Moving Beyond Phrase Pairs: The Relevance of the Corpus in a SMT World

Thesis Proposal

Aaron B. Phillips

March 22nd, 2010

Language Technologies Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania

Abstract

Machine translation has advanced considerably in recent years, but primarily due to the availability of larger data sets. Translation of low-frequency phrases and resource-poor languages is still a serious problem. In this work we explore a deeper integration of context, structure, and similarity within machine translation.

Instead of modeling phrase pairs in abstract, we propose modeling each instance of a translation in the corpus. Unlike the traditional SMT approach that builds a mixture of independent, simple distributions for each phrase pair, our model is a mixture of translation instances. The significance lies in that we use a distance measure to assesses the relevance of each translation instance. It permits simple the integration of instance-specific features which we plan to exploit in three key directions. First, we will introduce non-local features that identify the relevant context of an instance in order to favor those that are most similar to the input. Second, we will mark-up the corpus with metadata from multiple external sources to sharpen the scoring of each translation instance and guide the overall translation process. Third, we will identify related instances of translation and combine information across all forms. Each of these techniques explicitly takes advantage of the dynamic combination of models for each translation instance; similar extensions would be impossible or quite difficult in a traditional SMT system.

This shift in modeling moves translation beyond the concept of static, discrete translation entities and permits a more fluid assessment of each translation instance. We believe this approach will provide more robust estimation, a more consistent model, and a more flexible framework for integration of novel features. Furthermore, it is our expectation that this work will not only result in higher quality machine translation, but it will also lead to a better understanding of the connection between SMT and EBMT.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine translation has advanced considerably in recent years, but this has been primarily due to the availability of larger data sets. Translation of low-frequency phrases and resource-poor languages is still a serious problem. In this work we explore a deeper integration of context, structure, and similarity within machine translation. In particular, we seek to move beyond the notion of modeling translation as a combination of independent models. Instead, we treat each piece of the translation process as belonging to a broader web of information. The goal is that this approach allows for a more robust translation model that is truly data-driven.

General purpose, fully-automated, high-quality machine translation necessitates modeling the translation task as a decomposable process. Current state-of-the-art machine translation systems model a series of discrete, countable entities. Depending on the architecture, these entities take on different forms: phrase pairs, grammar rules, or factors. Yet, in all these approaches, one theme persists–namely, each entity is modeled independently using a log-linear combination of features.

This approach has some nice properties: it is easy to extend each entity with more features, and because of the independence assumption, the calculations are usually simple and scalable to large amounts of data. Furthermore, given enough data, the modeling space can be fragmented into very fine-grained, specialized entities. Indeed, as available bilingual datasets have grown, so too has performance of these systems. However, this situation also creates a delicate balancing act–with too little data, broader, more generic entities must be used or the model will over-fit. Selecting the 'right' entities is somewhat of a black art and is often dependent both on the amount of data and the language-pair.

What we propose is a shift in modeling: instead of modeling phrase pairs in abstract, we model each instance of a translation in the corpus. The accepted generative story in statistical machine translation describes a bilingual corpus as being formed by a probabilistic synchronous grammar or a sequence of phrase pairs with some distribution for re-ordering. There exists a 'true' distribution for each entity (phrase pair or grammar rule); the training corpus provides observations that one uses to estimate this distribution. This approach is compelling for many applications, but we are concerned that it is not sufficient to handle the complex nuances of natural language–and especially translation between two natural languages where rare events are the norm. Instead, we propose turning the traditional story upside down and argue that the corpus *is* the 'true' distribution, and that all future translations will be a synthetic combination of components of the training corpus. The goal is to select a translation that is *maximally consistent with instances of translation present in the corpus.* Instead of modeling the distribution of an entity in the corpus, we model the similarity between an entity and each translation instance.

The emphasis on modeling each instance of a translation bears resemblance to instance-based

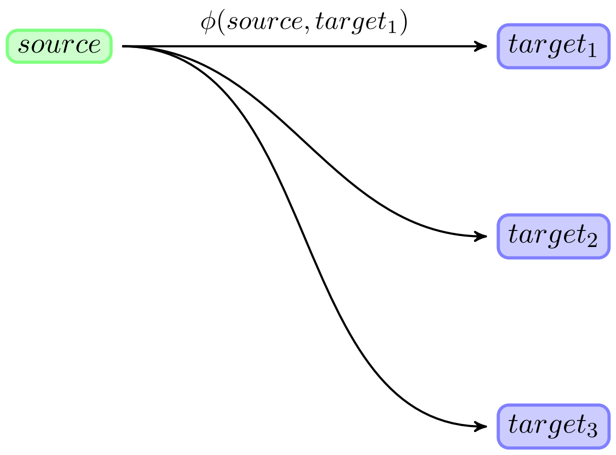


Figure 1.1: Standard translation model

or memory-based learning. Indeed, given a data point, a search is performed for the ‘nearest neighbors’ and a distance function determines the relative importance of each translation instance. A particular hypothesis is modeled by summing over the score of all translation instances (i.e. ‘neighbors’). The difference is that our distance measure is defined by a parametric, and in particular log-linear, model. This construction enables us to train the model and learn parameters for the distance function from the data. As a result, our model has both parametric and non-parametric characteristics.

Our aim is that this shift in modeling moves translation beyond the concept of static, discrete translation entities. Unlike the the traditional approach shown in Figure 1.1 that models a source and target entity *in abstract*, we propose the model shown in Figure 1.3 that introduces a modeling layer over each instance of the entity in the corpus. This middle layer represents actual instances present in the corpus which may not exactly match either the source or the target being modeled. Instead of one set of feature functions that operate only on the source and target, our approach applies three: $\phi_1(source, source')$, $\phi_2(source', target')$, and $\phi_3(target', target)$. The similarity metrics ϕ_1 and ϕ_3 measure the distance between what one desires to model and what information is available in the corpus. ϕ_1 calculates the distance between the $source$ phrase present in the input and the $source'$ of the instance while ϕ_3 measures the distance between the $target'$ of the instance and the $target$ hypothesis. Obviously, preference is given to instances of translation that are minimally divergent from the input and the hypothesis. But, when data is scarce, this framework allows one to capture a broader perspective of information from the corpus and yield more reliable feature estimates. Furthermore, this breakdown provides us the ability to model ϕ_2 in the context of where each instance of an entity occurs in the corpus. For example, one is permitted to generate features that use the document in which the instance is located or analyze the alignment probability of an instance with respect to the effect of the alignment on the remainder of the sentence. We represent each instance in the diagram as having multiple $target'$ phrases due to uncertainty in the alignment. This visual representation packs the alignment ambiguity and associates instances derived from the same location in the corpus. However, in practice our model considers each $(source', target')$ pair as a unique instance of translation.

The final log-linear model for an entity is composed by summing over the score of each translation instance. As indicated in Figure 1.3, *all* retrieved instances are used to model each target hypothesis.[1] The higher an instance's score, the more confident one can be that it represents a valid translation. Indeed, the score of each instance acts as an interpolation weight dictating how much it contributes to the final mixture. Due to ϕ_3, an instance's score is relative to the target hypothesis under consideration and will influence some target hypotheses more than others. We acknowledge that by discarding the reduction into phrase pairs or grammar rules, the model is considerably more complex. While at one point this approach may not have been practical, today it is, thanks to Moore's law.

An integral component of this proposal is that we will maintain the entire corpus at run-time. This allows our approach to *not* be limited to modeling one type of entity. Our current implementation as described in §3 is most similar to phrase-based SMT. However, in addition to modeling standard phrase pairs, we support modeling entities that contain gaps, additional factor-like sequences, and higher-level annotations. The training corpus is finite; thus, each entity that is extracted from the corpus will also be finite. However, by defining a broad range of entities that can be modeled, the proposed approach permits generalization and simulates continuous space modeling. Furthermore, the decision of what entities to model can be delayed until run-time. When sufficient information is not present for a particular type of entity, then the system will dynamically query more general entities. Because our model performs calculations on one instance at a time, we are only required to retrieve a small sample of relevant translation instances at run-time. This is unlike a traditional SMT system where the model requires statistics over the entire corpus to obtain relative frequency counts. Hence, the design of our model permits us to delay until run-time the computation of the model and the decision of what entities to model.

