (and thus does not require any hidden state information), is to obtain the set of trees (i.e., the **forest**) that can derive or generate a given input sentence. Formally, this operation can be described as the composition of a synchronous grammar with a weighted finite-state transducer (WFST) representation of the input sentence, and the result is an SCFG, compactly represented as a hypergraph (Dyer, 2010). For the second step, we introduce a novel tensor-based inside-outside algorithm that is similar to the one proposed in Cohen et al. (2014), but adapted to hypergraph parse forests.

3.2.1 Computing the Parse Forest

To compute the parse forest, we rely on a bottom-up dynamic programming algorithm with Earley-style rules, through which we can obtain a hypergraph representation (Klein and Manning, 2001) of the parse forest for a source language sentence. Hypergraphs are a compact way to represent a forest of multiple parse trees. Each node in the hypergraph corresponds to an NT span, and can have multiple incoming and outgoing hyperedges. Hyperedges, which connect one or more tail nodes to a single head node, correspond exactly to rules, and tail or head nodes correspond to children (RHS NTs) or parent (LHS NT).

Broadly speaking, the algorithm is based on a variant of the CKY algorithm that handles non-Chomsky normal form grammars (Chappelier and Rajman, 1998), suitably modified for translation (Chiang, 2007). The basic idea is to maintain two charts: one corresponding to active items, namely rules that have been incompletely applied to that particular span, and passive items, namely rules that have been completed. For a presentation of this algorithm as a weighted logic program, please see §2.3.2.2 in Dyer (2010).

3.2.2 Tensor Inside-Outside Algorithm

In this section, we present an inside-outside algorithm that handles the hidden states in an L-SCFG by marginalizing or summing over these states during inference. We first introduce the appropriate structures and machinery to compute with these hidden states (tensors), along with a running example to show correctness of computation. Then, we show that tree probabilities (and by extension, marginal terms for subspans of sentences) are invariant to linear transformations of the hidden state parameters. This result is crucial, because it allows us to estimate hidden state parameters (specifically, linear transforms of them) using a singular value decomposition of observable moments (§3.3.1). We finish by presenting pseudocode for the algorithm and a brief discussion of its computational properties.

Tensor Structures

For a parameter t of rule r, the latent state h_1 attached to the LHS NT of r is associated with the outside tree for the sub-tree rooted at the LHS, and the states attached to the RHS NTs are associated with the inside trees of that NT. Since we do not assume conditional independence of these states, we need to consider all possible interactions, which can be compactly represented as a 3rd-order tensor in the case of a binary rule, a matrix (i.e., a 2nd-order tensor) for unary rules, and a vector for pre-terminal (lexical) rules. Preferences for certain outside-inside tree combinations are reflected in the values contained in these tensor structures. In this manner, we intend to capture interactions between non-local context of a phrase, which can typically be represented via features defined over outside trees of the node spanning the phrase, and the interior context, correspondingly defined via features over the inside trees. We refer to these tensor structures collectively as T^r for rules $r \in \mathcal{R}$, and they encompass the parameters t, which are conditional probabilities conditioned on the LHS NT category and state (Eq. 3.1).

For $r \in \mathcal{R}_0 : T^r \in \mathbb{R}^{m \times 1}$; similarly for $r \in \mathcal{R}_1 : T^r \in \mathbb{R}^{m \times m}$ and $r \in \mathcal{R}_2 : T^r \in \mathbb{R}^{m \times m \times m}$. We also maintain a vector $T^{\text{S}} \in \mathbb{R}^{1 \times m}$ corresponding to the parameters $\pi(\text{S}, h)$ for the goal node (root). Each entry in these structures corresponds to the probability of a full rule; for example, $T^r_{h_1,h_2,h_3} = t(r, h_2, h_3 | a, h_1)$ for $r \in \mathcal{R}_2$, and similarly for $r \in \mathcal{R}_1$ and $r \in \mathcal{R}_0$. As a result, for $r \in \mathcal{R}_0$ the parameter values in the associated vector T^r are greater than or equal to zero and sum to one (along with the entries in the vector T^{S}); the same can be said for each row $T^r_{h_1,*}$ in the matrices T^r for $r \in \mathcal{R}_1$, as well as each matrix $T^r_{h_1,*,*}$ in the tensor T^r for $r \in \mathcal{R}_2$. These parameters participate in tensor-vector operations: a 3rd-order tensor $T^r, r \in \mathcal{R}_2$ can be multiplied along each of its three modes ($\times_0, \times_1, \times_2$), and if multiplied by an $m \times 1$ vector, will produce an $m \times m$ matrix.[2] Note that matrix multiplication can be represented by $\times_1$ when multiplying on the right and $\times_0$ when multiplying on the left of the matrix.

[2] This operation is also called a contraction.

It is straightforward to show that these tensor structures correctly compute the probability of a tree. To show this computation, we introduce a simple example of a synchronous tree generated from a "Hiero" grammar in Fig. 3.1, along with the skeletal rules that generated the tree. The nodes in the tree are numbered, with the corresponding rules presented below. Within a rule, the superscripts on NTs simply indicate which NTs are aligned across source and target languages.

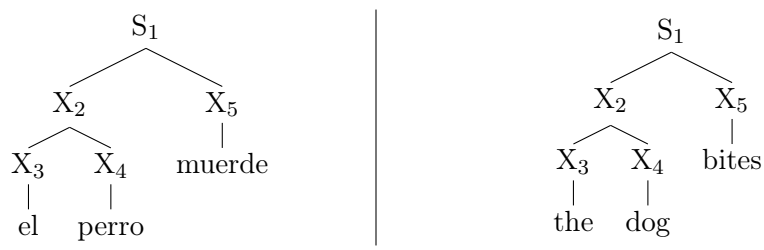


Figure 3.1: A simple synchronous tree example, consisting of the following rules:
$r_1 : \mathrm{S} \rightarrow \mathrm{X}^1\mathrm{X}^2 \ ||| \ \mathrm{X}^1\mathrm{X}^2$ $\quad r_2 : \mathrm{X} \rightarrow \mathrm{X}^1\mathrm{X}^2 \ ||| \ \mathrm{X}^1\mathrm{X}^2$ $\quad r_3 : \mathrm{X} \rightarrow \mathrm{el} \ ||| \ \mathrm{the}$
$r_4 : \mathrm{X} \rightarrow \mathrm{perro} \ ||| \ \mathrm{dog}$ $\quad r_5 : \mathrm{X} \rightarrow \mathrm{muerde} \ ||| \ \mathrm{bites}$

Each rule r_i has an associated parameter T^{r_i} that represents the probability distribution over latent states attached to RHS NT categories, conditional on the LHS category and latent state. Consider the term $T^{r_2} \times_1 T^{r_3} \times_2 T^{r_4}$. The result is an m-dimensional column vector, which we call b^2. By the definition of the tensor T^{r_2}, we have

$$\begin{aligned} b^2_h &= [T^{r_2} \times_1 T^{r_3} \times_2 T^{r_4}]_h \\ &= \sum_{h_2,h_3} t(\mathrm{X} \rightarrow \mathrm{X}^1\mathrm{X}^2 ||| \mathrm{X}^1\mathrm{X}^2, h_2, h_3 | h, \mathrm{X}) \times t(\mathrm{X} \rightarrow \mathrm{el} \ ||| \ \mathrm{the} | h_2, \mathrm{X}) \times t(\mathrm{X} \rightarrow \mathrm{perro} \ ||| \ \mathrm{dog} | h_3, \mathrm{X}) \end{aligned}$$

Similarly, $T^{r_1} \times_1 (T^{r_2} \times_1 T^{r_3} \times_2 T^{r_4}) \times_2 T^{r_5}$, which we call b^1, is

$$b^1_h = \sum_{h_2,h_3} t(\mathrm{S} \rightarrow \mathrm{X}^1\mathrm{X}^2 ||| \mathrm{X}^1\mathrm{X}^2, h_2, h_3 | h, \mathrm{X}) \times b^2_{h_2} \times t(\mathrm{X} \rightarrow \mathrm{muerde} \ ||| \ \mathrm{bites} | h_3, \mathrm{X})$$

And finally, the probability of the full tree is:

$$\sum_h b^1_h \pi(\mathrm{S}, h) = b^1 \times_0 T^{\mathrm{S}}$$

These are precisely the calculations used in the conventional CKY dynamic programming algorithm for computing tree probabilities.

Linear Transformations of Parameters

Assuming each entry in these structures represents the probability of a rule, replacing scalar multiplication with tensor-vector products in the inside-outside dynamic programming step should correctly compute the probability of the marginal terms $\mu(\mathrm{X}, i, j)$ (the marginal probability of X dominating the words indexed from i to j), and therefore the probability of an s-tree $p(r_1, \ldots, r_N)$.[3] However, Theorem 3 in Cohen et al. (2014) shows that it is sufficient, for the purposes of s-tree probability and marginal terms computation, to have parameters that are equal to the true parameters *up to a linear transform.* We restate the theorem below (with suitable generalizations from CFGs to SCFGs):

Theorem 1. *Assume that we have an L-SCFG with parameters T^r, and that there exist matrices $G^X \in \mathbb{R}^{m \times m}$ and $G^S \in \mathbb{R}^{m \times m}$ such that G^X and G^S are invertible, C^r is defined such that:*

1. *for $r \in \mathcal{R}_2$ with LHS NT X, $C^r \times_2 \mathbf{y}_2 \times_1 \mathbf{y}_1 = (G^X)^{-1}(T^r \times_2 (G^X \mathbf{y}_2) \times_1 (G^X \mathbf{y}_1))$*
2. *for $r \in \mathcal{R}_2$ with LHS NT S, $C^r \times_2 \mathbf{y}_2 \times_1 \mathbf{y}_1 = (G^S)^{-1}(T^r \times_2 (G^X \mathbf{y}_2) \times_1 (G^X \mathbf{y}_1))$*
3. *for $r \in \mathcal{R}_1$ with LHS NT X, $C^r \times_1 \mathbf{y}_1 = (G^X)^{-1}(T^r \times_1 (G^X \mathbf{y}_1))$*
4. *for $r \in \mathcal{R}_1$ with LHS NT S, $C^r \times_1 \mathbf{y}_1 = (G^S)^{-1}(T^r \times_1 (G^X \mathbf{y}_1))$*
5. *for $r \in \mathcal{R}_0$, $C^r = (G^X)^{-1} T^r$*
6. *$C^S = T^S G^S$*

for vectors $\mathbf{y}_1, \mathbf{y}_2 \in \mathbb{R}^{m \times 1}$. Then the algorithm in Fig. 3.2 correctly computes the marginal probabilities $\mu(X, i, j)$ under the L-SCFG over the parse forest and the sentence probability $\mu(S, 1, N)$ under the L-SCFG.

The proof of this theorem when the forest consists of a single tree is in Cohen et al. (2014), but we provide some intuition by walking through a sample inside computation below. The idea is relatively simple: take your existing parameters T^r, and transform them by a random linear transformation, such that the linear transformation will cancel out when marginals or other quantities from the dynamic programming algorithm are computed. Each rule r_i has an associated parameter C^r, which are the true parameters T^r but with an unknown linear transformation applied. Our aim is to show, under the assumptions in the theorem, that the marginal probability of the tree $p(r_1, \ldots, r_N)$ computed with parameters C^r is equivalent to using the true parameters T^r i.e., we show that the terms associated with the unknown linear transform cancel. While we only detail the computation of the overall probability of the tree, it is very similar to the computation of marginal terms $\mu(a, i, j), a \in \mathcal{N}$ for subspans $[i, j)$.

[3] Or the probability of a sentence over the forest of s-trees, as the hypergraph version of the algorithm computes.

Inputs: Sentence $f_1 \dots f_N$, L-SCFG $(\mathcal{N}, \mathrm{S}, m, n)$, parameters $C^r \in \mathbb{R}^{(m \times m \times m)}$, $\in \mathbb{R}^{(m \times m)}$, or $\in \mathbb{R}^{(m \times 1)}$ for all $r \in \mathcal{R}$, $C^{\mathrm{S}} \in \mathbb{R}^{(1 \times m)}$, hypergraph $\mathcal{H}$.

Data structures:

For each node $q \in \mathcal{H}$:

- $\boldsymbol{\alpha}(q) \in \mathbb{R}^{m \times 1}$ is a column vector of inside terms.
- $\boldsymbol{\beta}(q) \in \mathbb{R}^{1 \times m}$ is a row vector of outside terms.
- For each incoming edge $e \in \mathbf{B}(q)$ to node q, $\mu(e)$ is a marginal probability for edge (rule) e.

Algorithm:

▷ *Inside Computation*

For nodes q in topological order in $\mathcal{H}$,

  $\boldsymbol{\alpha}(q) = \mathbf{0}$

  For each incoming edge $e \in \mathbf{B}(q)$,

    tail $= \mathbf{t}$(e), rule $= \mathbf{r}$(e)

    if $|\text{tail}| = 0$, then $\boldsymbol{\alpha}(q) = \boldsymbol{\alpha}(q) + C^{\text{rule}}$

    else if $|\text{tail}| = 1$, then $\boldsymbol{\alpha}(q) = \boldsymbol{\alpha}(q) + C^{\text{rule}} \times_1 \boldsymbol{\alpha}(\text{tail}_0)$

    else if $|\text{tail}| = 2$, then $\boldsymbol{\alpha}(q) = \boldsymbol{\alpha}(q) + C^{\text{rule}} \times_2 \boldsymbol{\alpha}(\text{tail}_1) \times_1 \boldsymbol{\alpha}(\text{tail}_0)$

▷ *Outside Computation*

For $q \in \mathcal{H}$,

  $\boldsymbol{\beta}(q) = \mathbf{0}$

$\boldsymbol{\beta}(\text{goal}) = C^{\mathrm{S}}$

For q in reverse topological order in $\mathcal{H}$,

  For each incoming edge $e \in \mathbf{B}(q)$,

    tail $= \mathbf{t}$(e), rule $= \mathbf{r}$(e)

    if $|\text{tail}| = 1$, then

      $\boldsymbol{\beta}(\text{tail}_0) = \boldsymbol{\beta}(\text{tail}_0) + \boldsymbol{\beta}(q) \times_0 C^{\text{rule}}$

    else if $|\text{tail}| = 2$, then

      $\boldsymbol{\beta}(\text{tail}_0) = \boldsymbol{\beta}(\text{tail}_0) + \boldsymbol{\beta}(q) \times_0 C^{\text{rule}} \times_2 \boldsymbol{\alpha}(\text{tail}_1)$

      $\boldsymbol{\beta}(\text{tail}_1) = \boldsymbol{\beta}(\text{tail}_1) + \boldsymbol{\beta}(q) \times_0 C^{\text{rule}} \times_1 \boldsymbol{\alpha}(\text{tail}_0)$

▷*Edge Marginals*

Sentence probability $g = \boldsymbol{\alpha}(\text{goal}) \times \boldsymbol{\beta}(\text{goal})$

For edge $e \in \mathcal{H}$,

  head $= \mathbf{h}$(e), tail $= \mathbf{t}$(e), rule $= \mathbf{r}$(e)

  if $|\text{tail}| = 0$, then $\mu(e) = (\boldsymbol{\beta}(\text{head}) \times_0 C^{\text{rule}})/g$

  else if $|\text{tail}| = 1$, then $\mu(e) = (\boldsymbol{\beta}(\text{head}) \times_0 C^{\text{rule}} \times_1 \boldsymbol{\alpha}(\text{tail}_0))/g$

  else if $|\text{tail}| = 2$, then $\mu(e) = (\boldsymbol{\beta}(\text{head}) \times_0 C^{\text{rule}} \times_2 \boldsymbol{\alpha}(\text{tail}_1) \times_1 \boldsymbol{\alpha}(\text{tail}_0))/g$

Figure 3.2: The tensor form of the hypergraph inside-outside algorithm, for calculation of rule marginals $\mu(e)$. A slight simplification in the marginal computation yields NT marginals for spans $\mu(\mathrm{X}, i, j)$. $\mathbf{B}$(q) returns the incoming hyperedges for node q, and $\mathbf{h}$(e), $\mathbf{t}$(e), $\mathbf{r}$(e) return the head node, tail nodes, and rule for hyperedge e.

Using the parameters C^r, the probability of the tree is:

$$(C^{r_1} \times_1 (C^{r_2} \times_1 C^{r_3} \times_2 C^{r_4}) \times_2 C^{r_5}) \times_0 C^{\mathrm{S}}$$

The expression mirrors the structure of the tree: since $r_3, r_4, r_5 \in \mathcal{R}_0$, their corresponding parameters are m-dimensional column vectors, and since $r_1, r_2 \in \mathcal{R}_2$, their corresponding parameters are $m \times m \times m$ tensors. The last term corresponds to the root probability. Next,

we define an equivalence between the C^r and T^r parameters, starting with the axioms:

$$C^{r_3} = (G^{\mathrm{X}})^{-1} T^{r_3} \qquad C^{r_4} = (G^{\mathrm{X}})^{-1} T^{r_4} \qquad C^{r_5} = (G^{\mathrm{X}})^{-1} T^{r_5}$$

Using the first assumption of the theorem, we can write:

$$\begin{aligned} C^{r_2} &= (G^{\mathrm{X}})^{-1} (T^{r_2} \times_1 (G^{\mathrm{X}} C^{r_3}) \times_2 (G^{\mathrm{X}} C^{r_4})) \\ &= (G^{\mathrm{X}})^{-1} (T^{r_2} \times_1 (G^{\mathrm{X}} (G^{\mathrm{X}})^{-1} T^{r_3}) \times_2 (G^{\mathrm{X}} (G^{\mathrm{X}})^{-1} T^{r_4})) \quad = (G^{\mathrm{X}})^{-1} (T^{r_2} \times_1 T^{r_3} \times_2 T^{r_4}) \end{aligned}$$

The last equivalence $C^{r_1} = (G^{\mathrm{S}})^{-1} (T^{r_1} \times_1 T^{r_2} \times_2 T^{r_5})$ can be derived similarly. Finally, computing the overall probability of the sentence requires a multiplication on the left ($\times_0$) by the root probability parameter C^{S}, which is where the G^{S} terms cancel.

Algorithm

Figure 3.2 presents the tensor version of the inside-outside algorithm for L-SCFG inference. The algorithm computes inside and outside probabilities over the hypergraph using the tensor representations, and converts these probabilities to marginal rule probabilities. These marginals are computed by summing over the latent states, which in practice corresponds to simple tensor-vector products. Note that we provide linearly transformed parameters C^r as inputs, a strict generalization over operating on probabilities T^r.

The complexity of this decoding algorithm is $\mathcal{O}(n^3 m^3 |G|)$ where n is the length of the input sentence, m is the number of latent states, and $|G|$ is the number of production rules in the grammar *without* latent-variable annotations (i.e., $m = 1$).[4] The bulk of the computation is a series of tensor-vector products of relatively small size (each dimension is of length m), which can be computed very quickly and in parallel. The tensor computations can be significantly sped up using techniques described by Cohen and Collins (2012), so that they are linear in m and not cubic.

3.3 Parameter Estimation for L-SCFGs

To estimate the parameters of an L-SCFG, we assume the existence of a dataset composed of synchronous s-trees, which can be acquired from word alignments. Normally in phrase-based translation models, we consider all possible phrase pairs consistent with the word alignments and estimate features based on surface statistics associated with the phrase pairs or rules. The weights of these features are then learned using a discriminative training algorithm (Och,

[4] In practice, the term $m^3|G|$ can be replaced with a smaller term, which separates the rules in G by the number of NTs on the RHS. This idea relates to the notion of "effective grammar size" which we discuss in §3.4.

2003, Chiang, 2012, *inter alia*). In contrast, in this work we restrict the number of possible synchronous derivations for each sentence pair to just one; thus, derivation forests do not have to be considered, making parameter estimation more tractable.[5] To achieve this objective, for each sentence in the training data we extract the **minimal** set of synchronous rules consistent with the word alignments (§2.1.3). By using minimal rules as a starting point instead of the traditional heuristically-extracted rules (Chiang, 2007) or arbitrary compositions of minimal rules (Galley et al., 2006), we are also able to explore the transition from minimal rules to composed ones in a principled manner by encoding contextual information through the latent states.

We explore two methods for estimating the parameters C^r of the model: a likelihood-maximization approach based on EM (Dempster et al., 1977), and a spectral approach based on the method of moments (Hsu et al., 2009, Cohen et al., 2014), where we identify a subspace using a singular value decomposition (SVD) of the cross-product feature space between inside and outside trees and estimate parameters in this subspace. In the spectral approach, we base our parameter estimates on low-rank representations of moments of features, while EM explicitly maximizes a likelihood criterion. The parameter estimation algorithms are relatively similar, but in lieu of sparse feature functions in the spectral case, EM uses partial counts estimated with the current set of parameters. The nature of EM allows it to be susceptible to local optima, while the spectral approach comes with guarantees on obtaining the global optimum (Cohen and Collins, 2014). Lastly, computing the SVD and estimating parameters in the low-rank space is a one-shot operation, as opposed to the iterative procedure of EM, and therefore is much more computationally efficient.

3.3.1 Estimation with Spectral Method

We generalize the parameter estimation algorithm presented in Cohen et al. (2014) to the synchronous or bilingual case. The central concept of the spectral parameter estimation algorithm is to learn an m-dimensional representation of inside and outside trees by defining these trees in terms of features, in combination with a projection step (SVD), with the hope being that the lower-dimensional space captures the syntactic and semantic regularities among rules from the sparse feature space. Every NT in an s-tree has an associated inside and outside tree; the inside tree contains the entire sub-tree at and below the NT, and the outside tree is everything else in the synchronous s-tree except the inside tree. The inside feature function $\boldsymbol{\phi}$ maps the domain of inside tree fragments to a d-dimensional Euclidean space, and the outside feature function $\boldsymbol{\psi}$ maps the domain of outside tree fragments to a d'-dimensional space. The spectral estimator is based on extracting latent-variable parameters from these **observable forms** (Jaeger, 2000), and a key result that we build upon is that it is possible to directly

[5] For future work, we will consider efficient algorithms for parameter estimation over derivation forests, since there may be multiple valid ways to explain the sentence pair via a synchronous tree structure.

estimate our latent-variable parameters from a training set consisting of s-trees.

Observable Operators

The main implication of Theorem 1 is that the linear transformations add an extra degree of freedom during parameter estimation, which crucially allows us to use observable forms and the SVD to estimate these linearly transformed parameters. Below, we explain how these parameters can be estimated.

We sample a full tree $r_1, \ldots, r_N, h_1, \ldots, h_N$ from the joint distribution $p(r_1, \ldots, r_N, h_1, \ldots, h_N)$. Our random variables are defined as follows:

- R_i is the rule r_i
- T_1 is the inside tree rooted at node i; for $r_i \in \mathcal{R}_2$, T_2 is the inside tree rooted at the left child of node i, and T_3 is the inside tree rooted at the right child of node i. For $r_i \in \mathcal{R}_1$ there is only a single inside tree random variable T_2, and for $r_i \in \mathcal{R}_0$, there are no children, and hence no child random variables T_2 and T_3.
- H_1, H_2, and H_3 are the corresponding latent variables associated with node i and its children.
- A_1, A_2, and A_3 are the NT categories associated with the node i and its children; in HPBT, these can only be X or S, with the added restriction that only A_1 (LHS NT) can take the value S. For subsequent discussion, we assume $a_1, a_2, a_3 = \text{X}$ unless otherwise noted, and thus drop the NT category references unless referring to S.
- O is the outside tree at node i.
- B is equal to 1 if node i is at the root of the tree, 0 otherwise.

Let $\boldsymbol{\phi}(t) \in \mathbb{R}^{d \times 1}$, $\boldsymbol{\psi}(o) \in \mathbb{R}^{d' \times 1}$ be the inside and outside feature functions for inside tree t and outside tree o. We also assume the existence of projection matrices $U \in \mathbb{R}^{d \times m}$ and $V \in \mathbb{R}^{d' \times m}$. With these matrices, we can define additional random variables $Y_1, Y_2, Y_3, Z \in \mathbb{R}^{m \times 1}$ as:

$$Y_1 = U^T \boldsymbol{\phi}(T_1) \quad Z = V^T \boldsymbol{\psi}(O)$$

$$Y_2 = U^T \boldsymbol{\phi}(T_2) \quad Y_3 = U^T \boldsymbol{\phi}(T_3)$$

With these random variables, we can define the following quantities in expectation:

$$\begin{aligned}
\Sigma &= \mathbb{E}[Z \otimes Y_1] \\
D^r &= \mathbb{E}\left[[[R_1 = r]] Z \otimes Y_2 \otimes Y_3\right] \quad \text{if } r \in \mathcal{R}_2 \\
D^r &= \mathbb{E}\left[[[R_1 = r]] Z \otimes Y_2\right] \quad \text{if } r \in \mathcal{R}_1 \\
D^r &= \mathbb{E}\left[[[R_1 = r]] Z\right] \quad \text{if } r \in \mathcal{R}_0
\end{aligned}$$

where the $[[R_1 = r]]$ notation is an indicator function which is 1 when $R_1 = r$ and 0 otherwise. As long as we have access to functions $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ and projection matrices U and V, these quantities can be estimated directly from training data consisting of a set of s-trees, one for every sentence pair. We can use the training examples to derive i.i.d. samples from the joint distribution over the random variables $(A_1, R_1, Y_1, Y_2, Y_3, Z, B)$ used in the definition of D^r. In terms of these expectations, our observable representations can be written as:

$$\begin{aligned}
C^r &= D^r \Sigma^{-1} \\
C^{\mathrm{S}} &= \mathbb{E}\left[[[A_1 = \mathrm{S}]] Y_1 | B = 1\right]
\end{aligned} \tag{3.2}$$

These quantities will satisfy the conditions of Theorem 1 under certain conditions. First, define the matrices $I \in \mathbb{R}^{d \times m}$ and $J \in \mathbb{R}^{d' \times m}$ as:

$$\begin{aligned}
[I]_{i,h} &= \mathbb{E}[\phi_i(T_1) | H_1 = h] \\
[J]_{i,h} &= \mathbb{E}[\psi_i(O) | H_1 = h]
\end{aligned}$$

We also define a column vector $\boldsymbol{\gamma} \in \mathbb{R}^{m \times 1}$ to denote the distribution of LHS latent states i.e., $\gamma_h = P(H_1 = h)$. The correctness of the observable representations in Eq. 3.2 depends on several conditions being satisfied (similar to Hsu et al. (2009)):

Theorem 2. *Assume that the following conditions are satisfied:*

1. *I and J have rank m i.e., they are full rank.*

2. *$\forall h \in [m]$, $\gamma_h > 0$.*

3. *The matrices $U, U \in \mathbb{R}^{d \times m}$ and $V^X, V \in \mathbb{R}^{d' \times m}$ are such that the matrices $G^X = U^T I$ and $K^X = V^T J$ (correspondingly, G^S and K^S) are invertible.*

Then, defining G^X and G^S in Theorem 1 as $G^X = (U^X)^T I^X$ and $G^S = (U^S)^T I^S$, the observable representations of Eq. 3.2 satisfy the equalities in Theorem 1 and can be used to compute marginal terms and s-tree/sentence probabilities.

Proof. Assume the following identities hold:

$$
\begin{aligned}
r \in \mathcal{R}_2 : D^r &= \text{diag}(\boldsymbol{\gamma})(K^{\text{X}})^T(T^r \times_2 (G^{\text{X}}\mathbf{y}_2) \times_1 (G^{\text{X}}\mathbf{y}_1)) \\
r \in \mathcal{R}_1 : D^r &= \text{diag}(\boldsymbol{\gamma})(K^{\text{X}})^T(T^r \times_1 (G^{\text{X}}\mathbf{y}_1)) \\
r \in \mathcal{R}_0 : D^r &= \text{diag}(\boldsymbol{\gamma})(K^{\text{X}})^T T^r \\
\Sigma &= G^{\text{X}}\text{diag}(\boldsymbol{\gamma})(K^{\text{X}})^T \\
C^{\text{S}} &= T^{\text{S}} G^{\text{S}}
\end{aligned}
$$

Under the conditions of the theorem, Σ is invertible, and therefore $\Sigma^{-1} = ((K^{\text{X}})^T)^{-1}(\text{diag}(\boldsymbol{\gamma}))^{-1}(G^{\text{X}})^{-1}$. Combining this information with Eq. 3.2 and the indentities above, the theorem follows. Therefore, in order to prove the theorem we need to show that the identities above hold. We show $D^r = \text{diag}(\boldsymbol{\gamma}^{\text{X}})(K^{\text{X}})^T T^r$ for $r \in \mathcal{R}_0$; the other identities can be derived similarly.

By definition, $D^r = \mathbb{E}\left[[[R_1 = r]]Z\right]$, or equivalently

$$
\begin{aligned}
D_i^r &= \mathbb{E}\left[[[R_1 = r]]Z_i\right] \\
&= \sum_h p(r,h)\mathbb{E}[Z_i | H_1 = h, R_1 = r] \\
&= \sum_h p(h)p(r|h)\mathbb{E}[Z_i | H_1 = h, R_1 = r] \qquad (3.3) \\
&= \sum_h \gamma_h T_h^r \mathbb{E}[Z_i | H_1 = h, R_1 = r] \\
&= \sum_h \gamma_h T_h^r \mathbb{E}[Z_i | H_1 = h] \qquad (3.4) \\
&= \sum_h \gamma_h T_h^r K_{i,h}^{\text{X}}
\end{aligned}
$$

where in line 3.3 we use the chain rule, and in line 3.4 we use the independence assumptions of the L-SCFG (the outside tree is conditionally independent of the rule given the LHS NT and the hidden state) and the definition of K. The identity $D^r = \text{diag}(\boldsymbol{\gamma})(K^{\text{X}})^T T^r$ for $r \in \mathcal{R}_0$ follows. □

Empirical Estimates

The previous theorem allows us to estimat latent-variable parameters from observable quantities. The following lemma motivates and justifies the use of SVD for finding values U and V that satisfy the second condition of the theorem (where we assume the first holds):

Lemma 1. *Assume that the first two conditions of Theorem 2 hold, and define the inside-*

outside feature covariance matrix as:

$$\Omega = \mathbb{E}\left[\boldsymbol{\phi}(T_1) \otimes \boldsymbol{\psi}(O)\right] \tag{3.5}$$

Then if U is a matrix of the m left singular vectors of Ω corresponding to non-zero singular values, and V is a matrix of the m right singular vectors of Ω corresponding to non-zero singular values, the third condition in Theorem 2 is satisfied.

This lemma can be proved by showing $\Omega = I\text{diag}(\boldsymbol{\gamma})J^T$; the remainder of the proof is very similar to lemma 2 in Hsu et al. (2009).

We can estimate the matrix Ω directly from a training set of s-trees. Let $\mathcal{O}$ be the set of all tuples of inside-outside trees in our training corpus, whose size is equivalent to the number of rule tokens (occurrences in the corpus) M, and $\boldsymbol{\phi}(t), \boldsymbol{\psi}(o)$ be defined as before. By computing the outer product $\otimes$ between the inside and outside feature vectors for each pair and aggregating, we obtain the *empirical* inside-outside feature covariance matrix:

$$\hat{\Omega} = \frac{1}{|\mathcal{O}|} \sum_{(o,t)\in\mathcal{O}} \boldsymbol{\phi}(t)\left(\boldsymbol{\psi}(o)\right)^\top \tag{3.6}$$

If m is the desired latent space dimension, we compute an m-rank truncated SVD of the empirical covariance matrix $\hat{\Omega} \approx U\Sigma V^\top$, where $U \in \mathbb{R}^{d\times m}$ and $V \in \mathbb{R}^{d'\times m}$ are the matrices containing the left and right singular vectors, and $\Sigma \in \mathbb{R}^{m\times m}$ is a diagonal matrix containing the m-largest singular values along its diagonal. The techniques from Hsu et al. (2009) can be used to extend the results from the true covariance matrix Ω to the empirically-estimated matrix $\hat{\Omega}$.

Figure 3.3 provides the remaining steps in the algorithm. Note that for notational convenience, we have included the Σ^{-1} term in the projection matrix V for the outer tree features. The M training examples are obtained by considering all nodes in all of the synchronous s-trees given as input. In step 1, for each inside and outside tree, we project its high-dimensional representation to the m-dimensional latent space. Using the m-dimensional representations for inside and outside trees, in step 2 for each rule type r we compute the covariance between the inside tree vectors and the outside tree vector using the *tensor product*, a generalized outer product to compute covariances between more than two random vectors. For binary rules, with two child inside vectors and one outside vector, the result $\hat{E}^r$ is a 3-mode tensor; for unary rules, a regular matrix, and for pre-terminal rules with no right-hand side non-terminals, a vector. The final parameter estimate is then the associated tensor/matrix/vector, scaled by the maximum likelihood estimate of the rule r, as in step 3.

The corresponding theoretical guarantees from Cohen et al. (2014) can also be generalized to the synchronous case. $\hat{\Omega}$ is an empirical estimate of the true covariance matrix Ω, and if Ω has rank m, then the marginals computed using the spectrally-estimated parameters will converge

Inputs:
Training examples $(r^{(i)}, t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}, b^{(i)})$ for $i \in \{1 \dots M\}$, where $r^{(i)}$ is a context free rule; $t^{(i,1)}$, $t^{(i,2)}$, and $t^{(i,3)}$ are inside trees; $o^{(i)}$ is an outside tree; and $b^{(i)} = 1$ if the rule is at the root of tree, 0 otherwise. A function $\boldsymbol{\phi}$ that maps inside trees t to feature-vectors $\boldsymbol{\phi}(t) \in \mathbb{R}^{d \times 1}$. A function $\boldsymbol{\psi}$ that maps outside trees o to feature-vectors $\boldsymbol{\psi}(o) \in \mathbb{R}^{d' \times 1}$.
Algorithm:
▷ *Step 0: Singular Value Decomposition*

- Compute the SVD of Eq. 3.6 to calculate matrices $\hat{U} \in \mathbb{R}^{(d \times m)}$ and $\hat{V} \in \mathbb{R}^{(d' \times m)}$.

▷ *Step 1: Projection*

$$Y(t) = U^\top \boldsymbol{\phi}(t)$$
$$Z(o) = \Sigma^{-1} V^\top \boldsymbol{\psi}(o)$$

▷ *Step 2: Calculate Correlations*

$$\hat{E}^r = \begin{cases} \frac{\sum_{o \in Q^r} Z(o)}{|Q^r|} & \text{if } r \in \mathcal{R}_0 \\ \frac{\sum_{(o,t) \in Q^r} Z(o) \otimes Y(t)}{|Q^r|} & \text{if } r \in \mathcal{R}_1 \\ \frac{\sum_{(o,t^2,t^3) \in Q^r} Z(o) \otimes Y(t^2) \otimes Y(t^3)}{|Q^r|} & \text{if } r \in \mathcal{R}_2 \end{cases}$$

Q^r is the set of outside-inside tree triples for binary rules, outside-inside tree pairs for unary rules, and outside trees for pre-terminals. ▷ *Step 3: Compute Final Parameters*

- For all $r \in \mathcal{R}$,
  $\hat{C}^r = \frac{\text{count}(r)}{M} \times \hat{E}^r$
- For all $r^{(i)} \in \{1, \dots, M\}$ such that $b^{(i)}$ is 1,
  $\hat{C}^{\text{S}} = \hat{C}^{\text{S}} + \frac{Y(t^{(i,1)})}{|Q^{\text{S}}|}$

Q^{S} is the set of trees at the root.

Figure 3.3: The spectral learning algorithm for estimating parameters of an L-SCFG.

to the true marginals, with the sample complexity for convergence inversely proportional to a polynomial function of the mth largest singular value of Ω. In particular, Theorem 8 states a PAC-style theorem for the learning algorithm, which provides a lower bound on the number of training examples needed to obtain an empirical probability estimate for an s-tree $\hat{p}(r_1, \dots, r_N)$ within ϵ of the true probability of the s-tree $p(r_1, \dots, r_N)$ (and similarly for the marginal terms).

3.3.2 Estimation with EM

A likelihood maximization approach can also be used to learn the parameters of an L-SCFG. Parameters are initialized by sampling each parameter value $\hat{C}^r(h_1, h_2, h_3)$ from the interval

Inputs:
Training examples $(r^{(i)}, t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}, b^{(i)})$ for $i \in \{1 \dots M\}$, where $r^{(i)}$ is a context free rule; $t^{(i,1)}$, $t^{(i,2)}$, and $t^{(i,3)}$ are inside trees; $o^{(i)}$ is an outside tree; $b^{(i)} = 1$ if the rule is at the root of tree, 0 otherwise; and MAX_ITERATIONS.
Algorithm:
▷ *Step 0: Parameter Initialization*
For rule $r \in \mathcal{R}$,

- if $r \in \mathcal{R}_0$: initialize $\hat{C}^r \in \mathbb{R}^{m \times 1}$
- if $r \in \mathcal{R}_1$: initialize $\hat{C}^r \mathbb{R}^{m \times m}$
- if $r \in \mathcal{R}_2$: initialize $\hat{C}^r \mathbb{R}^{m \times m \times m}$

Initialize $\hat{C}^{\mathrm{S}} \in \mathbb{R}^{m \times 1}$
$\hat{C}^r_0 = \hat{C}^r, \hat{C}^{\mathrm{S}}_0 = \hat{C}^{\mathrm{S}}$
For iteration $t = 1, \dots,$ MAX_ITERATIONS,

- Expectation Step:

  ▷ *Estimate Y and Z*

  Compute partial counts and total tree probabilities g for all t and o using Fig. 3.2 and parameters $\hat{C}^r_{t-1}, \hat{C}^{\mathrm{S}}_{t-1}$.

  ▷ *Calculate Correlations*

$$\hat{E}^r = \begin{cases} \sum\limits_{o,g \in Q^r} \frac{Z(o)}{g} & \text{if } r \in \mathcal{R}_0 \\ \sum\limits_{(o,t,g) \in Q^r} \frac{Z(o) \otimes Y(t)}{g} & \text{if } r \in \mathcal{R}_1 \\ \sum\limits_{(o,t^2,t^3,g) \in Q^r} \frac{Z(o) \otimes Y(t^2) \otimes Y(t^3)}{g} & \text{if } r \in \mathcal{R}_2 \end{cases}$$

  ▷ *Update Parameters*

  For all $r \in \mathcal{R}$, $\hat{C}^r_t = \hat{C}^r_{t-1} \odot \hat{E}^r$

  For all $r^{(i)} \in \{1, \dots, M\}$ such that $b^{(i)}$ is 1, $\hat{C}^{\mathrm{S}}_t = \hat{C}^{\mathrm{S}}_t + (\hat{C}^{\mathrm{S}}_{t-1} \odot Y(r^{(i)}))/g$

  Q^{S} is the set of trees at the root.

- Maximization Step

  if $r \in \mathcal{R}_0$: $\forall h_1 : \hat{C}^r(h_1) = \frac{\hat{C}^r(h_1)}{\sum_{r'=r} \sum_{h_1} \hat{C}^{r'}(h_1)}$

  if $r \in \mathcal{R}_1$: $\forall h_1, h_2 : \hat{C}^r(h_1, h_2) = \frac{\hat{C}^r(h_1,h_2)}{\sum_{r'=r} \sum_{h_2} \hat{C}^{r'}(h_1,h_2)}$

  if $r \in \mathcal{R}_2$: $\forall h_1, h_2, h_3 : \hat{C}^r(h_1, h_2, h_3) = \frac{\hat{C}^r(h_1,h_2,h_3)}{\sum_{r'=r} \sum_{h_2,h_3} \hat{C}^{r'}(h_1,h_2,h_3)}$

  if LHS$(r) = $ S: $\forall h_1 : \hat{C}^r(h_1) = \frac{\hat{C}^r(h_1)}{\sum_{r'=r} \sum_{h_1} \hat{C}^{r'}(h_1)}$

Figure 3.4: The EM-based algorithm for estimating parameters of an L-SCFG.

$[0, 1]$ uniformly at random.[6] We first decode the training corpus using an existing set of parameters to compute the inside and outside probability vectors associated with NTs for every rule in each s-tree, constrained to the tree structure of the training example. These

[6] In our experiments, we also tried the initialization scheme described in Matsuzaki et al. (2005), but found that it provided little benefit.

probabilities can be computed using the decoding algorithm in Figure 3.2 (where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ correspond to the inside and outside probabilities respectively), except the parse forest consists of a single tree only. These vectors represent partial counts over latent states. We then define functions Y and Z (analogous to the spectral case) which map inside and outside tree instances to m-dimensional vectors containing these partial counts. In the spectral case, Y and Z are estimated just once, while in the case of EM they have to be re-estimated at each iteration.