Ultimately, this approach provides more robust estimation, a more consistent model, and a more flexible framework for integration of novel features. To substantiate this claim, we propose injecting novel sources of information that exploit instance-specific, run-time features. Therefore, our thesis is that

> **Thesis Statement**: Modeling each instance of a translation in the corpus will improve machine translation quality and facilitate the integration of non-local features, metadata, and the use of related translations.

We have already implemented this approach to modeling through the development of the Cunei Machine Translation Platform that will be described in §3. Our thesis work presented in §4 extends this platform in three primary directions. First, we introduce non-local features that identify the relevant context of an instance in order to favor those that are most similar to the input (§4.1.1). Second, we provide a mechanism to mark-up the corpus with metadata from multiple external sources (e.g. statistical word clustering, part-of-speech tagger, dependency parser, or syntactic tree parser) to provide additional information about each instance of a translation and guide the overall translation process (§4.1.2). Third, we identify related translations and combine information across all forms (§4.1.3). Each of these techniques explicitly takes advantage of the fact that the outlined approach models each instance of translation individually and combines the information dynamically at run-time; similar extensions would be impossible or quite difficult in a traditional SMT system.

In the next section we will walk through several different ways that traditional SMT models have been extended to cope with data sparsity and learn from multiple forms of information. Some of

[1]While it is conceptually simpler to describe the collection of target hypotheses separate from the collection of targets present over all instances, these are actually one and the same. The model cannot create a hypothesis that does not occur in the corpus. Therefore, the set of possible target hypotheses is determined by the targets retrieved for each instance from the corpus. The calculation for $\phi_3(target', target)$ operates over a cross-product of all possible (unique) targets. Issues of computational feasibility will be discussed later, but this quite tractable due to sampling.

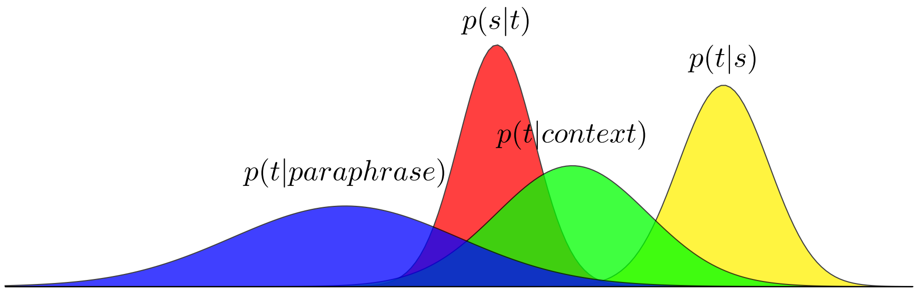


Figure 1.2: An SMT system as a mixture model encompassing multiple forms of knowledge

this will sound similar to what we are proposing above. However, the model we use and, therefore, the way in which additional information is incorporated into it, is fundamentally different. The SMT approach has been to construct multiple independent translation models where each takes an additional piece of information into account. The resulting log-linear translation model is essentially a mixture model of these independent distributions as graphically summarized in Figure 1.2.

Our approach is different in that it leverages ideas from instance-based learning to score each instance of a translation. The additional information we propose introducing into the model will be used to better estimate the relevance of each translation instance. Albeit not in the same manner, the same information could be used within a traditional SMT system if one were to carefully craft appropriate distributions to include in the mixture model. Even so, in a SMT system all the distributions must be discrete, decided on *apriori*, and generally they are expected to be independent. When we introduce non-local information or metadata, we are not forced to bin like translations and calculate relative frequency estimates. Instead, our approach calculates a smooth estimation of an instance's relevance using a distance measure. The distance measure, which is part of what we will develop throughout the thesis, uses multiple features to asses similarities in context and metadata particular to one instance of translation. Furthermore, instead of multiple independent models, the use of a distance measure incorporates joint dependencies. Instead of building a mixture of independent, simple distributions, our model is a mixture of translation instances. It is our expectation that this work will not only result in higher quality machine translation, but it will also lead to a better understanding of the connection between SMT and EBMT.

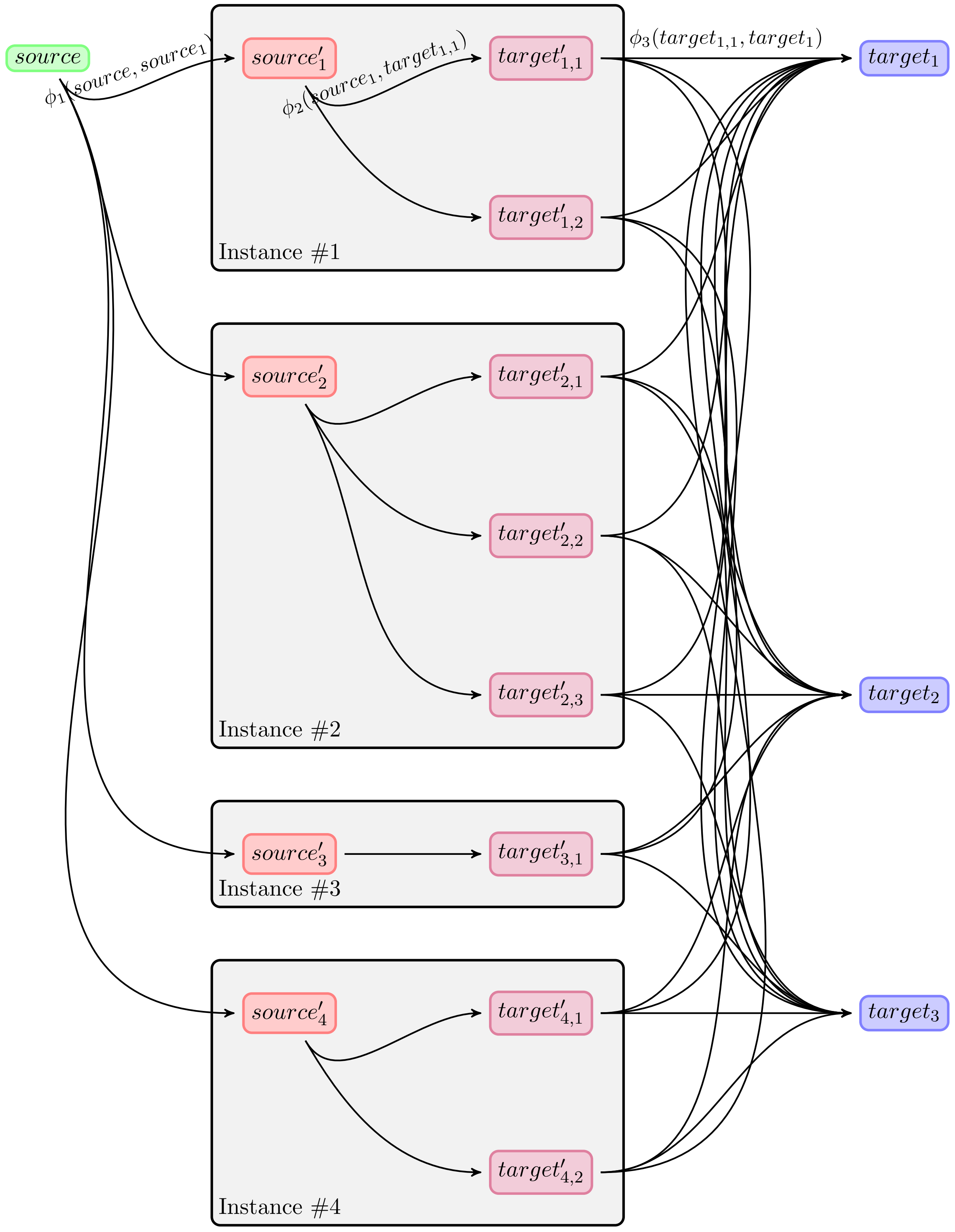


Figure 1.3: Proposed translation model

Chapter 2

Prior and Related Work

2.1 Paradigms of Machine Translation

In the next chapter we will introduce Cunei, a new platform for machine translation. However, before plunging into the technical details, it is important to understand the broader context of machine translation that has shaped this research. To begin, Cunei is a hybrid between the traditional statistical (SMT) and example-based (EBMT) models of translation. It extends upon the modeling pioneered in SMT and maintains a log-linear model that allows for efficient minimum error rate training. However, like EBMT, our approach indexes the entire corpus, allowing for the run-time extraction of novel entities and the ability to model each instance in context.

2.1.1 Statistical Machine Translation

The advent of Statistical MT defined a probabilistic approach to machine translation that was quickly adopted by many researchers. In both early word-based incarnations and more modern phrase-based systems, SMT has sought to formulate well-defined models that use a limited quantity of information to represent the translation process. These simple models are a key asset as they can be easily trained to minimize an objective function.

SMT originally leveraged the source-channel paradigm using Bayes' rule and a well-structured probabilistic model. However, the log-linear model, introduced to SMT by Berger et al. (1996) and popularized by Och and Ney (2002), threw away the *a priori* structural dependencies of the model and has since become the prevailing modeling approach. Log-linear models not only incorporate all the information of their Bayesian counterparts, but they can be easily extended with more features. Crucially, the system builder does not need to understand how all the features interact. Training the model on held-out data automatically determines the relative importance of each feature and accounts for statistical dependencies. Furthermore, minimum error rate training of log-linear models is straight-forward, as discussed in §2.4.

Due both in part to its origins in the source-channel paradigm and the desire to build simple models, SMT has adopted a static, top-down approach to modeling the translation process. Each combination of a source phrase and a target phrase, commonly referred to as a phrase pair, is represented by one log-linear model that represents the probability that the target phrase is a translation of the source phrase. The feature functions for each phrase pair are calculated over the aggregate of all instances of the phrase pair in the training corpus. Generally, they are relative frequency counts conditioned on some attribute of the phrase pair where each occurrence of the phrase pair is considered equal. Once the model is constructed, the training corpus can be "thrown away" as it provides no further information.

In its default configuration, the popular open-source software package Moses (Koehn et al., 2007) creates a log-linear model with eight features. Each phrase pair consisting of a source phrase s and a target phrase t is assigned five features: two lexical scores that estimate the phrase probability using the word alignment a and the lexical weights w ($lex(s|t) = \max_a \prod_{i=1}^{n} \frac{1}{|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(s_i|t_j)$ and $lex(t|s) = \max_a \prod_{i=1}^{n} \frac{1}{|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(t_i|s_j)$), two relative frequencies based on the phrase alignment ($p(s|t)$ and $p(t|s)$), and a constant phrase penalty to prefer the use of longer phrases. The last three features are applied during decoding: a language model score, a distortion penalty, and a word penalty. A more detailed description of these features is present in Koehn et al. (2003).

Nearly all extant SMT systems have these eight features at their core because their log-linear combination has been found to perform quite well. To be clear, this is not an exhaustive list of possible features and the incorporation of new features is a common research direction. For example, in their submission to the 2005 IWSLT workshop, Koehn et al. (2005) describe additional lexical reordering features. The point is that by understanding the eight features present in Moses, the reader should be able to grasp the foundation of most SMT systems.

SMT's simplicity and efficient optimization has enabled it to best the competition in competitive evaluations that use automated metrics. This, in turn, has led to its current position of dominance within the research community.

2.1.2 Example-Based Machine Translation

Example-Based MT proposes translations that closely match previous training examples. Many methods have been employed, but the unifying theme is that the system searches the corpus and extracts examples of translations that have a high degree of similarity to the input. Thus, the simplest representation or model of the translation process *is the training corpus*. Examples of translations may be retrieved based on a deeper structure such as the syntax or semantics of a phrase (Nakazawa et al., 2006; Way, 2003). Alternatively, a shallow approach as described in Brown (1996), which only retrieves exact lexical matches, may be employed. In this case, all retrieved examples are lexically equivalent to the input, but still vary according to their surrounding context, alignment probability, genre, location in the corpus, etc. This example-specific knowledge is a crucial component of all EBMT systems, and it is the mechanism by which they select the most appropriate translation.

Scoring is performed on an example-by-example basis, but there is no clear modeling approach favored by EBMT. The score for a translation is often modeled by selecting the maximal score from the set of examples or summing over the score of each example that is consistent with the translation. Selecting the most likely example discards highly valuable frequency information while the summation makes optimization inefficient. Lacking better alternatives, the example-specific features of an EBMT system are typically tuned by hill-climbing each parameter independently, e.g. as described in Brown et al. (2003).

To be clear, the modeling approach we propose has precedent in EBMT as it leverages the key idea that each instance of a translation is modeled separately. Our contribution is novel in that it embeds the modeling of each instance within a larger statistical framework. Furthermore, our model significantly outperforms all known EBMT systems in that we are able to demonstrate performance competitive with SMT.

2.1.3 Run-Time Machine Translation

The nature of Example-Based MT necessitates the extraction of examples at run-time. It is perhaps no surprise then that there exists a long history of research in EBMT on how to perform phrase alignment and scoring efficiently at run-time. However, a handful of SMT systems have also been developed that extract phrase pairs or grammar rules at run-time in order to reduce the memory-footprint of large phrase-tables.

Vogel (2005) and Callison-Burch et al. (2005) describe remarkably similar, but independently developed, SMT systems that model phrase pairs on the fly *only* for the phrases contained in the input. When building a SMT model offline, one normally faces a trade-off between the quality and the size of the model. Callison-Burch et al. (2005) estimated that storing a phrase-table containing the word alignments and features for all phrase pairs up to length 10 in the NIST Arabic-English dataset would consume approximately 38GB! Thus, even though it is well known that longer phrase pairs improve translation quality, it is common practice to only model short phrase pairs. The work of Vogel (2005) and Callison-Burch et al. (2005) addresses this problem by using a suffix array of the training data as a compact and efficient alternative to the standard phrase-table. Both approaches describe how to retrieve *any* phrase pair present in the corpus–regardless of length–and calculate the standard SMT features for it.[1] This is made possible by the fact that the standard SMT model is very simple and that the nature of suffix arrays makes counting the occurrence of a particular word or phrase in the corpus quite inexpensive.

As the years have progressed, the data problem has only grown worse. Lopez (2008) is a more recent treatment of the same problem with state-of-the-art results. The author uses a suffix array to index a very large data set and perform Heiro-style statistical decoding with rules generated on-the-fly. This avoids the bottleneck of modeling *all* possible rules by extracting only the rules needed to decode the input at run-time. This work does attempt to take advantage of the lazy calculation by introducing a new 'dynamic' feature. That feature, coherence, is the number of times that a phrase was successfully aligned (to any target) over the number of times the source phrase was sampled. However, aside from this addition, the feature set is limited to the same Moses-style features (albeit calculated at run-time) that have been used for several years. Indeed, the author notes that using discriminative, contextual, feature-rich models would be preferred, but states it is an open problem.

Our approach, as implemented in Cunei, is also a run-time system. It builds upon the work outlined above and uses many of the same techniques (such as suffix arrays). The distinction, which will be made clear later, is that this platform has been explicitly designed to permit retrieval of related instances of translation and score them individually with contextual, feature-rich models.