The expectation step thus consists of computing the partial counts of inside and outside trees t and o, i.e., recovering the functions Y and Z, and updating parameters C^r by computing correlations, which involves summing over partial counts (across all occurrences of a rule in the corpus). Each partial count's contribution is divided by a normalization factor g, which is the total probability of the tree which t or o is part of. Note that unlike the spectral case, there is a specific normalization factor for each inside-outside tuple. Lastly, the correlations are scaled by the existing parameter estimates.

To obtain the next set of parameters, in the maximization step we normalize $\hat{C}^r$ for $r \in \mathcal{R}$ such that for every $h_1, \sum_{r'=r,h_2,h_3} \hat{C}^{r'}(h_1, h_2, h_3) = 1$ for $r \in \mathcal{R}_2$, $\sum_{r'=r,h_2} \hat{C}^{r'}(h_1, h_2) = 1$ for $r \in \mathcal{R}_1$, and $\sum_{r'=r,h_2} \hat{C}^{r'}(h_2) = 1$ for $r \in \mathcal{R}_0$. We also normalize the root rule parameters $\hat{C}^r$ where $\text{LHS}(r) = \text{S}$. It is also possible to add sparse, overlapping features to an EM-based estimation procedure (Berg-Kirkpatrick et al., 2010) and we leave this extension for future work.

3.4 Experiments

The goal of the experimental section is to evaluate the performance of the latent-variable SCFG in comparison to a baseline without any additional NT annotations (Min-Grammar), and to compare the performance of the two parameter estimation algorithms. We first present results from a synthetic grammar experiment to demonstrate the effic We also compare L-SCFGs to a Hiero baseline (Chiang, 2007). The language pair of evaluation is Chinese–English (ZH-EN).

We score translations using BLEU (Papineni et al., 2002). The latent-variable model is integrated into the standard MT pipeline by computing marginal probabilities for each rule in the parse forest of a source sentence using the algorithm in Figure 3.2 with the parameters estimated through the algorithms in Figures 3.4 and 3.3, and is added as a feature for the rule during MERT (Och, 2003). These probabilities are conditioned on the LHS (X), and are thus joint probabilities for a source-target RHS pair. We also write out as features the conditional relative frequencies $\hat{P}(e|f)$ and $\hat{P}(f|e)$ as estimated by our latent-variable model, i.e., conditioned on the source and target RHS.

Overall, we find that both the spectral and the EM-based estimators improve upon a minimal grammar baseline with only a single category, but the spectral approach does better. In fact, it matches the performance of the standard HIERO baseline, despite learning on top of a minimal grammar.

3.4.1 Data and Baselines

The ZH-EN data is the BTEC parallel corpus (Paul, 2009); we combine the first and second development sets in one, and evaluate on the third development set. The development and test sets are evaluated with 16 references. Statistics for the data are shown in Table 3.1. We used the `cdec` decoder (Dyer et al., 2010) to extract word alignments and the baseline hierarchical grammars, MERT tuning, and decoding. We used a 4-gram language model built from the target-side of the parallel training data. The Python-based implementation of the tensor-based decoder, as well as the parameter estimation algorithms is available at `http://www.github.com/asaluja/spectral-scfg/`.

	ZH-EN
TRAIN (SRC)	334K
TRAIN (TGT)	366K
DEV (SRC)	7K
DEV (TGT)	7.6K
TEST (SRC)	3.8K
TEST (TGT)	3.9K

Table 3.1: Corpus statistics (in words). For the target DEV and TEST statistics, we take the first reference (16 references total).

The baseline HIERO system uses a grammar extracted by applying the commonly used heuristics (Chiang, 2007). Each rule is decorated with two lexical and phrasal features corresponding to the forward $(e|f)$ and backward $(f|e)$ conditional log frequencies, along with the log joint frequency (e, f), the log frequency of the source phrase (f), and whether the phrase pair or the source phrase is a singleton. Weights for the language model (and language model OOV), glue rule, and word penalty are also tuned. The MIN-GRAMMAR baseline[7] maintains the same set of weights.

Grammar sizes are presented in Table 3.2. For the latent-variable models, we provide the effective grammar size, where the number of NTs on the RHS of a rule is taken into account when computing the grammar size, by assuming each possible latent variable configuration amongst the NTs generates a different rule. Furthermore, all singletons are mapped to the

[7] Code to extract the minimal derivation trees is available at `http://www.cs.rochester.edu/u/gildea/mt/`.

Grammar	Number of Rules
HIERO	1.69M
MIN-GRAMMAR	59K
LV $m = 1$	27.56K
LV $m = 8$	3.18M
LV $m = 16$	22.22M

Table 3.2: Grammar sizes for the different systems; for the latent-variable models, effective grammar sizes are provided.

OOV rule, while we include singletons in MIN-GRAMMAR.[8] Hence, effective grammar size can be computed as $m(1 + |\mathcal{R}_0^{>1}|) + m^2|\mathcal{R}_1| + m^3|\mathcal{R}_2|$, where $\mathcal{R}_0^{>1}$ is the set of pre-terminal rules that occur more than once.

3.4.2 Spectral Features

We use the following set of sparse, binary features in the spectral learning process:

- **Rule Indicator.** For the inside features, we consider the rule production containing the current non-terminal on the left-hand side, as well as the rules of the children (distinguishing between left and right children for binary rules). For the outside features, we consider the parent rule production along with the rule production of the sibling (if it exists).
- **Lexical.** for both the inside and outside features, any lexical items that appear in the rule productions are recorded. Furthermore, we consider the first and last words of spans (left and right child spans for inside features, distinguishing between the two if both exist, and sibling span for outside features). Source and target words are treated separately.
- **Length.** the span length of the tree and each of its children for inside features, and the span length of the parent and sibling for outside features.

In our experiments, we instantiated a total of 170,000 rule indicator features, 155,000 lexical features, and 80 length features.

3.4.3 Chinese–English Experiments

Table 3.3 presents a comprehensive evaluation of the ZH-EN experimental setup. The first section consists of the various baselines we consider. In addition to the aforementioned base-

[8] This OOV mapping is done so that the latent-variable model can handle unknown tokens.

	Setup	BLEU Dev	BLEU Test
Baselines	HIERO	46.08	55.31
	MIN-GRAMMAR	43.38	51.78
	MLE	43.24	52.80
Spectral	$m = 1$ RI	44.18	52.62
	$m = 8$ RI	44.60	53.63
	$m = 16$ RI	46.06	55.83
	m=16 RI+Lex+Sm	46.08	55.22
	m=16 RI+Lex+Len	45.70	55.29
	m=24 RI+Lex	43.00	51.28
	m=32 RI+Lex	43.06	52.16
EM	$m = 8$	40.53 (0.2)	49.78 (0.5)
	$m = 16$	42.85 (0.2)	52.93 (0.9)
	$m = 32$	41.07 (0.4)	49.95 (0.7)

Table 3.3: Results for the ZH-EN corpus, comparing across the baselines and the two parameter estimation techniques. RI, Lex, and Len correspond to the rule indicator, lexical, and length features respectively, and Sm denotes smoothing. For the EM experiments, we selected the best scoring iteration by tuning weights for parameters obtained after 25 iterations and evaluating other parameters with these weights. Results for EM are averaged over 5 starting points, with standard deviation given in parentheses. Spectral, EM, and MLE performances compared to the MIN-GRAMMAR baseline are statistically significant ($p < 0.01$).

lines, we evaluated a setup where the spectral parameters simply consist of the joint maximum likelihood estimates of the rules. This baseline should perform *en par* with MIN-GRAMMAR, which we see is the case on the development set. The performance on the test set is better though, primarily because we also include the reverse log relative frequency $(f|e)$ computed from the latent-variable model as an additional feature in MERT. Furthermore, in line with previous work (Galley et al., 2006) which compares minimal and composed rules, we find that minimal grammars take a hit of more than 2.5 BLEU points on the development set, compared to composed (HIERO) grammars. The $m = 1$ spectral baseline with only rule indicator features performs slightly better than the minimal grammar baseline, since it overtly takes into account inside-outside tree combination preferences in the parameters, but improvement is minimal with one latent state naturally and the performance on the test set is in line with the MLE baseline.

On top of the baselines, we looked at a number of feature combinations and latent states for the spectral and EM-estimated latent-variable models. For the spectral models, we tuned MERT parameters separately for each rank on a set of parameters estimated from rule indicator features only; subsequent variations within a given rank, e.g., the addition of lexical or length features or smoothing, were evaluated with the same set of rank-specific weights from MERT. For EM, we ran parameter estimation with 5 randomly initialized starting points for

50 iterations; we tuned the MERT parameters with EM parameters obtained after 25th iterations. Similar to the spectral experiments, we fixed the MERT weight values and evaluated BLEU performance with parameters after every 5 iterations and chose the iteration with the highest score on the development set. The results are averaged over the 5 initializations, with standard deviation in parentheses.

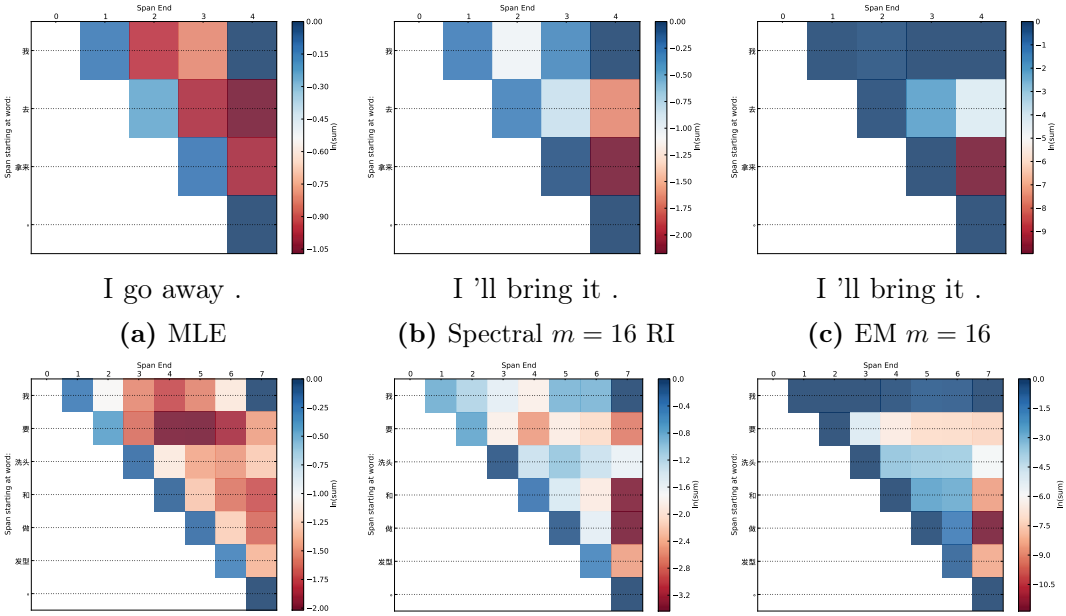
Firstly, we can see a clear dependence on rank, with peak performance for the spectral and EM models occurring at $m = 16$. In this instance, the spectral model roughly matches the performance of the HIERO baseline, but it only uses rules extracted from a minimal grammar, whose size is a fraction of the HIERO grammar. The gains seem to level off at this rank; additional ranks seem to add noise to the parameters. Feature-wise, additional lexical and length features add little, probably because much of this information is encapsulated in the rule indicator features. For EM, $m = 16$ outperforms the minimal grammar baseline, but is not at the level of the spectral results. All EM, spectral, and MLE results are statistically significant ($p < 0.01$) with respect to the MIN-GRAMMAR baseline (Zhang et al., 2004), and the improvement over the HIERO baseline achieved by the $m = 16$ rule indicator configuration is also statistically significant.

The two estimation algorithms differ significantly in their estimation time. Given a feature covariance matrix, the spectral algorithm (SVD, which was done with Matlab, and correlation computation steps) for $m = 16$ took 7 minutes, while the EM algorithm took 5 minutes for *each* iteration with this rank.

3.4.4 Analysis

Figure 3.5 presents a comparison of the non-terminal span marginals for two sentences in the development set. We visualize these differences through a heat map of the CKY parse chart, where the starting word of the span is on the rows, and the span end index is on the columns. Each cell is shaded to represent the marginal of that particular non-terminal span, with higher likelihoods in blue and lower likelihoods in red.

For the most part, marginals at the leaves (i.e., pre-terminal marginals) tend to score relatively similarly across different setups. Higher up in the chart, the latent SCFG marginals look quite different than the MLE parameters. Most noticeably, spans starting at the beginning of the sentence are much more favored. It is these rules that allow the right translation to be preferred since the MLE chooses not to place the object of the sentence in the subject's span. However, the spectral parameters seem to discriminate between these higher-level rules better than EM, which scores spans starting with the first word uniformly highly. Another interesting point is that the range of likelihoods is much larger in the EM case compared to the MLE and spectral variants. For the second sentence (row), the 1-best hypothesis produced by all systems are the same, but the heat map accentuates the previous observation.



I go away . | I 'll bring it . | I 'll bring it .

(a) MLE **(b)** Spectral $m = 16$ RI **(c)** EM $m = 16$

I 'd like a shampoo and style . | I 'd like a shampoo and style . | I 'd like a shampoo and style .

(d) MLE **(e)** Spectral $m = 16$ RI **(f)** EM $m = 16$

Figure 3.5: A comparison of the CKY charts containing marginal probabilities of non-terminal spans $\mu(\mathrm{X}, i, j)$ for the MLE, spectral $m = 16$ with rule indicator features, and EM $m = 16$, for the two Chinese sentences. Higher likelihoods are in blue, lower likelihoods in red. The hypotheses produced by each setup are below the heat maps.

3.5 Related Work

The goal of refining single-category HPBT grammars or automatically learning the NT categories in a grammar, instead of relying on noisy parser outputs, has been explored from several different angles in the MT literature. Blunsom et al. (2008a) present a Bayesian model for synchronous grammar induction, and place an appropriate nonparametric prior on the parameters. However, their starting point is to estimate a synchronous grammar with multiple categories from parallel data (using the word alignments as a prior), while we aim to refine a fixed grammar with additional latent states. Furthermore, their estimation procedure is extremely expensive and is restricted to learning up to five NT categories, via a series of mean-field approximations.

Another approach is to explicitly attach a real-valued vector to each NT: Huang et al. (2010) use an external source-language parser for this purpose and score rules based on the similarity between a source sentence parse and the information contained in this vector, which explicitly

requires the integration of a good-quality source-language parser. The EM-based algorithm that we propose here is similar to what they propose, except that we need to handle tensor structures. Mylonakis and Sima'an (2011) select among linguistically motivated non-terminal labels with a cross-validated version of EM. Although they consider a restricted hypothesis space, they do marginalize over different derivations therefore their inside-outside algorithm is $\mathcal{O}(n^6)$. In the syntax-directed translation literature, there have been efforts to relax or coarsen the hard labels provided by a syntactic parser in an automatic manner to promote parameter sharing (Venugopal et al., 2009, Hanneman and Lavie, 2013), which is the complement of our aim in this paper.

The idea of automatically learned grammar refinements comes from the monolingual parsing literature, where phenomena like head lexicalization can be modeled through latent variables. Matsuzaki et al. (2005) look at a likelihood-based method to split the NT categories of a grammar into a fixed number of sub-categories, while Petrov et al. (2006) learn a variable number of sub-categories per NT. The latter's extension may be useful for finding the optimal number of latent states from the data in our case.

The question of whether we can incorporate additional contextual information in minimal rule grammars in MT via auxiliary models instead of using longer, composed rules has been investigated before as well. n-gram translation models (Mariño et al., 2006, Durrani et al., 2011) seek to model long-distance contextual dependencies and reorderings through n-grams. The n-gram framework allows the use of heuristic smoothing techniques from language modeling (Chen and Goodman, 1999) to indirectly capture context low-dimensionally e.g., Vaswani et al. (2011) use a Markov model in the context of tree-to-string translation, where the parameters are smoothed with absolute discounting (Ney et al., 1994) (while in our instance we capture this smoothing effect through low rank or latent states), and while more principled approaches based on Pitman-Yor priors achieve good performance (Feng and Cohn, 2013), the n-gram methods are still limited by their unidirectional (i.e, left-to-right) notion of context and their reliance on smoothing as a proxy to reasoning about the effect of context dimensionality on translation.

Hsu et al. (2009) presented one of the initial efforts at spectral-based parameter estimation (using SVD) of observed moments for latent-variable models, in the case of Hidden Markov models. This idea was extended to L-PCFGs (Cohen and Collins, 2014), and our approach can be seen as a bilingual or synchronous generalization.

3.6 Summary

In this chapter, we presented an approach to refine synchronous grammars used in MT by inferring the latent categories for the single non-terminal in our grammar rules, and proposed two algorithms to estimate parameters for our latent-variable model. By fixing the synchronous

derivations of each parallel sentence in the training data, it is possible to avoid many of the computational issues associated with synchronous grammar induction. Improvements over a minimal grammar baseline and equivalent performance to a hierarchical phrase-based baseline are achieved by the spectral approach. For future work, we will seek to relax this consideration and jointly reason about non-terminal categories and derivation structures by devising estimation algorithms that can operatre over forests and not just fixed synchronous derivation trees.

Additionally, while the translation inventory does not grow in size since we handle context via auxiliary models, these models can still contain a very large number of parameters (§3.2). It would be useful to investigate another formulation of the spectral approach that is based on another assumption known as the "pivot assumption" (Cohen and Collins, 2014), where the actual parameters of the latent variable model (and not similarity transforms, as explained in §3.2) are recovered; in this formulation, it would be easier to impose sparsity-inducing regularization since we are dealing with the actual parameters. Recently, a clustering-based approach (Narayan and Cohen, 2015) was used to also induce sparser parameters, and this variant is also suitable for investigation in the MT setting.

The contributions are: a generalization of latent PCFGs (Matsuzaki et al., 2005) to latent SCFGs; an efficient tensor-based version of the inside-outside algorithm; an empirical demonstration that adding marginal rule probabilities from this model as features in the traditional linear translation model (Och and Ney, 2004) improves translation quality; and two algorithms that learn these latent categories (equivalently, the latent space) from the data without any externally imposed syntactic labels.

Chapter 4

Low-Dimensional Embeddings of Context

"There is nothing insignificant in the world. It all depends on the point of view."

— Johann Wolfgang von Goethe

In most scenarios during test time, the MT decoder has access to an entire document (or at least the entire sentence) prior to translation; it is thus absurd that traditional translation models use very little, if any, of the large amounts of observable context. Recent work has shown that extreme source-side context can be leveraged to great effect to improve translation (Devlin et al., 2014), as long as the representation of this context is manageable (which the authors achieve through neural network-based representations). Hence, in contrast to §3, where we compute low-dimensional representations of translation rules expressed in terms of their (high-dimensional) featurized context, this chapter concentrates on modeling the source context *directly* in a low-dimensional space.[1] While neural networks are amazingly expressive, they are notoriously difficult to train (Pascanu et al., 2013, *inter alia*), and instead we propose a simple method to recover a linear low-dimensional subspace which leverages the multi-view assumption (§2.2), where we use multiple "views" of the data to learn an appropriate low-dimensional basis in order to manage extremely amounts of context. As in §3, we use minimal grammars instead of composed ones and shift the context dependence to the lower-dimensional space, which makes estimation and inference more tractable.

[1] Including target context breaks one of the key independence assumptions made by phrase-based translation models, that translations of source phrases are conditionally independent of each other, given the source sentence. Target context is unobserved during evaluation, so conditioning upon this information directly in the translation model is computationally more difficult. Outside of the translation model, the language model also takes care of target-side context dependencies.

The basic setting in which we apply our low-dimensional context representations is reminiscent of "word-sense disambiguation" for MT (Carpuat, 2008), in that we condition on source contextual information to select the appropriate translation rule (i.e., source-target phrase pair) to be used in that context. Since we concentrate on selecting appropriate translations for source *phrases*, we use the term "phrase-sense disambiguation" (PSD) instead. Unlike most previous work, we hypothesize that by learning an appropriate low-dimensional basis first, we avoid significant feature engineering, and can take advantage of the reduced sample complexities when learning supervised models in the low-dimensional space. In order to learn such a basis for representing context, we primarily make use of CCA, but also compare against a bilingual generalization of the skip-gram model (Mikolov et al., 2013) that learns representations of phrase pairs instead of words (§4.1.1). Using the recovered context representations, we then investigate a number of models that use the target-side translations of source phrases as supervision (§4.1.2), with the aim being to take the context of a source phrase at test time, project it into the low-dimensional context space, and then reason about potential translation options directly in this space.