2.2 Phrase Alignment

The IBM word-alignment models (Brown et al., 1993) describe a reasonably intuitive and clean generative story. Even many years after their introduction, they continue to perform well and are widely-used. However, moving from word alignments to phrase alignments has proven to be a complex endeavor. The number of possible phrases contained within a sentence is exponential with respect to the length of the sentence and even in large corpora, the majority of phrases occur rarely. The widely-adopted solution follows the work of Koehn et al. (2003) to identify phrases and their alignments based on a heuristic function of the word alignments. The Moses system provides multiple heuristics for this task with the default method known as 'grow-diag-final'. One fairly serious downside to this approach is that instead of modeling a probability distribution, a sentence with a fully-specified word alignment will have one deterministic set of phrases and alignments. Only at the edges of phrases where word alignments are not present is it possible to extract multiple,

[1]The phrase probability can only be calculated in one direction, but this does not seem to greatly affect the overall performance.

inconsistent phrase pairs. Furthermore, lacking a probability distribution, all extracted phrase pairs are considered equal when generating relative frequency counts for the translation model.

Like the heuristic function, Vogel (2005) builds on top of the word alignment. However, this work models the phrase alignment by defining probability models over the word alignment. To ensure that the model does not greedily select an alignment that forces the remainder of the sentence to align poorly, it not only models the alignment within the phrase, but also models the alignment probability *outside* the phrase boundaries. Furthermore, word alignments from both directions are incorporated, resulting in a symmetric alignment model. All of the probability distributions are combined as features in a log-linear model. By calculating the partition function, one could obtain a valid probability distribution, but this is not necessary for selecting the most likely alignment(s). One advantage of this approach is that an alignment cannot strictly fail, but only return an unlikely candidate. This is helpful for low frequency items where one prefers to predict *something* even if it has a high degree of uncertainty. Perhaps most interesting is that all of these calculations depend only on the sentence in which the phrase occurs and therefore may be performed on-line or off-line.

To be clear, the focus of this work is not on alignment per-se, but alignment is a necessary step within any data-driven MT system. Alternative methods that directly model phrase alignments do exist (Blunsom et al., 2009; Cherry and Lin, 2007; Marcu and Wong, 2002). These approaches, while technically superior, are relatively new, expensive to compute, and not (at least currently) demonstrably better as data increases. As such, we adopt Vogel (2005)'s approach which is conceptually in agreement with modeling each instance. Furthermore, this parametric model allows for arbitrary sources of knowledge to be encoded in the alignment process and for the alignment features to be included with all other translation features at run-time. It integrates cleanly with our system and enables Cunei to jointly learn the alignment weights and the translation weights, yielding greater consistency in modeling.

2.3 Translation Modeling

An integral component of this thesis is *how* translations are modeled. Each subsection below surveys a different way in which researchers have extended the traditional translation model. We will expand on these ideas and generate similar features, but we will apply them within our modeling approach that assesses each translation instance. In doing so we will move beyond using several independent models to one unified model that ties these threads of research together.

2.3.1 Contextual Translations

The MT community generally agrees that context *should* improve translation quality, but how to model it and how to efficiently integrate it within the translation model are open questions. In addition, some local context is already captured by modeling phrases that consist of multiple words. The earliest works using a richer context model focused on resolving the ambiguity of known problematic words. Recognizing that this is essentially the same problem as Word Sense Disambiguation (WSD), Carpuat and Wu (2005) tried to pair a WSD system with a SMT system. However, their attempts at using the WSD system to filter possible translations or selectively replace words in the machine translation output failed.

The first clearly successful results were reported by Chan et al. (2007) using the Hiero system. After applying each grammar rule during decoding, their system updated context features for the current span. These context features scored the sequence of terminal symbols and were derived from an external WSD system. Furthermore, the context features acted like every other feature in the translation model and their weights could be learned with MERT. Their baseline Hiero system trained on the Chinese-English FBIS corpus scored 29.73 BLEU when evaluated on NIST MT03. The same system with the additional context features increased to 30.30 BLEU.

Following up on their initial negative results, Carpuat and Wu (2007) similarly demonstrated a successful and tight integration of context within a phrase-based translation model. Like Chan et al. (2007), they now modeled context as features within the log-linear translation model. Instead of using Hiero and incorporating these dynamically during decoding, Carpuat and Wu (2007) used a standard phrase-based SMT system and appended the context features to the phrase table. Unlike previous work, they did not actually build or use an external WSD system. Instead, they took the features that are commonly used in a WSD system and plugged them directly into the SMT model. The individual features they report using are bag-of-words context, local collocations, position-sensitive local POS, and basic dependency features.

Gimpel and Smith (2008) continue this direction of research but discard the notion that WSD defines the most appropriate context features for machine translation. Instead, they use a smorgasbord of features that operate over the entire input. Some of these features, such as using colocations of lexical or POS sequences, are similar to prior work. However, they add to this mixture several new syntactic and positional features. For example, they set features based on if the phrase is (exactly) a constituent, the nonterminal label of the lowest node in the parse tree that covers the phrase, if it occurs at the beginning of the sentence, or the fraction of the words in the input that are covered by the phrase. The result is minor gains in Chinese-English and German-English in-domain experiments, but large improvement (0.3615 to 0.3772 BLEU) from using the context features in a system trained on Chinese-English corpora that included out-of-domain United Nations data.

2.3.2 Adaptation

Instead of modeling specific context features in the translation model, a different approach is to alter the data from which the translation model is built. This technique, more commonly known as adaptation, is effective at globally skewing the translation model toward a particular target. Data-driven statistical models are merely a reflection of the text they are trained on. Filtering or re-weighting the training data will alter the model's predictions.

Hildebrand et al. (2005) is an early approach that applies information retrieval (IR) techniques to filter a training corpus such that it is maximally similar to the text to be translated. For every input sentence to be translated, the n-most-similar sentences are extracted from the corpus. These n-most-similar sentences for all input sentences are combined together (potentially including duplicates) to form a new training corpus. A standard translation model can then be built from this filtered training corpus. In order to calculate sentence similarity, the authors measure the cosine distance between TF-IDF term vectors–a common algorithm for document similarity in IR. They evaluated this approach on 506 lines of Chinese-English tourism dialogue. 20,000 lines of tourism dialogue and 9.1 million lines of newswire and speeches were available for training. The performance of the system was sensitive to the size of n in the selection of the n-most-similar sentences, but generally they were able to demonstrate improvement in NIST scores.

More recently, the work of Matsoukas et al. (2009) assigned weights to each sentence pair in a corpus to minimize an objective function during training. The weight for each sentence was calculated with a perceptron model that utilized several simple feature functions. The perceptron was trained by minimizing the expected Translation Error Rate (TER) over an n-best list for a tuning set. As the sentence weights are updated, the phrase and lexical translation probabilities also need to be re-estimated. To account for this divergence, the n-best list is regenerated after every 30 iterations. The researchers found that when the tuning set was similar to the test set, their approach led to significant gains in TER, BLEU, and METEOR. Furthermore, the baseline

Arabic-to-English system that they improve upon is one of the best in the world.

Both of these works partially capture what we are trying to achieve in our model. Instead of weighting instances of translation in the corpus, this line of research weights or filters sentences in the corpus. An important distinction is that even with Matsoukas et al. (2009)'s more integrated approach, there are still two separate models with one being built upon the other. This is a result of preserving the traditional SMT mixture of distributions. By modeling translation instances we avoid this situation and construct one unified model.

2.3.3 Source-Side Paraphrases

Building upon research in paraphrasing, researchers have experimented with augmenting standard translation models with phrase pairs from related phrases. Paraphrases can be identified using the pivot method introduced by Bannard and Callison-Burch (2005) which reasons over the phrase alignments in a bilingual corpus. Given a source phrase, the algorithm 'pivots' through the possible target translations and then adds to the set of paraphrases the alternative source translations found for each of the aligned targets. The probability of source phrases s_1 and s_2 being a paraphrase is calculated by marginalizing over all possible target phrases t with which they both align: $p(s_2|s_1) = \sum_t p(s_2|t)p(t|s_1)$. Callison-Burch et al. (2006) apply this concept to machine translation by augmenting a phrase table with source-side paraphrases. In addition, the probability of the paraphrase, $p(s_2|s_1)$ for paraphrases or 1 for original entries, was added as a new feature. The primary findings of this work was that paraphrases could be used to significantly improve translation coverage. However, it also resulted in a BLEU improvement in experiments using up to 320,000 sentence pairs from the Spanish-English and French-English Europarl corpora.