The evaluation of our proposed models is divided into several sections. First, we look at the impact of various hyperparameter choices related to the low-dimensional context, keeping a disambiguation model fixed (§4.2.2). Then, a variety of disambiguation models and hyperparameter settings are investigated using mean reciprocal rank (MRR, §4.2.3), including comparisons with disambiguation models that operate in the original high-dimensional space (§4.2.4). Last, an extrinsic evaluation (§4.2.5) measures the effect of these models under a number of different conditions and scenarios in terms of BLEU, by adding the model scores to an end-to-end MT setup. We found that while a low-dimensional context embedding does, in fact, exist, it is maximally captured by the language model (LM) and the surface-level relative frequency estimates (RFEs). For this reason, it is difficult to translate gains in MRR to gains in BLEU, but the low-dimensional models consistently do better than the high-dimensional ones in the BLEU experiments.

4.1 Phrase-Sense Disambiguation for MT

Focusing on the problem of sense disambiguation allows us to frame core translation operations in terms of a standard classification problem. To select amongst an inventory of phrasal rules, most current translation systems rely on surface-level relative frequency estimates (Koehn et al., 2003), as well as conditional lexical estimates that are computed on the words that make up a translation rule (and estimated via EM). These translation rule features are context-independent and therefore ignore the surrounding linguistic source context that is crucial for translation, although the LM mitigates this defect by handling long-range contextual dependencies on the target side. Thus, by developing models that can condition on context information in order to disambiguate various translation options, we can directly

investigate the impact that context, particularly of the low-dimensional kind, can have on translation models.

Our setup is similar to that of Ch. 3 in that we fix a fix a minimal derivation for a sentence pair *a priori* using the grammar extractor of Zhang et al. (2008), except we extract translation rules without any non-terminals. Thus, when referring to the translation rules in this setup, we use the terms "translation rule" and "phrase pair" interchangeably. To achieve this objective, we expand rules containing non-terminals by replacing the NTs on the RHS with child rules, up to a maximum length of five words on the source-side. Sentence pairs with difficult and/or noisy alignments may yield long phrase pairs by this procedure, and for such cases we do not include the translation rule in our phrasal inventory.[2] In order to avoid re-implementation of a decoder for test time, we add to this phrasal inventory two glue rules that combine NTs in a monotonic or inverted fashion, making our grammar essentially an inversion transduction grammar (ITG; Wu, 1997). These additions allow us to use the hypergraph algorithm from §3.2.1 and compute scores directly over the parse forest. By ensuring a fixed segmentation for each sentence pair (derived from the minimal derivation), we can sidestep estimation issues that would occur from considering an exponential number of possible segmentations (in the sentence length).

4.1.1 Low-Dimensional Context

In order to compute a low-dimensional representation of the context in which phrase pairs occur, we primarily make use of CCA. For occurrences of translation units in a parallel corpus, the source-side context of the rule can be split into two views: a natural one is context that occurs before the rule and context that occurs after. While each of the views is assumed to contain sufficient predictive power, by applying CCA to recover a shared latent space with a basis such that the projected points from the two views are maximally correlated, we compute the directions in which both views strongly agree and also reduce noise, since the assumption is that the noise in each view is uncorrelated. The "context CCA" recovers a pair of projection matrices (i.e., matrices that project sparse context vectors to their dense low-dimensional representations), one for the left context and one for the right context. These matrices project the left and right context into a *shared* latent space, and we concatenate the resulting low-dimensional vectors to yield a low-dimensional representation for the context in which a phrase occurs (Foster et al., 2008).

Let $X \in \mathbb{R}^{N \times d}$ and $Y \in \mathbb{R}^{N \times d'}$ contain our two views, where N is the number of translation rule tokens in our training corpus and d, d' are the dimensionalities of the left and right context spaces respectively. Recall that regularized CCA can be computed by applying a rank-k SVD

[2] While this admittedly results in incomplete sentence pairs, the rate at which such phrase pairs are extracted is relatively small.

on the following matrix (§2.2.1):

$$(X^T X + \gamma_1 I)^{-1/2} X^T Y (Y^T Y + \gamma_1 I)^{-1/2}$$

where $\gamma_1 > 0$ is a regularization parameter that ensures non-singular covariance matrices. The square root inverse (SRI) operation has the effect of whitening the X and Y spaces, since it is a normalization by the variances and covariances of individual features. However, computing the SRI of the empirical covariance matrices can be an expensive operation, especially if the original feature dimensionalities d, d' are very large. We thus apply a number of approximations in lieu of computing these quantities:

- identity (IDENT): $(X^T X + \gamma_1 I)^{-1/2} = I$ and $(Y^T Y + \gamma_1 I)^{-1/2} = I$. In this approximation, we ignore any the feature variances and covariances, and simply compute the SVD of $X^T Y$. The regularization parameter γ_1 is also ignored. Note that this approximation to the CCA is used in Ch. 4 (where the two views are inside and outside trees).

- diagonal approximation (DIAG): instead of considering the full $X^T X$ (or $Y^T Y$) empirical covariance matrix, we only consider the regularized diagonal terms: $(X^T X + \gamma_1 I)^{-1/2} = (\text{diag}(X^T X) + \gamma_1 I)^{-1/2}$, and similarly for Y. In other words, feature covariance effects are ignored and only the feature variances are taken into account. Computing the SRI of a diagonal matrix is linear in the length of the diagonal.

- randomized (RAND): computing the full SRI of X and Y is simply too costly for our problems, where feature dimensionalities are on the order of 10^5. Instead, we use the randomized algorithm presented by Mineiro and Karampatziakis (2014), which uses a randomized range finder to iteratively compute the (orthogonal) column space or range of the data matrices X and Y, after which a Cholesky decomposition and SVD is used to find the canonical correlations. The behavior of randomized range finders has been well-analyzed in the literature (Halko et al., 2011) and a reasonable approximation can be achieved with oversampling and multiple passes through the data.

4.1.2 Disambiguation Models

Naturally, the CCA computation to yield the low-dimensional context representations is a rank-reduced one; assuming a rank-k context CCA, the resulting dimensionality of the concatenated low-dimensional context representations is $2k$. We can use these low-dimensional representations in a number of different ways to learn supervised models with regularization parameter $\gamma_2 > 0$:

- 2-step CCA (CCA): similar to Dhillon et al. (2012), we compute a regularized rank-m CCA between the concatenated $2k$-dimensional context representations (an $N \times 2k$ matrix when considering the entire training data) and a sparse, high-dimensional matrix

corresponding to phrase pair tokens, which we call Z. Z is an $N \times P$ matrix, where P is the number of phrase pair types in the training corpus. Each row in Z contains only one non-zero value, indicating the identity of the phrase pair. The result is a latent space where the two sets of random variables (corresponding to context and phrase pairs) are maximally correlated, along with projection matrices from the $2k$ and P-dimensional spaces to the latent m-dimensional space. Translation options are then ranked through a nearest neighbor model in the m-dimensional space, where we use the cosine similarity between the m-dimensional phrase pair and context representations for similarity. The idea is that phrase pairs will reside relatively close to the contexts they occurred in during training in the m-dimensional space, and scoring phrase pairs now becomes a question of projecting new context to the same space and and querying nearby phrase pairs.[3]

- 2-step CCA with Matching Model (CCA+MM): phrase pair and context representations can also act as inputs to a simple multilayer perceptron (MLP), as is done in Tu et al. (2015). The MLP is trained as a regularized binary classifier (with hyperparameter $\gamma_3 > 0$) with a tanh non-linear activation function for the input-hidden layer and a logistic activation function for the hidden-output layer, and takes a concatenated context-translation rule representation in the low-dimensional space (of length $2m$) to predict whether the translation rule is the correct one to use in the context or not. Since we have a fixed translation inventory that is extracted from the training corpus before representation estimation, it is easy to convert our training corpus into one that can be used to train the MLP. For each of the N translation rule occurrences, we provide as negative examples all other representations of translation rules with the same source phrase, concatenated with the context at that occurrence. For most phrase pairs there is more than one negative example (i.e., there are more than two translation rules for that source phrase), and so we compensate for the class imbalance problem by oversampling the positive examples, such that the overall number of positive and negative examples provided for MLP learning is the same. The number of hidden units in the MLP is an additional parameter.

- Ordinary Least Squares (LS): by treating the phrase pair token matrix Z as the dependent variable and the concatenated low-dimensional representations $[X;Y]$ in a linear regression setup, we can compute the regularized normal equations $([X;Y]^T[X;Y] + \gamma_2 I)^{-1}[X;Y]^T Z$, which reveal parameters that minimize squared error and is akin to finding the best projections of Z onto the space spanned by the low-dimensional context representations. The computation of the normal equations involves the inversion of a $2k \times 2k$ matrix, which is generally small for our experimental settings ($k = 50$ to 200).

[3] Why not just compute a single CCA between the concatenated left-right context in the original $d + d'$-dimensional space and the phrase pair matrix in the P-dimensional space? The 2-step approach has the advantage of reduced sample complexity (Dhillon et al., 2012) and the additional flexibility of defining a context latent space of different dimension than the context-translation unit latent space.

Note that this approach is similar to the 2-step CCA method, except we do not take into account the variance of the high-dimensional translation rule space (§14).

- Online Least Squares (LS-ONLINE): this model is very similar to LS in that we minimize a regularized squared loss objective, with the independent variables once again being the concatenated low-dimensional representations $[X; Y]$, but the objective function is optimized using a variant of adaptive gradient descent (Duchi et al., 2011). The model is also similar to a maximum entropy classifier (Berger et al., 1996), which is a popular approach in natural language processing, except maximum entropy classifiers minimize a multiclass generalization of the log loss, which involves computing an expensive normalization factor that is avoided in our case.[4] Lastly, we can also add a hidden layer with a tanh non-linearity to this setup, making this a simple single-layer MLP (MLP). The number of hidden units in the MLP is an additional parameter.

The above models leverage the low-dimensional context from CCA (4.1.1), but we can also use a suitable generalization of the skip-gram model for learning word vector representations. The skip-gram model (SG) maximizes the following objective:

$$\sum_{i=1}^{N} p(\boldsymbol{c}|p_i) = \sum_{i=1}^{N} \prod_{j=1}^{|\boldsymbol{c}|} \frac{\exp(\mathbf{v}_{c_j} \cdot \mathbf{v}_{p_i})}{\sum_{w=1}^{W} \exp(\mathbf{v}_w \cdot \mathbf{v}_{p_i})} \tag{4.1}$$

where $\boldsymbol{c}$ is the context for a phrase p at position i and is a function of the phrase (usually consisting of a window of source words before and after the phrase), and $\mathbf{v}_{p_i}, \mathbf{v}_{c_j}$ are the low-dimensional vector representations for the phrase pair p_i and the particular context word c_j. As explained in Goldberg and Levy (2014), the context representations are distinct from the word, or in this case phrase pair, representations. Thus, at the end of training we obtain two sets of representations: one for the context *words*, and another for the translation rules i.e., phrase pairs. To avoid the expensive normalization factor in Eq. 4.1, we use the negative sampling technique which samples context words from an altered unigram distribution, using 5 negative samples[5] and the default hyperparameters from the `word2vec` software in other instances. Instead of maximizing Eq. 4.1 directly, we maximize the sum of the log probabilities $\sum_{i=1}^{N} \log p(\boldsymbol{c}|p_i)$ using backpropagation via a distributed, asynchronous version of regularized gradient descent (Mikolov et al., 2013) with hyperparameter $\gamma_1 > 0$. The resulting context and phrase pair vectors can be used in a nearest neighbor model as is done with the 2-step CCA, where the dot product is computed via cosine similarity.

[4] See `http://hunch.net/?p=547` for a discussion on why squared loss should be preferred to log loss.
[5] The results were generally invariant to the number of negative samples.

4.2 Evaluation

We evaluated the models presented in this chapter with the aim of eliciting several insights into the nature of low-dimensional context and how it applies to MT. First, establish the best settings to yield low-dimensional context spaces (§4.2.2); this objective entails experimenting with the various approximations to the square-root inverse (§4.1.1) and also the context rank k i.e., the size of the low-dimensional context space. Second, evaluate ways to leverage information in this space to effectively re-rank translation options (§4.2.3) using the models presented in §4.1.2; we vary hyperparameters associated with the supervised models. Third, establish the efficacy of low-dimensional context compared to its high-dimensional counterpart (§4.2.4); specifically, we compare our models against classifiers that minimize squared loss, trained on the sparse, high-dimensional context. And fourth, determine whether the additional ranking information helps in an end-to-end MT evaluation (§4.2.5), or if such information has already been provided by existing features in the MT setup, like the LM and the RFE probabilities. We also compared our linear CCA-based models to a phrase pair-based skip-gram variant that uses nonlinear activation functions, to see if nonlinearities could improve the learned representations and therefore the resulting supervised disambiguation models.

Except for the MT experiments, which are measured with BLEU (Papineni et al., 2002), we used the mean reciprocal rank (MRR) as our primary evaluation metric. MRR is an intuitive metric to evaluate ranking models, and we use it as an intrinsic evaluation on a heldout set to understand the performance of our models without the additional complications of MT systems, including optimizer instability (Clark et al., 2011) and the effect of the LM. The MRR is computed in the following manner:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{rank}_i}$$

Intuitively, if a ranking model achieves a higher MRR than another on a fixed evaluation set, then it means (on average) the model ranks the correct answer higher than the other model.

The software used during the evaluation was written primarily in Python, and has been released as a package for broader consumption: `CCA-MT`[6]. Certain aspects of the computations were done with specialized software. All SVD computations for the purposes of CCA as well as Normal Equation evaluations were done in MATLAB. Vowpal Wabbit[7] was used for our MLP matching models in CCA+MM, the LS-ONLINE models, and the sparse, high-dimensional LS-ONLINE baselines. For the skip-gram model, we modified the version presented in `word2vec`[8] (Mikolov

[6] https://github.com/asaluja/cca-mt
[7] https://github.com/JohnLangford/vowpal_wabbit
[8] https://code.google.com/p/word2vec/

et al., 2013) to learn representations of phrase pairs by fitting parameters that maximize the likelihood of source-side context. The modified version is also included in `CCA-MT`.

4.2.1 Corpora

Chinese-English (ZH-EN) was the chosen language pair for evaluation; in particular, we looked at two different corpora. The first is the BTEC corpus, which is the same corpus used in Ch. 3 (§3.4), and we refer the reader to Table 3.1 for additional information regarding the corpus and the evaluation sets it comes with. The second is the FBIS corpus[9], with the NIST MT03 and MT06 evaluation sets used for development and testing purposes respectively. Corpus statistics for this corpus are presented in Table 4.1. Unlike BTEC, which consists exclusively of conversational speech, the FBIS corpus is primarily news commentary.

FBIS ZH-EN	Words
TRAIN (SRC)	7.667M
TRAIN (TGT)	9.096M
DEV (SRC)	24.1K
DEV (TGT)	29.2K
TEST (SRC)	38.8K
TEST (TGT)	46.4K

Table 4.1: FBIS corpus statistics (in words). For the target DEV and TEST statistics, we take the first reference (four references total).

The BTEC corpus is used in all evaluations, and is the only corpus used in our context analysis (§4.2.2). The FBIS corpus is utilized to evaluate promising disambiguation models and compare against high-dimensional context. Both corpora are evaluated on during the MT experiments. For the MRR evaluations, we selected a random (but fixed across model variations so that the comparisons are meaningful) 5% held-out subset of the training data. For BTEC, after filtering for phrase pairs that we did not estimate representations for (see §4.2.3 for the criteria used to filter phrase pairs) and source phrases that have only one translation, the heldout set consists of more than 10K examples. For FBIS, applying the same criteria results in more than 166K examples in the heldout set.

4.2.2 Context Experiments

For the context experiments, we looked at the impact of the SRI approximations as well as the size of the low-dimensional space. In order to form a meaningful comparison, we fixed a number of hyperparameters related to the context as well as the disambiguation model.

[9] LDC2003E14

A window size of 5 context words (features) on each side were used (fewer words if at the beginning/end of a sentence or if the sentence is short), with the positions of the context words taken into account (i.e., *not* a bag-of-words model). A minimum context word count of three words is imposed; if below this threshold, the words are used for position-dependent OOV representation estimation. For BTEC, these thresholds result in the removal of almost 27K features from the left context and more than 29K features from the right context, leaving high-dimensional feature spaces of size $d = 14,612$ for the left context and $15,769$ for the right context.

For the disambiguation model, we select LS since, unlike the 2-step CCA variants, the disambiguation is not dependent on the SRI approximation, and is easy and fast to compute. We vary the rank for the three approximations, and fix $\gamma_1 = 2$ and $\gamma_2 = 1$. Figure 4.1 presents the results as the context CCA rank k is varied. Overall, we find that while there is a consistent (but small) increase in MRR as we vary the rank, there is also little to separate the approximation techniques (less than 0.005 MRR across all ranks). Since the diagonal approximation works well in the low-dimensional regimes ($k = 25, 50, 100$), we fix this approximation for all subsequent experiments.

4.2.3 Model Variants

Next, we looked at the various disambiguation models proposed in §4.1.2. For the BTEC corpus, we varied the context CCA rank $k = 50, 100$ but found that increasing k beyond 50 provided only a small improvement to the MRR scores. Thus, k is fixed to 50; other context-related settings (window size, minimum count thresholds) remain the same as in §4.2.2. Furthermore, we prune all singleton phrase pairs, and for a given source phrase, keep only the top 20 phrase pairs sorted by forward RFEs. The result is that we estimate parameters for 12,791 phrase pairs in the BTEC corpus.

Table 4.2 presents results with varying hyperparameters and/or ranks for BTEC. For the CCA and CCA+MM setups, the rank of the CCA between the concatenated low-dimensional context and the phrase pair matrix is varied, although note that when rank = 100, we simply bi-orthogonalize the spaces and no rank reduction is done. The CCA+MM setup has an additional rank parameter and hyperparameter γ_3, which dictate the MLP matching model that is built on top of the 2-step CCA. For the LS, LS-ONLINE, and MLP setups, the rank is 101 because in addition to the concatenated $2k$-dimensional context, we add a bias term. The LS-ONLINE and MLP setups have much smaller regularization strengths since the regularizer is added on a per-example (online) basis, and thus has a different interpretation from the regularization hyperparameters in the other setups. Lastly, the hyperparameter for the SG setup has a similar interpretation to the LS-ONLINE and MLP setups, and is set to the `word2vec` default.

The CCA model is clearly inferior, but with the help of the matching model we can achieve performance roughly equivalent to the LS model. Using a different optimization algorithm

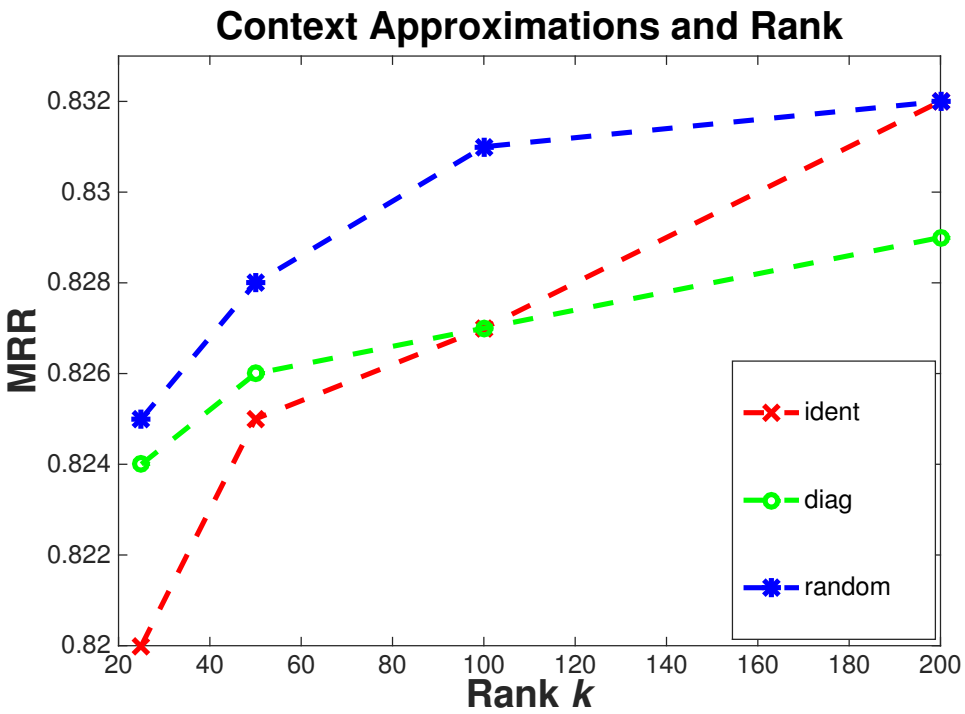


Figure 4.1: MRR as a function of rank for the three different square-root inverse approximations, with an LS model as the disambiguation model. The MRR is relatively invariant to the approximation used.

through the LS-ONLINE setup, we can get a slight improvement in MRR. Interestingly, adding a hidden layer to this setup, thus making it akin to an MLP model, does not seem to help; even adding 10 hidden layers decreases the MRR. Lastly, the SG model performs much better than the CCA setup, despite both models using a nearest neighbor predictor, indicating that the additional non-linearities added by the SG model are useful in ranking translation options.