Marton et al. (2009) describe another approach using paraphrases to propose phrase pairs for unknown words in SMT. However, in this case the paraphrases are collected from a large monolingual corpus using 'distributional profiles'–essentially monolingual clustering based on the context of each phrase. Given an input document to translate, the authors augment the phrase-table with new phrase pairs from paraphrases *only* for unknown words in the input. Every phrase pair includes a similarity score between the input and the source of the original phrase pair. For phrase pairs original to the model and not generated via paraphrasing, the log-score will be zero. The authors artificially limit their approach with this restriction and unfortunately do not report any results when applied to the entire phrase-table. However, they do show marginal improvement with this limitation on a small Spanish-English dataset. Using monolingually-derived similarity metrics is likely not as accurate as using pivot-based methods, but the authors suggest that this difference is offset in practice by the availability of *much* more monolingual data.

2.3.4 Target-Side Similarity

In the methods above, translation matching on the source-side was extended using knowledge from similar phrases. Context-Based MT (Carbonell et al., 2004) focuses on the other end of the translation pipeline–calculating distributional profiles and similarity among the target phrases. A high-precision lexicon is used to translate content words and get a rough idea of what target words should be present. Long n-grams are retrieved that match this collection of target words as closely as possible. Preferably, each target n-gram will not introduce any novel content words, but if it does, the word must occur in the monolingual corpus in the same contexts as the word in which it is replacing–i.e. it is believed to be a paraphrase. Decoding a translation is thus the selection of a sequence of overlapping n-grams that diverge minimally from the target words predicted by the lexicon.

An alternative approach that does not make use of an additional monolingual resource is to bias decoding to favor translations that are minimally divergent. Minimum Bayes-Risk Decoding as introduced by Kumar and Byrne (2004) takes into account the loss associated with choosing one translation given all the other translations in the n-best list. The loss metric measures how similar the predicted translation is to the alternatives. Indeed, MBR can be viewed as a specialized case of using related translations during decoding as we propose in §4.1.3. The crucial difference is that our approach allows for arbitrary features that measure similarity across both the source and the target, and it incorporates these features directly as part of the parametric model such that their weights are learned during optimization.

2.3.5 Continuous Space Modeling

Modeling the source-side or target-side similarity of translations generalizes the model and results in a smoother distribution. An alternative approach is to move from representing the translation model in a discrete space to a continuous space.

Schwenk et al. (2006) take the first step toward continuous space modeling in machine translation by adding a continuous space language model. In particular, the language model was built using a neural network that maps histories into a continuous space. The source and target words are still discrete, but the set of histories that a target word follows are modeled in continuous space. Rather than backing off to shorter contexts when an n-gram probability is unknown, the neural network posterior probabilities are interpolated across all contexts of the same length. In experimentation, the authors showed this led to a 0.8 BLEU improvement in translation over a 4-gram Kneser-Ney back-off language model. For comparison, this result is approximately the same magnitude of improvement they report when using a 4-gram instead of a 3-gram Kneser-Ney back-off language model.

More recently, Sarikaya et al. (2008) applied continuous modeling to the translation process with a parametric model employing Gaussian mixtures to represent phrase pairs. Their approach is very similar to acoustic modeling and they actually retrofit a speech recognizer to perform translation. First, they use Latent Semantic Analysis (LSA) to generate a mapping of related words. Each word is then mapped into a continuous space vector representing its relation to all other words in the vocabulary. Source language word vectors are concatenated with target language word vectors to form word-pair vectors. These word-pair vectors are modeled using a mixture of Gaussians. In order to reduce the number of parameters, all word-pairs share the same set of Gaussian densities but have different mixture weights (which are learned during training).

Sarikaya et al. (2008) criticize current statistical translation models (i.e. phrase tables), as they are known to suffer from 1) overtraining 2) lack of generalization 3) lack of adaptation and 4) lack of discrimination. Not surprisingly, these are largely the same problems that this thesis intends to address. A continuous space model is initially compelling, but in order to reason in a continuous space, one has to perform a mapping from the discrete words or phrases into a vector of elements. Performing this transformation with LSA results in a mapping that is not updatable or learned as part of the model, but instead committed to *a priori*.

Our model is not continuous, but it does exhibit some continuous-like attributes. In particular, our distance function allows us to glean information in the modeling process from instances that do not exactly match the input or output. By scoring the relevance of each instance we do not have to discretely bin them, but rather each instance can contribute to multiple hypotheses.

2.3.6 Massive Features Spaces

A lure of the log-linear model is that it allows for numerous arbitrary features. While theoretically *any* feature can be used, in practice not all features are equal and careful feature engineering can significantly improve a system. One suggestion has been that if we make the feature space *much* larger, then optimization can sort out the good features from the bad features. Recently Chiang et al. (2009) did just that and extended a Heiro-based and a syntax-based MT system to include over 10,000 features. Most of the features were linguistically informed to catch specific situations where grammar rules failed. In addition, the authors incorporated features modeling source-side context and discount weights for low-frequency entries. In sum, their approach to translation remained the same, but they added a vast amount of new discriminative knowledge to the system. They report success in learning appropriate weights for each feature using MIRA and showed respectable improvement in translation quality on a held-out test set.

The idea of using a large feature space, while frequently a component of discriminative modeling, is itself orthogonal to the modeling approach. We also plan to significantly expand our feature space in §4, but the focus of our work will be on the type of features and not specifically the quantity of them. For example, the majority of features present in Chiang et al. (2009) are contextual features that trigger when a particular source word is present before or after a particular word-alignment. While this same feature could be programmed into our system, we would prefer to incorporate distance measures that are not so rigidly lexicalized and can be used in multiple situations. We believe such discriminative features may not be necessary in our approach because we are already discriminating between instances. Furthermore, because Chiang et al. (2009) model an abstract grammar rule, the aforementioned context features rely on the most frequent word alignment. In contrast, our model could score each instance of a grammar rule application in the corpus and permit varying word alignments across the instances.

2.4 Optimization

The flexibility of adding numerous features to a log-linear model comes at the cost of learning appropriate weights for each feature in the parametric model. One relatively simple technique that was favored in early systems was to maximize entropy of the translation probability distribution subject to empirical constraints (Berger et al., 1996). This optimization function is convex with a unique global optimum that fits the expectation of the data. However, translations that are selected by this learning process are not necessarily the same translations that will result in the optimal score according to a particular evaluation metric.

As a result, Och (2003) introduced minimum error rate training (MERT) which minimizes error on a particular dataset *according to a specific evaluation measure*. While translation systems can produce an n-best list of scored translation hypotheses, evaluation metrics only evaluate the single best translation hypothesis. MERT uses an efficient linear programming solution to select weights that favor this single optimal hypothesis; it is not concerned with the rank or score of any other hypothesis. Unfortunately, there is no longer the guarantee of finding a global optimum as most evaluation metrics are not convex. However, in practice MERT works quite well as long as the number of parameters is relatively few.

Watanabe et al. (2007) address the issue of scaling MERT to a larger number of features by adapting the Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006) for use in machine translation. Like MERT, this approach uses a loss measure which is typically the inverse log of BLEU. However, instead of minimizing loss, this approach tries to find a set of weights such that the difference in the score (i.e. margin) between a correct and incorrect translation is at least as large as the loss of the incorrect translation. Furthermore, instead of using the 1-best translation, Watanabe et al. (2007) perform the calculation on a handful of the top translations. (The algorithm could use the entire n-best list, but the author reports that this would be significantly more expensive to compute.) The margin creates an upper bound for the score of each of the selected 'best translations' and forces them to loosely fit the objective function. The key advantage to this approach is that it is conservative and tends to be more robust to using a large number of features.

Alternatively, Smith and Eisner (2006) propose an approach that minimizes the *expected* loss of a translation. The expected loss is calculated by taking the evaluation metric's error rate for a translation and multiplying it by the likelihood of selecting that particular translation (i.e. the score for the translation divided by the sum of all translation scores). It is not possible to generate all possible translations, but the space can be approximated by the translations present in the n-best list. This approach is conceptually very similar to MERT with the distinction that this algorithm operates over the entire n-best list. In order to avoid local optima, Smith and Eisner (2006) apply an annealing process to sharpen the expected scores of the n-best list over time. In the early stages all translations are considered equal, but by the end of the optimization process nearly all the weight for the expected score is concentrated in the top translation.

While Watanabe et al. (2007) and Smith and Eisner (2006) are clearly better alternatives than Och (2003), there have been no comparisons between the former two. As will be presented in §3.3.5, we select the approach of Smith and Eisner (2006) in our system, but overall the thesis is neutral toward the objective function and the optimization algorithm used to learn the model's weights.

Chapter 3

Cunei Machine Translation Platform

The research we propose necessitates a new platform for machine translation. In particular, the approach differs significantly enough from the traditional SMT model that we could not simply extend an existing system. An EBMT system would have been more amenable to extension, but none existed that were on par with state-of-the-art SMT systems. Much of the preparatory work for this thesis proposal has gone into building Cunei, a new platform for machine translation, that *is* state-of-the-art and *will* support this line of research.

3.1 The Problem

A static, log-linear model encoded within a phrase-table provides a convenient format for representing how translations are modeled. It is easy to understand and conceptually simple to extend or modify. Unfortunately, in practice for new features to have impact they need to be carefully crafted. This task of feature engineering is complex, messy, and often language-dependent. Even worse, otherwise good research can go to waste if the features are not carefully engineered.

Consider, for example, the task of translating newswire from French to English given some parallel text in two domains: newswire and parliamentary proceedings. The provided newswire training data is in-domain and likely to contain relevant translations. The parliamentary proceedings, on the other hand, sometimes use archaic language that is appropriate in the context of parliamentary discourse, but ill-suited for general-purpose translation. Actual translations for the French word 'interrompue' from European parliamentary proceedings are provided in Table 3.1. The first sentence

French	je vous ai **interrompue** , excusez-moi .	(1)
English	i **interrupted** you , please forgive me .	
French	l' heure des votes est **interrompue** .	(2)
English	voting time is **suspended** .	
French	je déclare **interrompue** la session du parlement européen .	(3)
English	i declare the session of the european parliament **adjourned** .	

Table 3.1: Example out-of-domain translations from the European Parliament

provides a reasonable general-purpose translation. However, translations like those present in the second and third sentences occur much more frequently in this particular corpus. The in-domain newswire text will have the reverse distribution with ‘interrompue’ most frequently translating as ‘interrupted’ and occasionally translating as something else.

Unfortunately, we are provided copious out-of-domain parliamentary proceedings and only a small quantity of in-domain newswire text. How do we combine knowledge from these two different sources of knowledge? If we treat all the text equally, then the out-of-domain translations from the parliamentary proceedings will overwhelm all else. Ideally, we want to build a translation model that is more informed by translations present in the newswire text while still being able to back off to translations present in the much larger parliamentary proceedings.

Conceptually, the obvious solution is to add a new feature that describes whether the translation is from the in-domain or out-of-domain training data. But pragmatically defining the calculation of this feature is not as obvious. A simple binary feature that specifies whether or not this translation is present in the in-domain data is too coarse as the feature would still be set if all but one of the instances resides in the out-of-domain text. Alternatively, one could calculate the probability that given an instance of the translation pair, it will occur in the in-domain text. This is more reasonable, but it would still necessitate further refinement in order to reflect the fact that the out-of-domain data is considerably larger.

Instead of creating one new feature to model in-domain vs. out-of-domain, a more comprehensive approach would be to duplicate all translation features and condition the new set of features on the in-domain data. The original set of features would not change; they would still be estimated over all available data and provide back-off estimates. The new set of features would use the same calculation, but only collect counts over the in-domain data. By having both coarse and fine-grained features, model weights can be learned that appropriately emphasize some dependencies and safely ignore others.

After building the domain-sensitive model, we then realize that some of the sentences in the training data have high-quality alignments whereas others include many unknown alignment links. We can marginalize over the alignment in estimating each feature, but this does not provide a new feature through which the model can adjust how much it believes the alignment. Instead, we want to condition the translation model on the quality of the sentence alignment in the same way we managed domain sensitivity. Following the method above, for every binary dependency we add, the feature-space will double. This is probably acceptable for a handful of dependencies, but it quickly became untenable as the feature-space grows exponentially. Not only does this generate a high-dimensional space that can cause problems with optimization, but the sufficient statistics become more and more complex to calculate. Furthermore, using a binary dependency of ‘good alignment’ vs ‘bad alignment’ is not very convenient when what we actually have is an alignment score. We could segment the alignment score into more bins, but fundamentally we are forced to condition on discrete events.

Nonetheless, our story is not complete and the problem is about to get worse. One of the more interesting areas of current research in machine translation is modeling non-local features. By definition, these are features that occur *outside* the phrase pair. Perhaps the most prominent example is source-side context. Continuing with the same example, we would like to learn that ‘interrompue’ is translated as ‘adjourned’ when the surrounding context contains the word ‘session’. Setting aside the fact that we would need to create yet another set of conditional features, storage of this information within a phrase table becomes problematic. Instead of one entry in the phrase table for each phrase pair, we will need a new entry for each context in which we want to model the phrase pair. This dramatically increases the size of the phrase table. Furthermore, it limits each possible context to a discrete event and the set of contexts used by the model must be decided on