Table 4.3 presents a similar set of results for the FBIS corpus. For the FBIS experiments, the hyperparameters used for the context are different since the corpus is much larger in terms of both tokens and types, so care must be taken to make computations tractable. We consider a window size of 4 words on each side, and increase the minimum context word count to 6 (from 3 in the BTEC case). The resulting left and right context dimensionalities are $d = 99,091$ and $d' = 96,745$ respectively. Unlike BTEC, where we did not find much improvement beyond $k = 50$, here we present results with varying Context CCA ranks ($k = 50, 100$), since the magnitude of the effect on MRR is larger. For the phrase pairs, we prune all phrase pairs that occur less than three times in the corpus, and as with BTEC, keep only the top 20 phrase

	Parameters	**Rank m**	**MRR**
CCA	$\gamma_2 = 1$	50	0.616
	$\gamma_2 = 1$	100	0.654
CCA+MM	$\gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	100,25	0.820
	$\gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	100,50	0.824
LS	$\gamma_2 = 1$	101	0.826
LS-ONLINE	$\gamma_2 = 1 \times 10^{-5}$	101	0.815
	$\gamma_2 = 1 \times 10^{-8}$	101	**0.834**
MLP	$\gamma_2 = 1 \times 10^{-8}$	101,10	0.826
	$\gamma_2 = 1 \times 10^{-8}$	101,25	0.812
SG	$\gamma_1 = 2.5 \times 10^{-2}$	50	0.799

Table 4.2: MRR results on the 5% heldout set for the BTEC corpus (roughly 10K examples). The context CCA rank $k = 50$ in all cases. A naive 2-step CCA approach (CCA) does not perform well, but by utilizing the supervision in a more direct way, we get significant improvements. The best performing setup (LS-ONLINE) is in bold.

	k	**Parameters**	m	**MRR**
CCA	50	$\gamma_2 = 1$	50	0.389
	50	$\gamma_2 = 1$	100	0.426
	100	$\gamma_2 = 1$	100	0.420
CCA+MM	50	$\gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	100,25	0.569
	50	$\gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	100,50	0.68
	100	$\gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	100,50	0.668
LS	50	$\gamma_2 = 1$	101	0.728
	100	$\gamma_2 = 1$	101	**0.730**
LS-ONLINE	50	$\gamma_2 = 1 \times 10^{-5}$	101	0.657
	50	$\gamma_2 = 1 \times 10^{-8}$	101	0.725
	100	$\gamma_2 = 1 \times 10^{-8}$	101	0.727
MLP	50	$\gamma_2 = 1 \times 10^{-8}$	101,10	0.702
SG	50	$\gamma_1 = 0.025$	50	0.658
	100	$\gamma_1 = 0.025$	100	0.694
	200	$\gamma_1 = 0.025$	200	0.710

Table 4.3: MRR results on the 5% heldout set for the FBIS corpus (roughly 166K examples). Recall that k refers to the context CCA rank, and m refers to the rank of the disambiguation model. The LS and LS-ONLINE models once again perform quite strongly.

pairs sorted by forward RFEs. Correspondingly, we estimate phrase pair representations for 108,958 phrase pairs.

	Regularization	1×10^{-2}	1×10^{-5}	1×10^{-8}
BTEC	ℓ_1	0.476	0.625	0.812
	ℓ_2	0.696	**0.812**	0.811
FBIS	ℓ_1	0.271	0.271	**0.726**
	ℓ_2	0.475	0.654	0.718

Table 4.4: MRR results on the 5% heldout sets for the high-dimensional MLR models. The models in bold are also evaluated for BLEU impact in §4.2.5.

4.2.4 High-Dimensional Comparison

One of the objectives of the evaluation is to explicitly compare against an LS-ONLINE model which operates in the original, high-dimensional space; specifically, each example is the concatenation of the d-dimensional left context space and the d'-dimensional right context space, resulting in a sparse, $d + d'$-dimensional vector. As with our LS-ONLINE models that predict phrase pairs based on the low-dimensional context representations, we use Vowpal Wabbit to minimize a regularized squared loss objective using adaptive stochastic gradient descent. For the BTEC corpus, $d + d' = 30,381$, and for the FBIS corpus, $d + d' = 195,836$.

Table 4.4 presents results for both corpora. We vary the ℓ_2 regularization strength, and also experiment with a sparsity-inducing ℓ_1 regularization term with varying strengths due to the high-dimensional nature of the model. For both corpora, the high-dimensional baseline performs surprisingly well; on BTEC, this model is superior to the CCA approach and even slightly better than the SG method. Regardless of regularization strength, the LS-ONLINE model, which is the most similar setup to the high-dimensional case, always does better. On FBIS, the high-dimensional baselines also do quite well, and in fact do better than most of the disambiguation model setups. We thus also evaluate their end-to-end MT impact in §4.2.5 by evaluating the bolded models from Table 4.4.

4.2.5 MT Experiments

We used the `cdec` decoder (Dyer et al., 2010) to extract word alignments from a parallel corpus, tune the MT decoder feature weights, and for decoding. For comparison, we built two baselines: the primary one is the minimal grammar baseline (MINIMAL), which is a setup with the same grammar as our models, but which doesn't score rules, as our models do in a context-dependent fashion. The HIERO system uses a grammar extracted by applying the commonly used heuristics (Chiang, 2007). Each rule is decorated with two lexical and phrasal features corresponding to the forward $(e|f)$ and backward $(f|e)$ conditional log frequencies, along with the log joint frequency (e, f), the log frequency of the source phrase (f), and whether the phrase pair or the source phrase is a singleton. Weights for the language model (and language

model OOV), glue rule, and word penalty are also tuned. The MINIMAL baseline maintains the same set of weights. For additional comparison, we also compare against the sparse, high-dimensional LS-ONLINE models and evaluate how they perform in an MT setting.

For the BTEC experiments, the language model is a 3-gram model extracted from the target-side of the training data, and for the FBIS corpus we use a very large, 5-gram model trained on the English Gigaword corpus. Both language models are built using KenLM (Heafield, 2011). To tune the MT feature weights, we make use of MIRA (Chiang, 2012), which is a large-margin learning technique. All MT tuning runs were repeated three times to control for optimizer instability (Clark et al., 2011), and the average BLEU scores on the development and test sets are stated, with standard deviation in parentheses. Statistically significant improvements over the minimal grammar baselines are bolded ($p < 0.05$).

Table 4.5 presents the BLEU results for the BTEC corpus. In all instances, we score a phrase pair in its context using our models, but actually output the reciprocal rank $\frac{1}{\text{rank}_i}$ for the phrase pair, where we rank translation options for each source phrase. For example, if the phrase pair "'el perro ||| a dog"' is scored just below "'el perro ||| the dog"' and the source phrase "'el perro"' has only two translation options, the former will receive a score of 0.5 and the latter will receive a score of 1. Scoring phrase pairs in terms of reciprocal rank normalizes any scale or range differences that may arise from the different variations. In Table 4.5, we find that the LS and LS-ONLINE setups, which performed fairly well in terms of MRR, do not improve on the baseline noticeably in terms of BLEU. In fact, along with the SG representations, these approaches do not do as well as the LS-ONLINE high-dimensional model. Surprisingly, the CCA and CCA+MM models perform quite strongly in terms of BLEU.

With the CCA and CCA+MM setups, we added some variants when writing out the scores, including an indicator feature for the best-scoring translation for a given source phrase (best), and passing the raw score through a logistic function (logit). These additions improved scores marginally for the CCA model, but not for the CCA+MM one, so we only state results for the former in Table 4.5. Overall, our best setup achieves a 0.72-BLEU point improvement over the corresponding minimal baseline. Unlike the results in §3.4, none of the setups are able to get close to the HIERO BLEU scores. The weights after MIRA training are also instructive: for BTEC, while the context-based score gets a relatively high weights (0.20–0.25), the LM weight is even higher (more than 0.70). Interestingly, while the high-dimensional LS-ONLINE models were very competitive in terms of MRR, they do not improve upon the MINIMAL baseline in a significant manner. Table 4.6 presents results for the FBIS corpus. The high-dimensional baseline in this instance does worse than the MINIMAL baseline, and the CCA variants (CCA, CCA+MM) do not statistically improve upon this baseline. However, the skip-gram model does surprisingly well (+1.04 BLEU), given that other models do slightly better in terms of MRR, and both this model as well as the LS model statistically improve upon the baseline. These gains can either be attributed to jointly learning the low-dimensional context and phrase pair representations instead of the two-stage learning procedure adopted for the other models, or in

	Setup	BLEU Dev	BLEU Test
Baselines	HIERO	46.58 (0.26)	57.04 (0.47)
	MINIMAL	42.80 (0.29)	51.40 (0.06)
	High-Dim $\ell_2 = 1 \times 10^{-5}$	42.81 (0.07)	51.85 (0.31)
Variations	CCA $k = 50, m = 100, \gamma_1 = 2, \gamma_2 = 1$	43.15 (0.13)	52.01 (0.27)
	CCA+MM $k = 50, m = 100{+}25, \gamma_1 = 2, \gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	43.05 (0.03)	**51.94** (0.20)
	LS $k = 50, m = 101, \gamma_1 = 2, \gamma_2 = 1$	42.86 (0.43)	51.52 (0.22)
	LS-ONLINE $k = 50, m = 101, \gamma_1 = 2, \gamma_2 = 1 \times 10^{-8}$	42.62 (0.05)	51.59 (0.35)
	SG $k = 50, m = 50, \gamma_1 = 2.5 \times 10^{-2}$	43.14 (0.10)	51.67 (0.18)
Additions	CCA+best	42.99 (0.13)	**52.12** (0.14)
	CCA+logit	43.12 (0.26)	52.09 (0.41)
	CCA+best+logit	42.91 (0.13)	52.09 (0.21)

Table 4.5: Results in BLEU on the BTEC development and test sets. All MIRA tuning runs were repeated 3 times, with the mean score reported (standard deviation in parentheses). Results in bold are statistically significant improvements over the MINIMAL baseline ($p < 0.05$). While only two results are statistically significant at this level, several other setups are significant at the $p < 0.10$ level.

	Setup	BLEU Dev	BLEU Test
Baselines	HIERO	34.57 (0.39)	30.26 (0.43)
	MINIMAL	28.87 (0.07)	23.98 (0.31)
	High-Dim $\ell_1 = 1 \times 10^{-8}$	28.65 (0.48)	**23.73** (0.65)
Variations	CCA $k = 50, m = 100, \gamma_1 = 2, \gamma_2 = 1$	29.02 (0.11)	24.03 (0.24)
	CCA+MM $k = 50, m = 100{+}50, \gamma_1 = 2, \gamma_2 = 1, \gamma_3 = 1 \times 10^{-5}$	**29.17** (0.45)	24.04 (0.29)
	LS $k = 100, m = 201, \gamma_1 = 2, \gamma_2 = 1$	**29.50** (0.21)	**24.66** (0.41)
	LS-ONLINE $k = 100, m = 201, \gamma_1 = 2, 1 \times 10^{-8}$	28.95 (0.90)	24.16 (0.50)
	SG $k = 100, m = 100, \gamma_1 = 2.5 \times 10^{-2}$	**29.46** (0.37)	**25.02** (0.08)

Table 4.6: Results in BLEU on the NIST MT03 and MT06 sets, using the FBIS corpus as a training set. All MIRA tuning runs were repeated 3 times, with the mean score reported (standard deviation in parentheses). Results in bold are statistically significant improvements over the MINIMAL baseline ($p < 0.05$). Unlike Table 4.2, we do not provide results for the additions since it was found that these elaborations do not help for this corpus. Note that the result of the high-dimensional baseline is significantly *worse* than the minimal baseline.

terms of the non-linearity used for the hidden layer, since in general adding such non-linearities allows for better function approximation.

	MRR	
Feature	BTEC	FBIS
LM	0.900	0.748
$P(f\|e)$	0.435	0.414
$P(e\|f)$	0.816	0.723
LM + $P(f\|e)$	0.82	0.692
LM + $P(e\|f)$	0.911	0.845

Table 4.7: MRR results on the 5% heldout set using basic features available to an MT decoder. Surprisingly, these features perform very strongly and better than any of the proposed models (see Tables 4.2 and 4.3), which suggests that our low-dimensional disambiguation models are primarily capturing information that is similar to the existing features.

Given the relatively high MRR scores achieved by the proposed models, it is somewhat surprising that the BLEU gains are quite small. Furthermore as noted above, certain setups that perform well in terms of MRR (Tables 4.2, 4.3, and 4.4) do not necessarily do well in terms of BLEU. In order to understand the nature of these BLEU gains, we decided to evaluate the MRR on the same heldout set, but this time using information or scores that are available to an MT system during decoding (i.e., the language model scores) or based on surface-level RFEs. Table 4.7 presents the MRR scores of these features on both the BTEC and FBIS corpora. When combining score information e.g., "LM + $P(e|f)$", we weight each term equally. The results show that the LM feature alone is better than any disambiguation model, low-dimensional or otherwise, and suggests that the disambiguation models capture information already contained in existing MT features, like the RFEs or the LM; even though many disambiguation models score well in terms of MRR, the BLEU gains are minimal. This effect is stronger on BTEC than FBIS, and the weight of the LM feature after MIRA training is indicative: greater than 0.70 on BTEC, whereas on FBIS it is in the 0.15-0.20 range. Furthermore, this hypothesis may also explain why models that may score relatively lower in terms of MRR perform relatively well in terms of BLEU (like CCA on the BTEC corpus, and SG on the FBIS corpus): they are providing information that is relatively more orthogonal to the LM and RFE scores than other models.

4.3 Related Work

Using context to drive disambiguation decisions in MT has been looked at from several angles, but Carpuat and Wu (2005) were the first to use word-sense disambiguation techniques, by predicting the HowNet (Chinese WordNet) sense of a word using WSD, and then using the English gloss of the HowNet sense as the model's predicted translation. However, they do not get improvements. Chan et al. (2007) managed to get better translations by using a multiclass SVM-based model and integrating this model by adding additional features in a traditional

MT setup; unlike Carpuat and Wu (2005), the same data was used to train the disambiguation model as was used to train the translation model (phrase pair extraction, feature estimation). They evaluate on FBIS, and get a small (+0.57 BLEU) but statistically significant improvement. A follow-up work by Carpuat and Wu (2007) however, did perform better by utilizing several improvements, for example multi-word phrasal lexical disambiguation and a training regime integrated into the overall framework. They show a small (+0.5 BLEU), but statistically significant improvement on BTEC. Some of these ideas have subsequently been used for specific applications in MT, e.g., suggesting translations for OOV words (Daumé III and Jagarlamudi, 2011) or identifying when words obtain new senses due to a change in domain (Carpuat et al., 2013).

In these works as in our models, features are extracted from the source sentence only, and this idea has also been used in approaches (Stroppa et al., 2007, Gimpel and Smith, 2008, He et al., 2008) that enrich the standard linear translation setting by adding source context features and tuning their weights, along with the weights of standard MT features like the relative phrasal frequency estimates and the language model, using standard algorithms like MERT (Och, 2003). While some of these approaches have been effective (potentially due to the intrinsic dimensionality of context), this line of work has in general achieved mixed results because the low-dimensional view is never explicitly leveraged. Most of these models are heavily dependent on high-dimensional lexical features and manually-defined coarser features like part-of-speech (POS) tags, and minimal effort is made to reason about context in a lower-dimensional space or about translation rules (in terms of context) in such spaces. Furthermore, PSD is carried out on top of a massive, heuristically-extracted phrase table, where much of the local context is incorporated within large translation rules, instead of a minimal grammar where the context can be more directly handled by auxiliary models.

There has also recently been a spurt of research that uses neural network models to either learn phrase pair representations directly in an end-to-end setup (Kalchbrenner and Blunsom, 2013, Cho et al., 2014), or learn compact representations of the context (Devlin et al., 2014, Tu et al., 2015), which is then conditioned on downstream for translation purposes. Cho et al. (2014) use an autoencoder framework with a recurrent neural network (RNN) on either end which builds a dense, continuous representation as we scan through the words that constitute a phrase. They score phrase pairs from an existing model using this approach, although in theory it is possible to completely replace the existing translation model with their autoencoder. Kalchbrenner and Blunsom (2013) have a similar setup, except they use a convolutional neural network to compute a representation of the soure sentence. The approach of Devlin et al. (2014) essentially augments a neural network language model by taking into account large windows of source-side context, using the word alignments to incorporate the dependencies. They achieve significant gains through this approach, a +3 BLEU gain over a state-of-the-art system. In Tu et al. (2015), the authors use a convolutional neural network to come up with dense, low-dimensional representations of both the entire sentence in which a source phrase occurs as well as the phrase pair itself, and then use these representations as an input into an MLP matching

model, which is where we derive our inspiration for using the matching model. In addition, they propose a curriculum learning strategy where training examples are sorted from easiest to hardest (using heuristics to judge difficulty), but with all of these elaborations their gains are on the order of 1 BLEU point. All of these approaches once again are carried out on top of a large, composed phrase table or grammar.

More recently, Luong et al. (2015) make use of attention-based models (originally proposed in the image recognition domain) in an end-to-end setup to focus on parts of the source sentence that are relevant for translation of target words. The attention model can be seen as a type of alignment model[10], since it specifically weights the contribution of individual source token representations in the generation or translation of a target word. A hybrid method which makes use of the attention model to select relevant source tokens and CCA to compute low-dimensional representations of source context and phrase pairs is feasible, and we leave its exploration as future work.

The multi-view assumption was first introduced in Kakade and Foster (2007), and further elaborated in Foster et al. (2008); in that work, the authors show that a straightforward least-squares regression formulation can have improved sample complexity properties, by recovering a latent, low-dimensional feature space using CCA. Dhillon et al. (2011, 2012) apply the assumption to come up with word representations that they call "eigenwords"; it is their 2-step CCA version that generalize and make use of here in the MT setting.

4.4 Summary

In this chapter, we proposed using CCA to yield a low-dimensional basis for context, after which we suggested the use of several models that reason about translation options in this space. While we established the empirical efficacy of these models in terms of MRR, and found that low-dimensional disambiguation models perform far better than their high-dimensional counterparts in terms of BLEU, the overall improvement over reasonable baselines was small, indicating that much of the information our models are capturing are already present through other features in an MT system. The contributions of this section are: several proposed approaches to translation sense disambiguation that work directly with low-dimensional context representations; and empirical evidence that shows incorporating local context in conjunction with minimal grammars results in MRR and BLEU improvements.

[10] In a complete neural MT setup, word alignments from lexical models (§2.1.1) are not used.

Chapter 5

Low-Dimensional Context & Semi-Supervised Learning

"Manifolds are a bit like pornography: hard to define, but you know one when you see one."

— Shmuel Weinberger

Importantly, the source language context information we seek is not restricted to parallel sentence corpora, and representations can be learned from the much more copious amounts of monolingual data available. In this section, we investigate the central hypothesis of this thesis in the semi-supervised learning (SSL) setting. Although statistical approaches to MT use sentence-aligned, parallel corpora to learn translation rules along with their probabilities, a semi-supervised approach allows us to test our hypothesis in the limit.[1] By extracting context over large monolingual corpora, we have access to rich, variable contexts that exist in an extremely high-dimensional space of size proportional to vocabulary size. Since we hypothesize that the salient context information on which our models should condition resides in a low-dimensional space, utilizing this perspective should do better than the high-dimensional view, keeping in mind the obvious computational benefits it also introduces.[2]

The challenge of learning translations from monolingual data is of long standing interest, and has been approached in several ways (Rapp, 1995, Callison-Burch et al., 2006, Haghighi et al., 2008, Ravi and Knight, 2011). Our work introduces a new take on the problem using graph-based semi-supervised learning to acquire translation rules and probabilities by leveraging both monolingual and parallel data resources. On the source side, labeled phrases (those

[1] Even in resource-rich languages, learning reliable translations of multiword phrases is a challenge, and an adequate phrasal inventory is crucial for effective translation.