Original Model		$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
Domain-Sensitive Model		$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
		$P(s\|t,d)$	$P(t\|s,d)$	$lex(s\|t,d)$	$lex(t\|s,d)$
Alignment and Domain-Sensitive Model		$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
		$P(s\|t,d)$	$P(t\|s,d)$	$lex(s\|t,d)$	$lex(t\|s,d)$
		$P(s\|t,a)$	$P(t\|s,a)$	$lex(s\|t,a)$	$lex(t\|s,a)$
		$P(s\|t,d,a)$	$P(t\|s,d,a)$	$lex(s\|t,d,a)$	$lex(t\|s,d,a)$
Alignment, Domain, and Context-Sensitive Model	c_1	$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
		$P(s\|t,d)$	$P(t\|s,d)$	$lex(s\|t,d)$	$lex(t\|s,d)$
		$P(s\|t,a)$	$P(t\|s,a)$	$lex(s\|t,a)$	$lex(t\|s,a)$
		$P(s\|t,d,a)$	$P(t\|s,d,a)$	$lex(s\|t,d,a)$	$lex(t\|s,d,a)$
		$P(s\|t,c_1)$	$P(t\|s,c_1)$	$lex(s\|t,c_1)$	$lex(t\|s,c_1)$
		$P(s\|t,d,c_1)$	$P(t\|s,d,c_1)$	$lex(s\|t,d,c_1)$	$lex(t\|s,d,c_1)$
		$P(s\|t,a,c_1)$	$P(t\|s,a,c_1)$	$lex(s\|t,a,c_1)$	$lex(t\|s,a,c_1)$
		$P(s\|t,d,a,c_1)$	$P(t\|s,d,a,c_1)$	$lex(s\|t,d,a,c_1)$	$lex(t\|s,d,a,c_1)$
	c_2	$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
		$P(s\|t,d)$	$P(t\|s,d)$	$lex(s\|t,d)$	$lex(t\|s,d)$
		$P(s\|t,a)$	$P(t\|s,a)$	$lex(s\|t,a)$	$lex(t\|s,a)$
		$P(s\|t,d,a)$	$P(t\|s,d,a)$	$lex(s\|t,d,a)$	$lex(t\|s,d,a)$
		$P(s\|t,c_2)$	$P(t\|s,c_2)$	$lex(s\|t,c_2)$	$lex(t\|s,c_2)$
		$P(s\|t,d,c_2)$	$P(t\|s,d,c_2)$	$lex(s\|t,d,c_2)$	$lex(t\|s,d,c_2)$
		$P(s\|t,a,c_2)$	$P(t\|s,a,c_2)$	$lex(s\|t,a,c_2)$	$lex(t\|s,a,c_2)$
		$P(s\|t,d,a,c_2)$	$P(t\|s,d,a,c_2)$	$lex(s\|t,d,a,c_2)$	$lex(t\|s,d,a,c_2)$
	c_3	$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
		$P(s\|t,d)$	$P(t\|s,d)$	$lex(s\|t,d)$	$lex(t\|s,d)$
		$P(s\|t,a)$	$P(t\|s,a)$	$lex(s\|t,a)$	$lex(t\|s,a)$
		$P(s\|t,d,a)$	$P(t\|s,d,a)$	$lex(s\|t,d,a)$	$lex(t\|s,d,a)$
		$P(s\|t,c_3)$	$P(t\|s,c_3)$	$lex(s\|t,c_3)$	$lex(t\|s,c_3)$
		$P(s\|t,d,c_3)$	$P(t\|s,d,c_3)$	$lex(s\|t,d,c_3)$	$lex(t\|s,d,c_3)$
		$P(s\|t,a,c_3)$	$P(t\|s,a,c_3)$	$lex(s\|t,a,c_3)$	$lex(t\|s,a,c_3)$
		$P(s\|t,d,a,c_3)$	$P(t\|s,d,a,c_3)$	$lex(s\|t,d,a,c_3)$	$lex(t\|s,d,a,c_3)$
	...				
	c_n	$P(s\|t)$	$P(t\|s)$	$lex(s\|t)$	$lex(t\|s)$
		$P(s\|t,d)$	$P(t\|s,d)$	$lex(s\|t,d)$	$lex(t\|s,d)$
		$P(s\|t,a)$	$P(t\|s,a)$	$lex(s\|t,a)$	$lex(t\|s,a)$
		$P(s\|t,d,a)$	$P(t\|s,d,a)$	$lex(s\|t,d,a)$	$lex(t\|s,d,a)$
		$P(s\|t,c_n)$	$P(t\|s,c_n)$	$lex(s\|t,c_n)$	$lex(t\|s,c_n)$
		$P(s\|t,d,c_n)$	$P(t\|s,d,c_n)$	$lex(s\|t,d,c_n)$	$lex(t\|s,d,c_n)$
		$P(s\|t,a,c_n)$	$P(t\|s,a,c_n)$	$lex(s\|t,a,c_n)$	$lex(t\|s,a,c_n)$
		$P(s\|t,d,a,c_n)$	$P(t\|s,d,a,c_n)$	$lex(s\|t,d,a,c_n)$	$lex(t\|s,d,a,c_n)$

Table 3.2: Example translation models with dependencies on the domain d, alignment a, and contexts $c_1, c_2, ...c_n$.

prior to building the model.

The increasingly complex nature of each model is shown in Table 3.2. In a real world-system we would probably mix and match from the various approaches. Some dependencies we might define as a single independent feature, whereas others would be modeled jointly. Unfortunately, knowing which features to use and which to ignore is not well-defined. Fundamentally, the user is responsible for deciding on the structure of the model and injecting his knowledge concerning which dependencies are important to maintain. It is deeply unsatisfying that in SMT, which prides itself on well-defined statistical models, the process for incorporating new features into these models relies on heuristics, intuition, and trial-and-error.

These problems with incorporating more discriminative features within a standard SMT model stem from attempting to fit the data with a mixture model. The SMT perspective is that each feature describes a simple distribution and during training we learn weights that describe how to mix these distributions together. This is a reasonable form of statistical modeling, but it leaves open the question of what distributions to represent–i.e. what features to use. An illustrative example is the paper by Shen et al. (2009) that states clearly their unique contribution is that "Feature functions defined in this way are robust and ideal for practical translation tasks." The types of linguistic and context information their features capture is not new, but as they demonstrate, other approaches have been imperfect and have struggled to incorporate it. The point is simply that feature engineering in a traditional SMT model is hard. In a complex system like machine translation we don't always know *a priori* what distributions are useful and they may change. We would prefer if the model took care of this automatically.

3.2 Approach

Our approach differs from the traditional SMT approach by starting with a distribution-free, non-parametric model. We then compare fragments of the input to each translation instance with a parametric distance function. The score from the distance function permits us to map each translation instance to a weighted point in the hypothesis space. This process was illustrated in Figure 1.3 of the introduction. When multiple translation instances predict the same hypothesis, we increment the weight by summing over their scores. The model is essentially a kernel density estimator where, instead of the standard Gaussian, the kernel function is a parametric log-linear model. Our modification of the kernel function no longer guarantees that the model will produce a well-formed probability distribution, but this is not a requirement of our translation system.

This hybrid non-parametric model with a parametric distance function has several nice properties. First, the non-parametric model makes no assumptions about the distribution and can fit the shape of the underlying data. Second, the distance features can be maximally discriminative and operate over one specific translation instance in the corpus. This is the preferred approach, but because the distance function also takes the same form as the standard SMT log-linear model, it can also easily incorporate any traditional SMT features. When the features do operate only over a single instance of translation (and do not require computation of sufficient statistics over a larger set of translations like traditional SMT features), the model is incredibly simple to calculate. Last, having a parametric distance function permits us to learn weights for the distance function from parallel data without any particular linguistic knowledge.

Cunei was developed explicitly to provide a framework that enables this modeling approach. The non-parametric model needs to sum over the parametric distance scores of all instances that contribute to a translation. However, we soon learned that maintaining the distance models for each translation instance was unfeasible. Even though the summation of multiple log-linear models is no longer log-linear, we can approximate the summation with another log-linear model. Reducing the collection of instances to one log-linear model considerably simplifies decoding and optimization. Furthermore, the approximation exactly represents the score and the gradient of the entire collection of log-linear models under the current set of parameters. Only during optimization, when the parameters are modified, will the reduced model be merely an approximation.

3.2.1 Formalism

The complete mathematical derivation for the approximate summation can be found in Appendix B. Here we briefly summarize Cunei's approach and compare it with traditional SMT and EBMT models.

A typical SMT model will score a phrase pair with source s, target t, features ϕ, and weights λ using a log-linear model, as shown in Equation 3.1 (converted to real space).

$$score(s,t) = \prod_k \phi_k(s,t)^{\lambda_k} \tag{3.1}$$

An EBMT system identifies features for each example in the corpus. There is no prototypical model for EBMT, but Equation 3.2 demonstrates one method that calculates the total score by summing over all examples with source s' and target t' that are consistent with the phrase pair being modeled. In the strictest sense this means $s = s'$ and $t = t'$, but typically an EBMT system will have some notion of similarity and use examples that don't exactly match the input.

$$score(s,t) = \sum_{s',t'} \prod_k \phi_k(s,s',t',t)^{\lambda_k} \tag{3.2}$$

Cunei's approach, as shown in Equation 3.3, removes the outer summation that makes computation difficult. The introduction of δ, a slack variable, ensures that the score of this model is equal to Equation 3.2. While the inner term initially appears complex, it is simply the expectation of each feature under the distribution of translation instances and can be efficiently computed with an online update. Calculating the features in this manner ensures that the derivative of λ in our model is equal to the derivative of λ in Equation 3.2.

$$score(s,t) = \delta \prod_k \left(\frac{\sum_{(s',t')\in C} \phi_k(s,s',t',t) e^{\sum_i \lambda_i \phi_i(s,s',t',t)}}{\sum_{(s',t')\in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)}} \right)^{\lambda_k} \tag{3.3}$$

Our reduction from Equation 3.2 to the single log-linear model in Equation 3.3 ties together the two different modeling approaches pursued by SMT and EBMT. Indeed, the SMT model of Equation 3.1 is merely a special case of our model when the features for all instances of a translation are constant such that $\phi_k(s,s',t',t) = \phi_k(s,t)\ \forall s',t'$.