[2] This chapter is based on material published originally in Saluja et al. (2014b).

with known translations) are extracted from bilingual corpora, and unlabeled phrases are extracted from monolingual corpora; together they are embedded as nodes in a graph, with the monolingual data determining edge strengths between nodes (§5.1.2). Unlike previous work (Irvine and Callison-Burch, 2013a, Razmara et al., 2013), we use higher order n-grams instead of restricting to unigrams, since our approach goes beyond OOV mitigation and can enrich the entire translation model by using evidence from monolingual text. This enhancement alone results in an improvement of almost 1.4 BLEU points. On the target side, phrases initially consisting of translations from the parallel data are selectively expanded with generated candidates (§5.1.1), and are embedded in a target graph.

We then limit the set of translation options for each unlabeled source phrase (§5.1.3), and using a structured graph propagation algorithm, where translation information is propagated from labeled to unlabeled phrases proportional to *both* source and target phrase similarities, we estimate probability distributions over translations for the unlabeled source phrases (§5.1.4). The additional phrases are incorporated in the SMT system through a secondary phrase table (§5.1.5). We evaluated the proposed approach on both Arabic-English and Urdu-English under a range of scenarios (§5.2), varying the amount and type of monolingual corpora used, and obtained improvements between 1 and 4 BLEU points, even when using very large language models.

It should be noted that there are two forms of dimensionality reduction that we will discuss in this chapter. The first is the graph embedding itself: graphs are a useful way to encode neighborhood information for high-dimensional data, and our graph propagation algorithms take into account only this information (through the edge strength between nodes) during inference. Using contextual similarity to construct graphs (based on high-dimensional context) is an empirically effective approach and widely adopted in NLP (Subramanya et al., 2010, Das and Petrov, 2011, *inter alia*), but also from our perspective using low-dimensional representations of this context prior to graph construction reduces the intrinsic dimensionality of the problem. Essentially, the graphs are discrete approximations of continuous manifolds, which form non-linear subspaces of the original high-dimensional, ambient space which the data originally resides in. Therefore, recovering low-dimensional representations of phrases prior to embedding them in the graph means that the training data is a more dense sampling of the manifold structure, resulting in sample complexity benefits too, and this viewpoint is the second kind of dimensionality reduction that we discuss. In light of these observations, it is surprising that little previous work evaluating the various low-dimensional representations and how they improve graph quality and performance in various tasks exists, and none in MT. This chapter of the thesis aims to address these gaps.

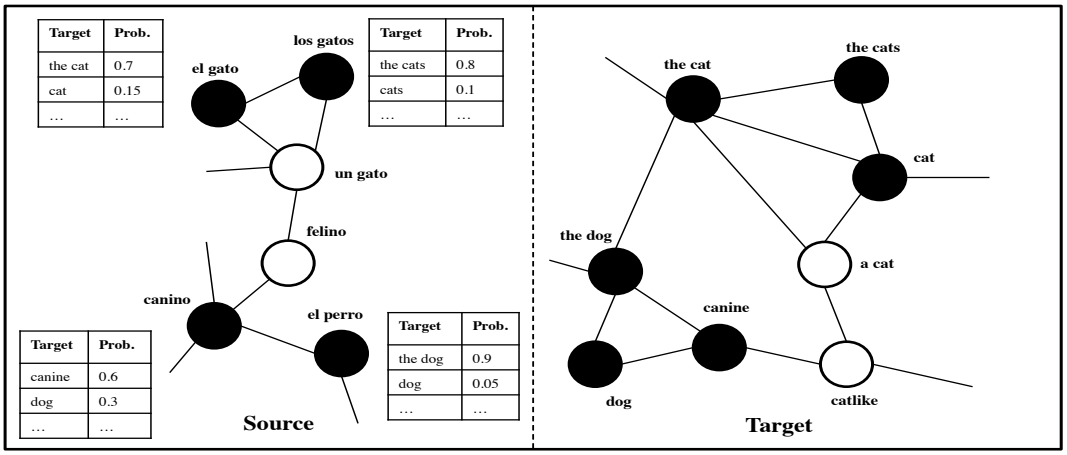


Figure 5.1: Example source and target graphs used in our approach. Labeled phrases on the source side are black (with their corresponding translations on the target side also black); unlabeled and generated (§5.1.1) phrases on the source and target sides respectively are white. Labeled phrases also have conditional probability distributions defined over target phrases, which are extracted from the parallel corpora.

5.1 Generation & Propagation

We first provide an overview of our graph-based framework for translation model expansion. While previous chapters (and most other work on this topic) have emphasized how a low-dimensional perspective can assist in parameter estimation and smoothing, in this section we present a technique that uses graph embeddings of phrases (via context) to increase the support of the translation model. Our goal is to obtain translation distributions for source phrases that are not present in the phrase table extracted from the parallel corpus. Both parallel and monolingual corpora are used to obtain these probability distributions over target phrases. We assume that sufficient parallel resources exist to learn a basic translation model using standard techniques, and also assume the availability of larger monolingual corpora in both the source and target languages. Although our technique applies to phrases of any length, in this work we concentrate on unigram and bigram phrases, which provides substantial computational cost savings.

Monolingual data is used to construct *separate* similarity graphs over phrases (word sequences), as illustrated in Fig. 5.1. The source similarity graph consists of phrase nodes representing sequences of words in the source language. If a source phrase is found in the baseline phrase table it is called a **labeled** phrase: its conditional empirical probability distribution over target phrases (estimated from the parallel data) is used as the label, and is subsequently never changed. Otherwise it is called an **unlabeled** phrase, and our algorithm finds labels (translations) for these unlabeled phrases, with the help of the graph-based representation.

The label space is thus the phrasal translation inventory, and like the source side it can also be represented in terms of a graph, initially consisting of target phrase nodes from the parallel corpus.

For the unlabeled phrases, the set of possible target translations could be extremely large (e.g., all target language n-grams). Therefore, we first **generate** and fix a list of possible target translations for each unlabeled source phrase. We then **propagate** by deriving a probability distribution over these target phrases using graph propagation techniques. Next, we will describe the generation, graph construction and propagation steps.

5.1.1 Generation

The objective of the generation step is to populate the target graph with *additional* target phrases for all unlabeled source phrases, yielding the full set of possible translations for the phrase. Prior to generation, one phrase node for each target phrase occurring in the baseline phrase table is added to the target graph (black nodes in Fig. 5.1's target graph). We only consider target phrases whose source phrase is a bigram, but it is worth noting that the target phrases are of variable length.

The generation component is based on the observation that for structured label spaces, such as translation candidates for source phrases in SMT, even similar phrases have slightly different labels (target translations). The exponential dependence of the sizes of these spaces on the length of instances is to blame. Thus, the target phrase inventory from the parallel corpus may be inadequate for unlabeled instances. We therefore need to enrich the target or label space for unknown phrases. A naïve way to achieve this goal would be to extract all n-grams, from $n = 1$ to a maximum n-gram order, from the monolingual data, but this strategy would lead to a combinatorial explosion in the number of target phrases.

Instead, by intelligently expanding the target space using linguistic information such as morphology (Toutanova et al., 2008, Chahuneau et al., 2013), or relying on the baseline system to generate candidates similar to self-training (McClosky et al., 2006), we can tractably propose novel translation candidates (white nodes in Fig. 5.1's target graph) whose probabilities are then estimated during propagation. We refer to these additional candidates as "generated" candidates.

To generate new translation candidates using the baseline system, we decode each unlabeled source bigram to generate its m-best translations. This set of candidate phrases is filtered to include only n-grams occurring in the target monolingual corpus, and helps to prune passed-through OOV words and invalid translations. To generate new translation candidates using morphological information, we morphologically segment words into prefixes, stem, and suffixes using linguistic resources. We assume that a morphological analyzer which provides context-independent analysis of word types exists, and implements the functions $\textsc{stem}(f)$ and $\textsc{stem}(e)$

for source and target word types. Based on these functions, source and target sequences of words can be mapped to sequences of stems. The morphological generation step adds to the target graph all target word sequences from the monolingual data that map to the same stem sequence as one of the target phrases occurring in the baseline phrase table. In other words, this step adds phrases that are morphological variants of existing phrases, differing only in their affixes.

5.1.2 Graph Construction

At this stage, there exists a list of source bigram phrases, both labeled and unlabeled, as well as a list of target language phrases of variable length, originating from both the phrase table and the generation step. To determine pairwise phrase similarities in order to embed these nodes in their graphs, we utilize the monolingual corpora on both the source and target sides to extract high-dimensional distributional features based on the context surrounding each phrase. For a phrase, we look at the p words before and the p words after the phrase, explicitly distinguishing between the two sides, but not distance (i.e., bag of words on each side). Co-occurrence counts for each feature (context word) are accumulated over the monolingual corpus, and these counts are converted to pointwise mutual information (PMI) values, as is standard practice when computing distributional similarities. Thus, each phrase is represented in a high-dimensional manner by a vector of PMI values, where each dimension is indexed by a context word and the side it occurs, and the dimensionality of the vector is roughly twice the vocabulary size.

Cosine similarity between two phrases' PMI vectors is used for similarity, and we take only the k most similar phrases for each phrase, to create a k-nearest neighbor similarity matrix for both source and target language phrases. These graphs are distinct, in that propagation happens within the two graphs but not between them. Representing a phrase in terms of its k-nearest neighbors and their respective similarities, instead of in the ambient space of size proportional to the number of types in the vocabulary, is the first kind of dimensionality reduction.

While accumulating co-occurrence counts for each phrase, we also maintain an inverted index data structure, which is a mapping from features (context words) to phrases that co-occur with that feature within a window of p.[3] The inverted index structure reduces the graph construction cost from $\theta(n^2)$, by only computing similarities for a subset of all possible pairs of phrases, namely other phrases that have at least one feature in common.

Using this graph-based framework, Zhao et al. (2015) studied the use of continuous representations for words to achieve the same objective i.e., generate translation rules for unknown or infrequent phrases. In particular, the authors used the continuous bag-of-words (CBOW)

[3] The q most frequent words in the monolingual corpus were removed as keys from this mapping, as these high entropy features do not provide much information.

model as implemented in `word2vec` (Mikolov et al., 2013), where the aim of the model is to predict a word based on its surrounding context. By maximizing the likelihood of a monolingual corpus with this model, a 300-dimensional representation for each word (the parameters of the model) are learned. For phrasal representations, Zhao et al. (2015) use simple element-wise addition as the compositional operator. In §5.2, we present results for graphs constructed in the ambient (high-dimensional) space, as well as in the 300-dimensional subspace recovered by the CBOW model.

5.1.3 Candidate Translation List Construction

As mentioned previously, we construct and fix a set of translation candidates, i.e., the label set for each unlabeled source phrase. The probability distribution over these translations is estimated through graph propagation, and the probabilities of items outside the list are assumed to be zero.

We obtain these candidates from two sources:[4]

1. The union of each unlabeled phrase's labeled neighbors' labels, which represents the set of target phrases that occur as translations of source phrases that are similar to the unlabeled source phrase. For *un gato* in Fig. 5.1, this source would yield *the cat* and *cat*, among others, as candidates.

2. The generated candidates for the unlabeled phrase – the ones from the baseline system's decoder output, or from a morphological generator (e.g., *a cat* and *catlike* in Fig. 5.1).

The morphologically-generated candidates for a given source unlabeled phrase are initially defined as the target word sequences in the monolingual data that have the same stem sequence as one of the baseline's target translations for a source phrase which has the same stem sequence as the unlabeled source phrase. These candidates are scored using stem-level translation probabilities, morpheme-level lexical weighting probabilities, and a language model, and only the top 30 candidates are included.

After obtaining candidates from these two possible sources, the list is sorted by forward lexical score, using the lexical models of the baseline system. The top r candidates are then chosen for each phrase's translation candidate list. In Figure 5.2 we provide example outputs of our system for a handful of unlabeled source phrases, and explicitly note the source of the translation candidate ('G' for generated, 'N' for labeled neighbor's label).

[4] We also obtained the k-nearest neighbors of the translation candidates generated through these methods by utilizing the target graph, but this had minimal impact.

5.1.4 Graph Propagation

A graph propagation algorithm transfers label information from labeled nodes to unlabeled nodes by following the graph's structure. In some applications, a label may consist of class membership information, e.g., each node can belong to one of a certain number of classes. In our problem, the "label" for each node is actually a probability distribution over a set of translation candidates (target phrases). For a given node f, let e refer to a candidate in the label set for node f; then in graph propagation, the probability of candidate e given source phrase f in iteration $t+1$ is:

$$\mathbb{P}^{t+1}(e|f) = \sum_{j \in \mathcal{N}(f)} T_s(j|f)\mathbb{P}^t(e|j) \tag{5.1}$$

where the set $\mathcal{N}(f)$ contains the (labeled and unlabeled) neighbors of node f, and $T_s(j|f)$ is a term that captures how similar nodes f and j are. This quantity is also known as the propagation probability, and its exact form will depend on the type of graph propagation algorithm used. For our purposes, node f is a source phrasal node, the set $\mathcal{N}(f)$ refers to other source phrases that are neighbors of f (restricted to the k-nearest neighbors as in §5.1.2), and the aim is to estimate $P(e|f)$, the probability of target phrase e being a phrasal translation of source phrase f.

A classic propagation algorithm that has been suitably modified for use in bilingual lexicon induction (Tamura et al., 2012, Razmara et al., 2013) is the **label propagation** (LP) algorithm of Zhu et al. (2003). In this case, $T_s(f, j)$ is chosen to be:

$$T_s(j|f) = \frac{w^s_{f,j}}{\sum_{j' \in \mathcal{N}(f)} w^s_{f,j'}} \tag{5.2}$$

where $w^s_{f,j}$ is the cosine similarity (as computed in §5.1.2) between phrase f and phrase j on side s (the source side).

As evident in Eq. 5.2, LP only takes into account source language similarity of phrases. To see this observation more clearly, let us reformulate Eq. 5.1 more generally as:

$$\mathbb{P}^{t+1}(e|f) = \sum_{j \in \mathcal{N}(f)} T_s(j|f) \sum_{e' \in \mathcal{H}(j)} T_t(e'|e)\mathbb{P}^t(e'|j) \tag{5.3}$$

where $\mathcal{H}(j)$ is the translation candidate set for source phrase j, and $T_t(e'|e)$ is the propagation probability between nodes or phrases e and e' on the *target* side. We have simply replaced $\mathbb{P}^t(e|j)$ with $\sum_{e' \in \mathcal{H}(j)} T_t(e'|e)\mathbb{P}^t(e'|j)$, defining it in terms of j's translation candidate list.

Note that in the original LP formulation the target side information is disregarded, i.e., $T_t(e'|e) = 1$ if and only if $e = e'$ and 0 otherwise. As a result, LP is suboptimal for our

needs, since it is unable to appropriately handle generated translation candidates for the unlabeled phrases. These translation candidates are usually not present as translations for the labeled phrases (or for the labeled phrases that neighbor the unlabeled one in question). When propagating information from the labeled phrases, such candidates will obtain no probability mass since $e \neq e'$. Thus, due to the setup of the problem, LP naturally biases away from translation candidates produced during the generation step (§5.1.1).

Structured Label Propagation

The label set we are considering has a similarity structure encoded by the target graph. How can we exploit this structure in graph propagation on the source graph? In Liu et al. (2012), the authors generalize label propagation to **structured label propagation** (SLP) in an effort to work more elegantly with structured labels. In particular, the definition of target similarity is similar to that of source similarity:

$$T_t(e'|e) = \frac{w^t_{e,e'}}{\sum_{e'' \in \mathcal{H}(j)} w^t_{e,e''}} \tag{5.4}$$

Therefore, the final update equation in SLP is:

$$\mathbb{P}^{t+1}(e|f) = \sum_{j \in \mathcal{N}(f)} T_s(j|f) \sum_{e' \in \mathcal{H}(j)} T_t(e'|e) \mathbb{P}^t(e'|j) \tag{5.5}$$

With this formulation, even if $e \neq e'$, the similarity $T_t(e'|e)$ as determined by the target phrase graph will dictate propagation probability. We re-normalize the probability distributions after each propagation step to sum to one over the fixed list of translation candidates, and run the SLP algorithm to convergence.[5]

5.1.5 Phrase-based SMT Expansion

After graph propagation, each unlabeled phrase is labeled with a categorical distribution over the set of translation candidates defined in §5.1.3. In order to utilize these newly acquired phrase pairs, we need to compute their relevant features. The phrase pairs have four log-probability features with two likelihood features and two lexical weighting features. In addition, we use a sophisticated lexicalized hierarchical reordering model (HRM) (Galley and Manning, 2008) with five features for each phrase pair.

We utilize the graph propagation-estimated forward phrasal probabilities $\mathbb{P}(e|f)$ as the forward likelihood probabilities for the acquired phrases; to obtain the backward phrasal probability

[5] Empirically within a few iterations and a wall-clock time of less than 10 minutes in total.

for a given phrase pair, we make use of Bayes' Theorem:

$$\mathbb{P}(f|e) = \frac{\mathbb{P}(e|f)\mathbb{P}(f)}{\mathbb{P}(e)}$$

where the marginal probabilities of source and target phrases e and f are obtained from the counts extracted from the monolingual data. The baseline system's lexical models are used for the forward and backward lexical scores. The HRM probabilities for the new phrase pairs are estimated from the baseline system by backing-off to the average values for phrases with similar length.

5.2 Evaluation

We performed an extensive evaluation to examine various aspects of the approach along with overall system performance. Two language pairs were used: Arabic-English and Urdu-English. The Arabic-English evaluation was used to validate the decisions made during the development of our method and also to highlight properties of the technique. With it, in §5.2.2 we first analyzed the impact of utilizing phrases instead of words and SLP instead of LP; the latter experiment underscores the importance of generated candidates. We also look at how adding morphological knowledge to the generation process can further enrich performance. In §5.2.3, we then examined the effect of using a very large 5-gram language model training on 7.5 billion English tokens to understand the nature of the improvements in §5.2.2. The Urdu to English evaluation in §5.2.4 focuses on how noisy parallel data and completely monolingual (i.e., not even comparable) text can be used for a realistic low-resource language pair, and is evaluated with the larger language model only. We also examine how our approach can learn from noisy parallel data compared to the traditional SMT system. Laslty, in §5.2.5, we compare graphs constructed with high-dimensional phrase representations compared to low-dimensional representations learned with the CBOW model.

Baseline phrasal systems are used both for comparison and for generating translation candidates for unlabeled phrases as described in §5.1.1. The baseline is a state-of-the-art phrase-based system; we perform word alignment using a lexicalized hidden Markov model, and then the phrase table is extracted using the `grow-diag-final` heuristic (Koehn et al., 2003). The 13 baseline features (2 lexical, 2 phrasal, 5 HRM, and 1 language model, word penalty, phrase length feature and distortion penalty feature) were tuned using MERT (Och, 2003), which is also used to tune the 4 feature weights introduced by the secondary phrase table (2 lexical and 2 phrasal, other features being shared between the two tables). For all systems, we use a distortion limit of 4. We use case-insensitive BLEU (Papineni et al., 2002) to evaluate translation quality.

5.2.1 Datasets

Bilingual corpus statistics for both language pairs are presented in Table 5.1. For Arabic-English, our training corpus consisted of 685k sentence pairs from standard LDC corpora[6]. The NIST MT06 and MT08 Arabic-English evaluation sets (combining the newswire and weblog domains for both sets), with four references each, were used as tuning and testing sets respectively. For Urdu-English, the training corpus was provided by the LDC for the NIST Urdu-English MT evaluation, and most of the data was automatically acquired from the web, making it quite noisy. After filtering, there are approximately 65k parallel sentences; these were supplemented by an additional 100k dictionary entries. Tuning and test data consisted of the MT08 and MT09 evaluation corpora, once again a mixture of news and web text.

Corpus	**Sentences**	**Words (Src)**
Ar-En Train	685,502	17,055,168
Ar-En Tune (MT06)	1,664	33,739
Ar-En Test (MT08)	1,360	42,472
Ur-En Train	165,159	1,169,367
Ur-En Tune (MT08)	1,864	39,925
Ur-En Test (MT09)	1,792	39,922

Table 5.1: Bilingual corpus statistics for the Arabic-English and Urdu-English datasets used.

Table 5.2 contains statistics for the monolingual corpora used in our experiments. From these corpora, we extracted all sentences that contained at least one source or target phrase match to compute features for graph construction. For the Arabic to English experiments, the monolingual corpora are taken from the AFP Arabic and English Gigaword corpora and are of a similar date range to each other (1994-2010), rendering them comparable but not sentence-aligned or parallel.

For the Urdu-English experiments, completely non-comparable monolingual text was used for graph construction; we obtained the Urdu side through a web-crawler, and a subset of the AFP Gigaword English corpus was used for English. In addition, we obtained a corpus from the ELRA[7], which contains a mix of parallel and monolingual data; based on timestamps, we extracted a comparable English corpus for the ELRA Urdu monolingual data to form a roughly 470k-sentence “noisy parallel" set. We used this set in two ways: either to augment the parallel data presented in Table 5.1, or to augment the non-comparable monolingual data in Table 5.2 for graph construction.

For the parameters introduced throughout the text, we present in Table 5.3 a reminder of their interpretation as well as the values used in this work.

[6] LDC2007T08 and LDC2008T09
[7] ELRA-W0038

Corpus	Sentences	Words
Ar Comparable	10.2m	290m
En I Comparable	29.8m	900m
Ur Noisy Parallel	470k	5m
En II Noisy Parallel	470k	4.7m
Ur Non-Comparable	7m	119m
En II Non-Comparable	17m	510m

Table 5.2: Monolingual corpus statistics for the Arabic-English and Urdu-English evaluations. The monolingual corpora can be sub-divided into comparable, noisy parallel, and non-comparable components. En I refers to the English side of the Arabic-English corpora, and En II to the English side of the Urdu-English corpora.

Parameter	Description	Value
m	m-best candidate list size when bootstrapping candidates in generation stage.	100
p	Window size on each side when extracting features for phrases.	2
q	Filter the q most frequent words when storing the inverted index data structure for graph construction. Both source and target sides share the same value.	25
k	Number of neighbors stored for each phrase for both source and target graphs. This parameter controls the sparsity of the graph.	500
r	Maximum size of translation candidate list for unlabeled phrases.	20

Table 5.3: Parameters, explanation of their function, and value chosen.

5.2.2 Experimental Variations

In our first set of experiments, we looked at the impact of choosing bigrams over unigrams as our basic unit of representation, along with performance of LP (Eq. 5.2) compared to SLP (Eq. 5.4). Recall that LP only takes into account source similarity; since the vast majority of generated candidates do not occur as labeled neighbors' labels, restricting propagation to the source graph drastically reduces the usage of generated candidates as labels, but does not completely eliminate it. In these experiments, we utilize a reasonably-sized 4-gram language model trained on 900m English tokens, i.e., the English monolingual corpus.

Table 5.4 presents the results of these variations; overall, by taking into account generated candidates appropriately and using bigrams ("SLP 2-gram"), we obtained a 1.13 BLEU gain on the test set. Using unigrams ("SLP 1-gram") actually does worse than the baseline, indicating the importance of focusing on translations for sparser bigrams. While LP ("LP 2-gram") does reasonably well, its underperformance compared to SLP underlines the importance of enriching the translation space with generated candidates and handling these candidates appropriately.[8]

[8] It is relatively straightforward to combine both unigrams and bigrams in one source graph, but for experimental clarity we did not mix these phrase lengths.

	BLEU	
Setup	**Tune**	**Test**
Baseline	39.33	38.09
SLP 1-gram	39.47	37.85
LP 2-gram	40.75	38.68
SLP 2-gram	41.00	39.22
SLP-HalfMono 2-gram	40.82	38.65
SLP+Morph 2-gram	41.02	39.35

Table 5.4: Results for the Arabic-English evaluation. The LP vs. SLP comparison highlights the importance of target side enrichment via translation candidate generation, 1-gram vs. 2-gram comparisons highlight the importance of emphasizing phrases, utilizing half the monolingual data shows sensitivity to monolingual corpus size, and adding morphological information results in additional improvement.

In "SLP-HalfMono", we use only half of the monolingual comparable corpora, and still obtain an improvement of 0.56 BLEU points, indicating that adding more monolingual data is likely to improve the system further. Interestingly, biasing away from generated candidates using all the monolingual data ("LP 2-gram") performs similarly to using half the monolingual corpora and handling generated candidates properly ("SLP-HalfMono").

Additional morphologically generated candidates were added in this experiment as detailed in §5.1.3. We used a simple hand-built Arabic morphological analyzer that segments word types based on regular expressions, and an English lexicon-based morphological analyzer. The morphological candidates add a small amount of improvement, primarily by targeting genuine OOVs.

5.2.3 Large Language Model Effect

In this set of experiments, we examined if the improvements in §5.2.2 can be explained primarily through the extraction of language model characteristics during the semi-supervised learning phase, or through orthogonal pieces of evidence. Would the improvement be less substantial had we used a very large language model?

To answer this question we trained a 5-gram language model on 570M sentences (7.6B tokens), with data from various sources including the Gigaword corpus[9], WMT and European Parliamentary Proceedings[10], and web-crawled data from Wikipedia and the web. Only m-best generated candidates from the baseline were considered during generation, along with labeled neighbors' labels.

[9] LDC2011T07
[10] http://www.statmt.org/wmt13/

	BLEU	
Setup	**Tune**	**Test**
Baseline+LargeLM	41.48	39.86
SLP+LargeLM	42.82	41.29

Table 5.5: Results with the large language model scenario. The gains are even better than with the smaller language model.

Table 5.5 presents the results of using this language model. We obtained a robust, 1.43-BLEU point gain, indicating that the addition of the newly induced phrases provided genuine translation improvements that cannot be compensated by the language model effect. Further examination of the differences between the two systems yielded that most of the improvements are due to better bigrams and trigrams, as indicated by the breakdown of the BLEU score precision per n-gram, and primarily leverages higher quality generated candidates from the baseline system. We analyze the output of these systems further in the output analysis section below (§5.2.6).

5.2.4 Urdu-English

In order to evaluate the robustness of these results beyond one language pair, we looked at Urdu-English, a low resource pair likely to benefit from this approach. In this set of experiments, we used the large language model in §5.2.3, and only used baseline-generated candidates. In the Arabic-English setup we have access to comparable corpora through time-stamped data such as AFP from the GigaWord corpus on both sides, but we do not have access to such data for Urdu-English. We experimented with two extreme setups that differed in the data assumed parallel, from which we built our baseline system, and the data treated as monolingual, from which we built our source and target graphs.

	BLEU	
Setup	**Tune**	**Test**
Baseline	21.87	21.17
SLP+Noisy	26.42	25.38
Baseline+Noisy	27.59	27.24
SLP	28.53	28.43

Table 5.6: Results for the Urdu-English evaluation evaluated with BLEU. All experiments were conducted with the larger language model, and generation only considered the m-best candidates from the baseline system.

In the first setup, we use the noisy parallel data for graph construction and augment the non-comparable corpora with it:

- parallel: "Ur-En Train"
- Urdu monolingual: "Ur Noisy Parallel"+"Ur Non-Comparable"
- English monolingual: "En II Noisy Parallel"+"En II Non-Comparable"

The results from this setup are presented as "Baseline" and "SLP+Noisy" in Table 5.6. In the second setup, we train a baseline system using the data in Table 5.1, augmented with the noisy parallel text:

- parallel: "Ur-En Train"+"Ur Noisy Parallel"+"En II Noisy Parallel"
- Urdu monolingual: "Ur Non-Comparable"
- English monolingual: "En II Non-Comparable"

The results from this setup are presented as "Baseline+Noisy" and "SLP" in Table 5.6. The two setups allow us to examine how effectively our method can learn from the noisy parallel data by treating it as monolingual (i.e., for graph construction), compared to treating this data as parallel, and also examines the realistic scenario of using completely non-comparable monolingual text for graph construction as in the second setup.

In the first setup, we get a huge improvement of 4.2 BLEU points ("SLP+Noisy") when using the monolingual data and the noisy parallel data for graph construction. Our method obtained much of the gains achieved by the supervised baseline approach that utilizes the noisy parallel data in conjunction with the NIST-provided parallel data ("Baseline+Noisy"), but with fewer assumptions on the nature of the corpora (monolingual vs. parallel). Furthermore, despite completely un-aligned, non-comparable monolingual text on the Urdu and English sides, and a very large language model, we can still achieve gains in excess of 1.2 BLEU points ("SLP") in a difficult evaluation scenario, which shows that the technique adds a genuine translation improvement over and above naïve memorization of n-gram sequences.

5.2.5 Low-Dimensional Graphs

In Zhao et al. (2015), the authors ran experiments on the same corpora, and compared SLP with the high-dimensional, PMI-based representations and SLP with continuous representations. Table 5.7 presents results for both language pairs. Numerically, the Arabic-English results are very close to Table 5.4, and we find that the continous representations improve upon the high-dimensional representations significantly, by 0.2-0.3 BLEU on the tune and test sets. For the Urdu-English results, the baseline setup and SLP results are ≈ 1.25 BLEU points lower on the tune set than in Table 5.6, but the overall trend is the same. In this instance, continuous representations have a negligible improvement over the high-dimensional ones, so this phenomenon does seem to vary in magnitude by corpus or language pair.

	Setup	BLEU Tune	BLEU Test
AR-EN	Baseline	39.33	38.09
	SLP w/ PMI	40.93	39.16
	SLP w/ CONTINUOUS	41.31	39.34
UR-EN	Baseline	26.32	27.41
	SLP w/ PMI	27.26	27.89
	SLP w/ continuous	27.34	27.73

Table 5.7: A comparison between using PMI-based graphs (high-dimensional) and continuous representation-based graphs (low-dimensional) across two language pairs, from Zhao et al. (2015).

Ex	Source	Reference	Baseline	System
1 (Ar)	ارسال التعزيزات	sending reinforcements	strong reinforcements	sending reinforcements (N)
2 (Ar)	ل+ الاندثار	with extinction	OOV	with extinction (N)
3 (Ar)	تحبط محاولة	thwarts	address	thwarted (N)
4 (Ar)	نسبت الي	was quoted as saying	attributed to	was quoted as saying (G)
5 (Ar)	أوضح عبد المحمود	abdalmahmood said	he said abdul mahmood	mahmood said (G)
6 (Ar)	تراه منكبا	it deems	OOV	it deems (G)
7 (Ur)	پر امید	I am hopeful	this hope	I am hopeful (N)
8 (Ur)	اپنا دفاع	to defend him	to defend	to defend himself (G)
9 (Ur)	گفتگو کی۔	while speaking	In the	in conversation (N)

Figure 5.2: Nine example outputs of our system vs. the baseline highlighting the properties of our approach. Each example is labeled (Ar) for Arabic source or (Ur) for Urdu source, and system candidates are labeled with (N) if the candidate unlabeled phrase's labeled neighbor's label, or (G) if the candidate was generated.

5.2.6 Analysis of Output

Figure 5.2 looks at some of the sample hypotheses produced by our system and the baseline, along with reference translations. The outputs produced by our system are additionally annotated with the origin of the candidate, i.e., labeled neighbor's label (N) or generated (G).

The Arabic-English examples are numbered 1 to 5. The first example shows a source bigram unknown to the baseline system, resulting in a suboptimal translation, while our system proposes the correct translation of "sending reinforcements". The second example shows a word that was an OOV for the baseline system, while our system got a perfect translation. The third and fourth examples represent bigram phrases with much better translations compared to backing off to the lexical translations as in the baseline. The fifth Arabic-English example demonstrates the pitfalls of over-reliance on the distributional hypothesis: the source bigram corresponding to the name "abd almahmood" is distributional similar to another named entity "mahmood" and the English equivalent is offered as a translation. The distributional hypoth-

esis can sometimes be misleading. The sixth example shows how morphological information can propose novel candidates: an OOV word is broken down to its stem via the analyzer and candidates are generated based on the stem.

The Urdu-English examples are numbered 7 to 9. In example 7, the bigram “par umeed" (corresponding to “hopeful") is never seen in the baseline system, which has only seen “umeed" (“hope”). By leveraging the monolingual corpus to understand the context of this unlabeled bigram, we can utilize the graph structure to propose a syntactically correct form, also resulting in a more fluent and correct sentence as determined by the language model. Examples 8 & 9 show cases where the baseline deletes words or translates them into more common words e.g., “conversation" to “the", while our system proposes reasonable candidates.

5.3 Related Work

The idea presented in this paper is similar in spirit to bilingual lexicon induction (BLI), where a seed lexicon in two different languages is expanded with the help of monolingual corpora, primarily by extracting distributional similarities from the data using word context. This line of work, initiated by Rapp (1995) and continued by others (Fung and Yee, 1998, Koehn and Knight, 2002) (*inter alia*) is limited from a downstream perspective, as translations for only a small number of words are induced and oftentimes for common or frequently occurring ones only. Recent improvements to BLI (Tamura et al., 2012, Irvine and Callison-Burch, 2013b) have contained a graph-based flavor by presenting label propagation-based approaches using a seed lexicon, but evaluation is once again done on top-1 or top-3 accuracy, and the focus is on unigrams.

Razmara et al. (2013) and Irvine and Callison-Burch (2013a) conduct a more extensive evaluation of their graph-based BLI techniques, where the emphasis and end-to-end BLEU evaluations concentrated on OOVs, i.e., unigrams, and not on enriching the entire translation model. As with previous BLI work, these approaches only take into account source-side similarity of words; only moderate gains (and in the latter work, on a subset of language pairs evaluated) are obtained. Additionally, because of our structured propagation algorithm, our approach is better at handling multiple translation candidates and does not need to restrict itself to the top translation.

Klementiev et al. (2012) propose a method that utilizes a pre-existing phrase table and a small bilingual lexicon, and performs BLI using monolingual corpora. The operational scope of their approach is limited in that they assume a scenario where unknown phrase pairs are provided (thereby sidestepping the issue of translation candidate generation for completely unknown phrases), and what remains is the estimation of phrasal probabilities. In our case, we obtain the phrase pairs from the graph structure (and therefore indirectly from the monolingual data) and a separate generation step, which plays an important role in good performance of

the method. Similarly, Zhang and Zong (2013) present a series of heuristics that are applicable in a fairly narrow setting.

The notion of translation consensus, wherein similar sentences on the source side are encouraged to have similar target language translations, has also been explored via a graph-based approach (Alexandrescu and Kirchhoff, 2009). Liu et al. (2012) extend this method by proposing a novel structured label propagation algorithm to deal with the generalization of propagating *sets* of labels instead of single labels, and also integrated information from the graph into the decoder. In fact, we utilize this algorithm in our propagation step (§5.1.4). However, the former work operates only at the level of sentences, and while the latter does extend the framework to sub-spans of sentences, they do not discover new translation pairs or phrasal probabilities for new pairs at all, but instead re-estimate phrasal probabilities using the graph structure and add this score as an additional feature during decoding.

The goal of leveraging non-parallel data in machine translation has been explored from several different angles. Paraphrases extracted by "pivoting" via a third language (Callison-Burch et al., 2006) can be derived solely from monolingual corpora using distributional similarity (Marton et al., 2009). Snover et al. (2008) use cross-lingual information retrieval techniques to find potential sentence-level translation candidates among comparable corpora. In this case, the goal is to try and construct a corpus as close to parallel as possible from comparable corpora, and is a fairly different take on the problem we are looking at. Decipherment-based approaches (Ravi and Knight, 2011, Dou and Knight, 2012) have generally taken a monolingual view to the problem and combine phrase tables through the log-linear model during feature weight training.

Lastly, Blake Shaw's thesis on non-linear dimensionality reduction and graph embedding (Shaw, 2011) deals directly with the case where a graph is provided *a priori*, to which non-linear dimensionality reduction techniques are applied. He argues that linear dimensionality reduction techniques like SVD are inappropriate for this task, unless it so happens that the manifold we are trying to approximate with the graph is a linear one. The approach is slightly different than the results in Table 5.7, where low-dimensional representations of words and phrases are obtained prior to graph construction i.e., the size of the ambient space is reduced prior to estimating the manifold.

5.4 Summary

In this chapter, we presented an approach that can expand a translation model extracted from a sentence-aligned, bilingual corpus using a large amount of unstructured, monolingual data in both source and target languages, which leads to improvements of 1.4 and 1.2 BLEU points over strong baselines on evaluation sets, and in some scenarios gains in excess of 4 BLEU points. The framework is used to investigate the low-dimensional hypothesis for context in the

semi-supervised setting, and not only smoothens parameter estimates using low-dimensional context, but actually expands the support of the model and adds new translation rules. Two views of low-dimensional context are used: a graph embedding perspective, where phrases are represented and reasoned about using solely their neighborhood information and form a lower-dimensional manifold in a higher-dimensional ambient space, and an actual dimensionality reduction of the ambient space itself prior to manifold estimation.

For future work, we wish to apply a decomposition algorithm (like eigendecomposition or SVD) to the Laplacian, a graph quantity derived from the similarity matrix, and truncate the dimensions to a pre-specified hyperparameter (§2.3.3). While such a truncation does not alter the basic propagation algorithm since we still express each point in terms of its nearest neighbors, the edge weights between nodes will be altered since the original weights were estimated in the high-dimensional, ambient space while the truncated Laplacian-based representations more accurately convey geodesic distances. Second, our graph propagation algorithms are based on the label propagation variant of Zhu and Ghahramani (2002); in such approaches, the distribution for each labeled node is kept fixed, and only the unlabeled nodes' distributions are updated. Other approaches relax this constraint, and allow re-estimation of the labeled nodes' distributions as well (Belkin et al., 2006). This relaxation is worth considering, especially when the original labeled nodes' distributions are noisy.

Chapter 6

Conclusion

"Game done changed."
"Game's the same, just got more fierce"

— Cutty and Slim Charles, The Wire

Due to the sheer complexity and infinite capacity of language, it is only natural that a key way to resolve ambiguities is to condition decisions on the context in which such decisions occur. While this observation has been leveraged in translation, the prevailing approaches have been somewhat limited in the extent to which context is incorporated. This thesis has argued that context's effect on translation is low-dimensional, and an appropriate form of dimensionality reduction or representation should be utilized when learning translation models.

The contributions of this thesis can be grouped into two broad areas: foundations and applications. Within foundations, we proposed a bilingual generalization of the L-PCFG model, L-SCFGs, with corresponding generalizations to the inference algorithm and two parameter estimation algorithms. The L-SCFG model can be seen as learning low-dimensional representations of translation rules in terms of the context in which they occur. We also proposed two sets of methods that directly learn to disambiguate translations by operating in a low-dimensional context-translation rule space. Lastly, we presented a graph-based SSL method that first embeds source and target language phrases in two separate graphs, and then operates only over the graph structure to expand the support of translation models i.e., add new rules to the inventory along with estimates of their scores.

There are a number of direct applications as a result of this thesis. The first is empirical validation that marginal rule probabilities from L-SCFG models improve translation, to the extent that translation models based on minimal grammars can match or even exceed the performance of those models based on composed grammars. We also thoroughly investigated the application of various translation disambiguation models, both high and low-dimensional

and found small improvements in most settings. In the semi-supervised setting, we showed that the graph-based framework is extremely effective in expanding translation models and improving overall translation quality. Lastly, all of the code in this thesis has been released in three separate packages: `spectral-scfg`, `cca-mt`, and `graphMT`.

As with any framework, there are a number of limitations to our approach, some of which may be inherent to the basic assumptions we have used, and others due to the limitations in our experimental procedures. Firstly, the multi-view assumption acts as a noise reduction mechanism since noise in the two views is assumed to be uncorrelated and will be removed by the resulting projection matrices. However, when using word alignments with parallel corpora to segment the source sentence into phrases and considering the lexical context surrounding phrases, the noise is not independent. Furthermore, it can be argued that the manifolds upon which we build our manifold assumption are not quite low-dimensional manifolds: by using a one-hot representation for the original ambient space, the manifold is more akin to a relatively high-dimensional hypercube. Yet, the fact remains that there is an underlying structure to the distribution of examples (phrases) in this space, and it is this structure that we seek to exploit by utilizing the manifold assumption. We used minimal grammars as the basis of our translation inventory in order to explore the incorporation of (low-dimensional) context through auxiliary models. However, shifting to these grammars introduces another moving part during the translation process, namely that the search space considered by the translation decoder is vastly different to a more traditional search space based on compositional models. This observation may be partly responsible for the large gap in performance between minimal and composed grammar baselines, and adjusting the beam size during decoding to consider more translation options is the ideal solution to normalize for such a difference.

6.1 Future Work

Within each chapter of this thesis, we have discussed possible extensions to the relevant subject matter. At a higher level, we now consider several more extensions of this work that are more speculative than previously considered ones.

Label smoothing The focus of this thesis has been on low-dimensional representations of basic linguistic units: whether they be hierarchical translation rules, phrase pairs, or the context itself. Many of the ideas that we have used for dimensionality reduction can also be used for function *smoothing* i.e., applied to the *label space* and not the data space as has been done in this thesis. In NLP and especially in translation, the label functions for linguistic units are extremely high-dimensional (because often the labels are combinatorial structures) and complex, so it is natural to apply smoothness criteria to these functions that also respect the structure of the label space. Some of these ideas have been explored before to

a limited extent e.g., dimensionality reduction methods like Laplacian eigenmaps that learn smooth functions over the graph (respecting graph structure) by truncating extraneous basis directions from the spectrum of the graph Laplacian (Belkin and Niyogi, 2003). More formally, the field of information geometry (Amari and Nagaoka, 2007) provides an extensive toolbox to tackle problem in the label space. Information geometry applies differential geometry to the space of probability distributions: distributions for a model family are points of a Riemannian manifold i.e., a statistical manifold. Since the objective during learning is to select a set of parameters from this model family that best matches observed data i.e., select a point from the statistical manifold, it is important to consider the structure of this manifold during learning. Thus, information geometric approaches consider smoothed, low-dimensional versions of label spaces. In particular, dimensionality reduction that operates on the probability simplex instead of the ambient Euclidean space have been proposed (Carter et al., 2008). These algorithms take into account the stochastic nature of each point on the manifold, and information divergence (computing using Kullback-Leibler divergence) instead of Euclidean distance is the measure of dissimilarity.

Other types of context In this thesis, we considered two different types of context: structured and unstructured. In Ch. 3, we used the minimal derivation trees provided to us during training to provide a structured, hierarchical form of context in the form of inside and outside trees. In Chs. 4 and 5, we used unstructured context in the form of windows around phrases that we were interested in representing. We can consider other forms of context that operate at a *global* level and leverage information at the granularity of paragraphs or even documents, which is particularly relevant in MT since translation systems often translate on a per-document basis. For example, topic information for a document can be extracted using unsupervised Bayesian approaches like latent Dirichlet allocation (Blei et al., 2003). Recently, paragraph-level representations have also been explored (Le and Mikolov, 2014), and thus a natural extension of this thesis would be to incorporate context at this granularity into translation models. Certain signals may be easily extractable from paragraph and document-level information, but may not be present when making local translation disambiguation decisions, and this information should always be conditioned on when translating.

More than two views Many of the approaches presented in this thesis make use of the multi-view assumption, but arguably this assumption is used to a rather limited extent. In all cases, we actually make use of a "two-view" assumption, and therefore a logical extension to the thesis is to truly evaluate the multi-view assumption. In both the L-SCFG models from Ch. 3 and the disambiguation models in Ch. 4, the true multi-view assumption necessitates the usage of higher-order tensor structures Of course, instead of CCA we need to make use of a generalized notion of this technique that operates on tensors and not matrices (Horst, 1961, Jain and Oh, 2014, Luo et al., 2015, *inter alia*); such techniques can be computationally intense,

but algorithms for approximate computation e.g., Rastogi et al. (2015) have been proposed to allay this difficulty. Through a generalized version of CCA, complementary signals that shed light on the correct form and function of phrases in context can be used e.g., lexical, part-of-speech, and semantic role labels can be combined. Topic information and other signals that operate at a higher level can also be incorporated.

Combining the assumptions As mentioned in §2.3.4, inference over a graph can be seen as a random walk traversal on the graph. Using the dynamics of these random walks over the entire graph structure, we can come up with a notion of distance on the graph, similar to geodesic distance in that it respects graph structure, but is more related to the random walk dynamics on the graph: the diffusion distance (Coifman and Lafon, 2006). Recently, Lindenbaum et al. (2015) has proposed a way of computing diffusion distances between examples while directly taking into account that each datapoint can have multiple views attributed to it. In particular, they constrain the diffusion process such that a random walk on a node representing one particular view can only transition to a node representing the other view. This desideratum can be easily achieved by imposing a particular block structure on the random walk matrix. Subsequently, previously proposed techniques can be used to extract diffusion distances from this matrix. Lindenbaum et al. (2015) thus present an interesting perspective on combining the two major assumptions used in this thesis, and exploring these techniques in conjunction with some of the ideas presented here would be an interesting avenue for future exploration.

Bibliography

Andrei Alexandrescu and Katrin Kirchhoff. Graph-based learning for statistical machine translation. In *Proceedings of NAACL*, 2009.

Shun-ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*. American Mathematical Society, 2007.

Rie Kubota Ando and Tong Zhang. Two-view feature generation model for semi-supervised learning. In *Proceedings of ICML*, 2007.

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *Proceedings of ICML*, 2013.

Haim Avron, Christos Boutsidis, Sivan Toledo, and Anastasios Zouzias. Efficient dimensionality reduction for canonical correlation analysis. In *Proceedings of ICML*, 2013.

Lalit R. Bahl, Frederick Jelinek, and Robert L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190, 1983.

Mikahil Belkin. *Problems of Learning on Manifolds*. PhD thesis, University of Chicago, 2003.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2008.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. Label propagation and quadratic criterion. In *Semi-Supervised Learning*, pages 193–216. MIT Press, 2006.

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. Painless unsupervised learning with features. In *Proceedings of NAACL*, 2010.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

Ake Björck and Gene H. Golub. Numerical methods for computing angles between linear subspaces. *AMS Mathematics of Computation*, 27:579–594, 1973.

David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of COLT*, 1998.

Phil Blunsom and Miles Osborne. Probabilistic inference for machine translation. In *Proceedings of EMNLP*, 2008.

Phil Blunsom, Trevor Cohn, and Miles Osborne. A discriminative latent variable model for statistical machine translation. In *Proceedings of ACL*, 2008a.

Phil Blunsom, Trevor Cohn, and Miles Osborne. Bayesian synchronous grammar induction. In *Proceedings of NIPS*, 2008b.

Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Frederick Jelinek, John. D Lafferty, Robert. L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):256–264, 1990.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. Improved statistical machine translation using paraphrases. In *Proceedings of NAACL*, 2006.

Marine Carpuat. *Word Sense Disambiguation for Statistical Machine Translation*. PhD thesis, Hong Kong University of Science and Technology, 2008.

Marine Carpuat and Dekai Wu. Word sense disambiguation vs. statistical machine translation. In *Proceedings of ACL*, 2005.

Marine Carpuat and Dekai Wu. Improving statistical machine translation using word sense disambiguation. In *Proceedings of EMNLP-CoNLL*, 2007.

Marine Carpuat, Hal Daume III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi, and

Rachel Rudinger. Sensespotting: Never let your parallel data tie you to an old domain. In *Proceedings of ACL*, 2013.

Kevin M. Carter, Raviv Raich, and Alfed O. Hero III. An information geometric framework for dimensionality reduction. arXiv:0809.4866, 2008.

Victor Chahuneau, Eva Schlinger, Noah A. Smith, and Chris Dyer. Translating into morphologically rich languages with synthetic phrases. In *Proceedings of EMNLP*, 2013.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. Word sense disambiguation improves statistical machine translation. In *Proceedings of ACL*, 2007.

Jean-Cédric Chappelier and Martin Rajman. A generalized CYK algorithm for parsing stochastic CFG. In *Proceedings of TAPD*, 1998.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999.

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

David Chiang. Hope and fear for discriminative training of statistical translation models. *Journal of Machine Learning Research*, 13(1):1159–1187, 2012.

David Chiang, Kevin Knight, and Wei Wang. 11,001 new features for statistical machine translation. In *Proceedings of NAACL*, 2009.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of EMNLP*, 2014.

Fan R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of ACL*, 2011.

Shay B. Cohen and Michael Collins. Tensor decomposition for fast parsing with latent-variable PCFGs. In *Proceedings of NIPS*, 2012.

Shay B. Cohen and Michael Collins. A provably correct learning algorithm for latent-variable PCFGs. In *Proceedings of ACL*, 2014.

Shay B. Cohen, Karl Stratos, Michael Collins, Dean P. Foster, and Lyle Ungar. Spectral learning of latent-variable PCFGs: Algorithms and sample complexity. *Journal of Machine Learning Research*, 15:2399–2449, 2014.

Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21:5–30, 2006.

Dipanjan Das and Slav Petrov. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of ACL*, 2011.

Hal Daumé III and Jagadeesh Jagarlamudi. Domain adaptation for machine translation by mining unseen words. In *Proceedings of ACL*, 2011.

Arthuer 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*, 39(1): 1–38, 1977.

John DeNero. *Phrase Alignment Models for Statistical Machine Translation*. PhD thesis, University of California, Berkeley, 2010.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In *Proceedings of ACL*, 2014.

Paramveer S. Dhillon, Dean Foster, and Lyle Ungar. Multi-view learning of word embeddings via CCA. In *Proceedings of NIPS*, 2011.

Paramveer S. Dhillon, Jordan Rodu, Dean P. Foster, and Lyle H. Ungar. Two step CCA: a new spectral method for estimating vector models of words. In *Proceedings of ICML*, 2012.

Manfredo Perdigão Do Carmo. *Riemannian Geometry*. Birkhäuser, 1992.

Qing Dou and Kevin Knight. Large scale decipherment for out-of-domain machine translation. In *Proceedings of EMNLP*, 2012.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Nadir Durrani, Helmut Schmid, and Alexander Fraser. A joint sequence translation model with integrated reordering. In *Proceedings of ACL*, 2011.

Chris Dyer. *A Formal Model of Ambiguity and its Applications in Machine Translation*. PhD thesis, University of Maryland, College Park, 2010.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan 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.

Yang Feng and Trevor Cohn. A markov model of machine translation using non-parametric bayesian inference. In *Proceedings of ACL*, 2013.

Dean P. Foster, Sham M. Kakade, and Tong Zhang. Multi-view dimensionality reduction via canonical correlation analysis. Technical report, Toyota Technological Institute (TTI), 2008.

Pascale Fung and Lo Yuen Yee. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of ACL*, 1998.

Michel Galley and Christopher D. Manning. A simple and effective hierarchical phrase reordering model. In *Proceedings of EMNLP*, 2008.

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

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of ACL*, 2006.

Zoubin Ghahramani. One hidden layer linear networks and canonical correlations. http://mlg.eng.cam.ac.uk/zoubin/papers/cancorr.pdf, 1996.

Kevin Gimpel and Noah A. Smith. Rich source-side context for statistical machine translation. In *Proceedings of WMT*, 2008.

Yoav Goldberg and Omer Levy. word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. arXiv:1402.3722, 2014.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1996.

Jonathan Graehl, Kevin Knight, and Jonathan May. Training tree transducers. *Computational Linguistics*, 34(3):391–427, 2008.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL*, 2008.

N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM*, 53(2):217–288, 2011.

Greg Hanneman and Alon Lavie. Improving syntax-augmented machine translation by coarsening the label set. In *Proceedings of NAACL*, 2013.

David R. Hardoon, Sandor R. Szedmak, and John R. Shawe-taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12): 2639–2664, 2004.

Zhongjun He, Qun Liu, and Shouxun Lin. Improving statistical machine translation using lexicalized rule selection. In *Proceedings of COLING*, 2008.

Kenneth Heafield. Kenlm: Faster and smaller language model queries. In *Proceedings of WMT*, 2011.

Matthias Hein, Jean yves Audibert, and Ulrike Von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proceedings of COLT*, 2005.

Matthias Hein, Jean yves Audibert, and Ulrike Von Luxburg. Graph Laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8: 1325–1370, 2007.

Paul Horst. Generalized canonical correlations and their applications to experimental data. *Journal of Clinical Psychology*, 17(4), 1961.

Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28:312–377, 1936.

Daniel Hsu, Sham M. Kakade, and Tong Zhang. A spectral algorithm for learning hidden Markov models. In *Proceedings of COLT*, 2009.

Liang Huang, Kevin Knight, and Aravind Joshi. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*, August 2006.

Zhongqiang Huang, Martin Čmejrek, and Bowen Zhou. Soft syntactic constraints for hierarchical phrase-based translation using latent syntactic distributions. In *Proceedings of EMNLP*, 2010.

Ann Irvine and Chris Callison-Burch. Supervised bilingual lexicon induction with multiple monolingual signals. In *Proceedings of NAACL*, 2013a.

Ann Irvine and Chris Callison-Burch. Combining bilingual and comparable corpora for low resource machine translation. In *Proceedings of WMT*, 2013b.

Herbert Jaeger. Observable Operator Models for Discrete Stochastic Time Series. *Neural Computation*, 12(6):1371–1398, 2000.

Prateek Jain and Sewoong Oh. Provable tensor factorization with missing data. In *Proceedings of NIPS*, 2014.

Sham M. Kakade and Dean P. Foster. Multi-view regression via canonical correlation analysis. In *Proceedings of COLT*, 2007.

Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of EMNLP*, 2013.

Dan Klein and Christopher D. Manning. Parsing and hypergraphs. In *Proceedings of IWPT*, 2001.

Alexandre Klementiev, Ann Irvine, Chris Callison-Burch, and David Yarowsky. Toward statistical machine translation without parallel corpora. In *Proceedings of EACL*, 2012.

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

Philipp Koehn and Kevin Knight. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL Workshop on Unsupervised Lexical Acquisition*, 2002.

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

Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM*, 51 (3):455–500, 2009.

Erwin Kreyszig. *Introductory Functional Analysis with Applications*. Wiley, 1989.

Stéphane Lafon. *Diffusion Maps and Geometric Harmonics*. PhD thesis, Yale University, 2004.

Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of ICML*, 2014.

Abby Levenberg, Chris Dyer, and Phil Blunsom. A Bayesian model for learning SCFGs with discontiguous rules. In *Proceedings of EMNLP-CoNLL*, 2012.

P. M. Lewis, II and R. E. Stearns. Syntax-directed transduction. *Journal of the ACM*, 15(3): 465–488, 1968.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. An end-to-end discriminative approach to machine translation. In *Proceedings of ACL*, 2006.

Percy Liang, Slav Petrov, Michael I. Jordan, and Dan Klein. The infinite PCFG using hierarchical dirichlet processes. In *Proceedings of EMNLP*, 2007.

Ofir Lindenbaum, Arie Yeredor, Moshe Salhov, and Amir Averbuch. Multiview diffusion maps. arXiv:1508.05550, 2015.

Shujie Liu, Chi-Ho Li, Mu Li, and Ming Zhou. Learning translation consensus with structured label propagation. In *Proceedings of ACL*, 2012.

Adam Lopez. Statistical machine translation. *ACM Computing Surveys*, 40(3):1–49, 2008.

David Lopez-Paz, Suvrit Sra, Alex Smola, Zoubin Ghahramani, and Bernhard Schölkopf. Randomized nonlinear component analysis. In *Proceedings of ICML*, 2014.

Yichao Lu and Dean P Foster. Large scale canonical correlation analysis with iterative least squares. In *Proceedings of NIPS*, 2014.

Yong Luo, Dacheng Tao, Yonggang Wen, Kotagiri Ramamohanarao, and Chao Xu. Tensor canonical correlation analysis for multi-view dimension reduction. arXiv:1502.02330, 2015.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP*, 2015.

Daniel Marcu and William Wong. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of EMNLP*, 2002.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Kanti V. Mardia, John T. Kent, and John M. Bibby. *Multivariate Analysis*. New York Academic Press, 1979.

José B. Mariño, Rafael E. Banchs, Josep M. Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-jussà. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, 2006.

Yuval Marton, Chris Callison-Burch, and Philip Resnik. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of EMNLP*, 2009.

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. Probabilistic CFG with latent annotations. In *Proceedings of ACL*, 2005.

David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of NAACL*, 2006.

I. Dan Melamed. Statistical machine translation by parsing. In *Proceedings of ACL*, 2004.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv:1301.3781, 2013.

Paul Mineiro and Nikos Karampatziakis. A randomized algorithm for CCA. arXiv:1411.3409, 2014.

Markos Mylonakis and Khalil Sima'an. Learning hierarchical translation structure with linguistic annotations. In *Proceedings of ACL*, 2011.

Shashi Narayan and Shay B. Cohen. Diversity in Spectral Learning for Natural Language Parsing. In *Proceedings of EMNLP*, 2015.

Hermann Ney, Ute Essen, and Reinhard Kneser. On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38, 1994.

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Proceedings of NIPS*, 2002.

Partha Niyogi. Manifold regularization and semi-supervised learning: Some theoretical analyses. *Journal of Machine Learning Research*, 14:1229–1250, 2013.

Franz Josef Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL*, 2003.

Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL*, 2002.

Franz Josef Och and Hermann Ney. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(4):417–449, 2004.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, 2002.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of ICML*, 2013.

Michael Paul. Overview of the IWSLT 2009 evaluation campaign. In *Proceedings of IWSLT*, 2009.

Karl Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of ACL*, 2006.

William H. Press. Canonical correlation clarified by singular value decomposition. http://www.nr.com/whp/notes/CanonCorrBySVD.pdf, 2011.

Reinhard Rapp. Identifying word translations in non-parallel texts. In *Proceedings of ACL*, 1995.

Pushpendre Rastogi, Benjamin Van Durme, and Raman Arora. Multiview lsa: Representation learning via generalized cca. In *Proceedings of NAACL*, 2015.

Sujith Ravi and Kevin Knight. Deciphering foreign language. In *Proceedings of ACL*, 2011.

Majid Razmara, Maryam Siahbani, Gholamreza Haffari, and Anoop Sarkar. Graph propagation for paraphrasing out-of-vocabulary words in statistical machine translation. In *Proceedings of ACL*, 2013.

Roman Rosipal and Nicole Krämer. Overview and recent advances in partial least squares. In *Proceedings of the International Conference on Subspace, Latent Structure and Feature Selection*, 2006.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

Avneesh Saluja, Chris Dyer, and Shay B. Cohen. Latent-variable synchronous CFGs for hierarchical translation. In *Proceedings of EMNLP*, 2014a.

Avneesh Saluja, Kristina Toutanova, Chris Quirk, and Hany Hassan. Graph-based semi-supervised learning of translation models from monolingual data. In *Proceedings of ACL*, 2014b.

Blake Shaw. *Graph Embedding and Nonlinear Dimensionality Reduction*. PhD thesis, Columbia University, 2011.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. Language and translation model adaptation using comparable corpora. In *Proceedings of EMNLP*, 2008.

Nicolas Stroppa, Antal Van den Bosch, and Andy Way. Exploiting source similarity for SMT using context-informed features. In *Proceedings of the International Conference on Theoretical Issues in Machine Translation*, 2007.

Amarnag Subramanya, Slav Petrov, and Fernando Pereira. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of EMNLP*, 2010.

Liang Sun, Shuiwang Ji, and Jieping Ye. A least squares formulation for canonical correlation analysis. In *Proceedings of ICML*, 2008.

Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. Bilingual lexicon extraction from comparable corpora using label propagation. In *Proceedings of EMNLP-CoNLL*, 2012.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

Kristina Toutanova, Hisami Suzuki, and Achim Ruopp. Applying morphology generation models to machine translation. In *Proceedings of ACL*, 2008.

Zhaopeng Tu, Baotian Hu, Zhengdong Lu, and Hang Li. Context-dependent translation selection using convolutional neural network. In *Proceedings of ACL*, 2015.

Takeaki Uno and Mutsunori Yagiura. Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26, 2000.

Ashish Vaswani, Haitao Mi, Liang Huang, and David Chiang. Rule markov models for fast tree-to-string translation. In *Proceedings of ACL*, 2011.

Ashish Venugopal, Andreas Zollmann, Noah A. Smith, and Stephan Vogel. Preference grammars: softening syntactic constraints to improve statistical machine translation. In *Proceedings of NAACL*, 2009.

Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, 1997.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of ACL*, 1995.

Hao Zhang, Daniel Gildea, and David Chiang. Extracting synchronous grammar rules from word-level alignments in linear time. In *Proceedings of COLING*, 2008.

Jiajun Zhang and Chengqing Zong. Learning a phrase-based translation model from monolingual data with application to domain adaptation. In *Proceedings of ACL*, 2013.

Ying Zhang, Stephan Vogel, and Alex Waibel. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system. In *Proceedings of LREC*, 2004.

Kai Zhao, Hany Hassan, and Michael Auli. Learning translation models from monolingual continuous representations. In *Proceedings of NAACL*, 2015.

Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.

Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of ICML*, 2003.

Andreas Zollmann and Ashish Venugopal. Syntax augmented machine translation via chart parsing. In *Proceedings of WMT*, 2006.