3.3 Run-Time

Sufficient statistics for each instance of a translation could be computed off-line, but the space requirements would be much larger than those for the traditional phrase table and much of the data would go unused. As a result, another distinguishing mark of Cunei is that it delays as much computation as possible until run-time. In particular, translations are not retrieved from a pre-built phrase-table, but rather generated dynamically at run-time. Estimating the model at run-time has three key advantages:

Part-of-Speech	PRP	VBZ	TO	VB	VBN	VBN	IN	DT	NNS	.
Lemma	it	seem	to	have	be	build	by	the	ancient	.
Lexical	it	seems	to	have	been	built	by	the	ancients	.

Figure 3.1: Example phrase with multiple type sequences

1. Run-time feature extraction makes it easy to model non-local features dependent on the particular input or surrounding context. Currently we use alignment features that are dependent on the sentence in which an instance is retrieved. In §4.1.1 we will show that sentence-level and document-level context can also be made available and thrown into the mix of features. In addition, we provide the capability to index multiple forms of metadata and will show in §4.1.2 how to use this additional information in modeling. This metadata will differ among otherwise similar instances and can be used to guide the translation process to favor particular instances of translation.

2. Generating the translations at run-time produces a tightly integrated translation model. As described in §2.2, a traditional SMT system performs alignment and phrase-extraction once when the phrase-table is constructed, using a series of heuristics. However, phrase alignment in reality is not so neatly deterministic. For each source phrase in the parallel corpus, there is some probability (perhaps very small) that it translates to any given target phrase with which it has ever co-occurred. Cunei models the phrase extraction at run-time as part of the translation process with a set of alignment features that determine the probability that a given source phrase and target phrase are aligned. These alignment features are part of the final translation model, and during each iteration of optimization the weights for these features are modified. The new weights change the score for each possible alignment and have the potential to extract different translations. As described in §3.3.2, the extraction and scoring of translations are not two processes, but form one consistent model.

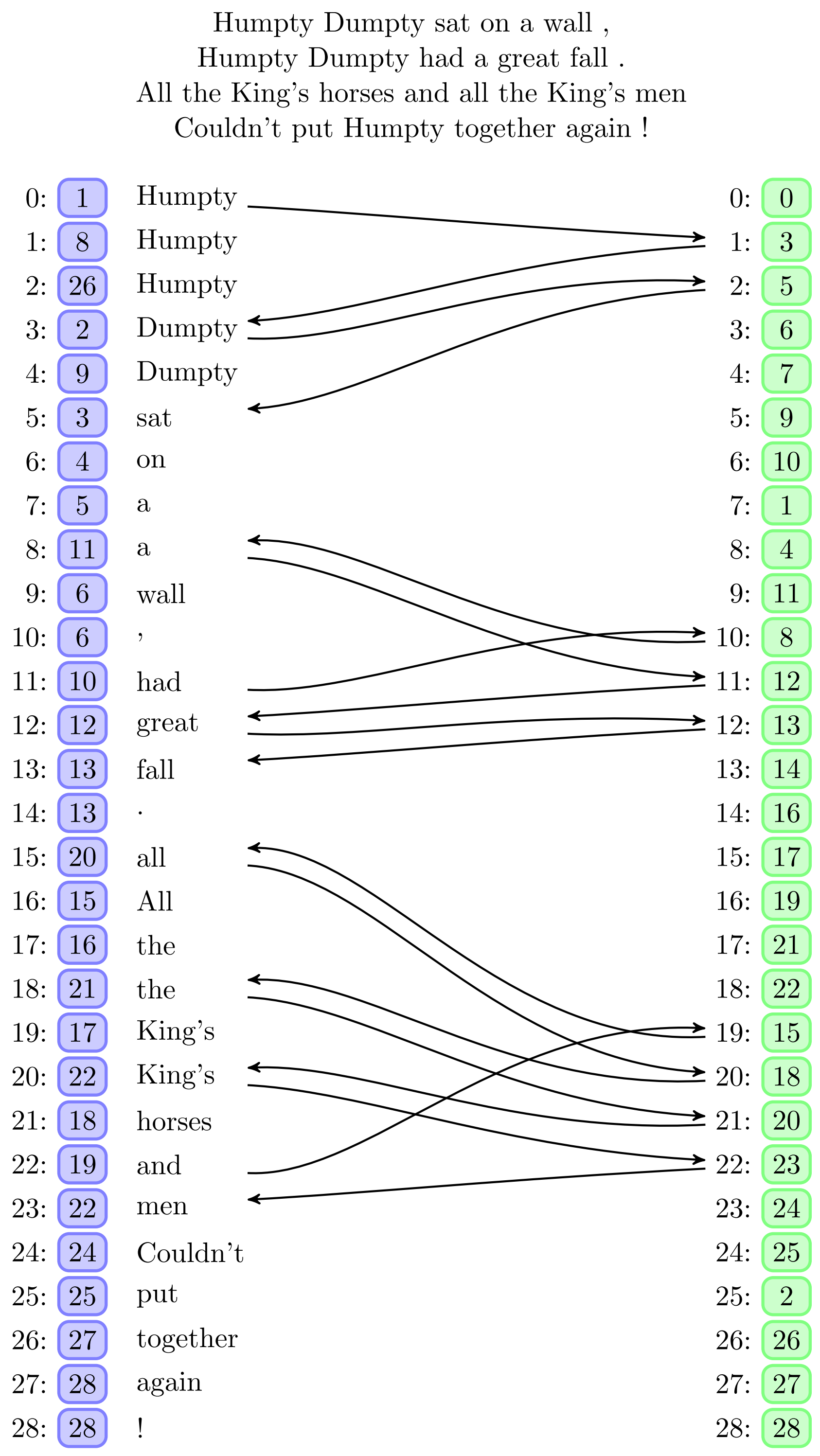
3. The lazy nature of the system allows us to explore a larger search space. We are not required to generate all possible generalizations or compute the similarity among translations *a priori* which would be expensive and necessitate significant pruning. Instead, we dynamically model more complex phenomena as suggested by the data. First, Cunei looks up translations for phrases that exactly, word-for-word, match some span of the input. If the given phrase occurs frequently and Cunei believes it can model the translation well, then its task is complete. In situations where this is not sufficient, Cunei can extend the search space and retrieve similar phrases or generalized templates in order to better model the translation process.

The remainder of this section will describe in detail how translations are constructed at run-time.

3.3.1 Matching

When given a new input to translate, Cunei searches the source-side of the corpus for phrases that match any sub-section of the input. The input phrase and the corpus may consist of multiple type sequences as shown in Figure 3.1. Minimally, they will both contain a lexical sequence representing the unaltered surface strings (or less precisely, 'words'). Lemmas, part-of-speech, or statistical cluster labels may be used to generate alternative types of sequences.

To support retrieving translations at run-time, Cunei constructs a suffix array index for each type of sequence present in the parallel corpus. Suffix arrays provide a compact and efficient data structure for locating for arbitrary sequences of tokens within a large corpus (Yamamoto and



The array on the left is sorted by suffix and contains the position in the corpus of the next token. The array on the right maps a position in the corpus to an index in the left array of sorted suffixes.

Figure 3.2: A suffix array with position information.

Church, 2001). The search algorithm has a worst-case time complexity of $O(m \log_2 n)$ where n is the number of tokens in the index and m is the number of tokens in the phrase being looked up.[1] As evidenced by the work of Brown (2004), Zhang and Vogel (2005), Callison-Burch et al. (2005), and Lopez (2008), they are also increasingly popular in machine translation

As shown in Figure 3.2 we extend the traditional suffix array to include position information. A suffix array is a *sorted* list of suffixes, and while it is very efficient for searching, we lose the information regarding where each suffix is located in the corpus. Indeed, without iterating from the very beginning it is impossible to reconstruct the prefix for a known suffix. Our solution is to store two arrays. Instead of storing a pointer to the location of the suffix in the sorted array, we store a pointer to the position in the corpus of the next suffix. In addition we maintain an array that maps from each position of a suffix in the corpus to the location of the suffix in the sorted array. The position information also permits us to store sequences of types that span varying amounts of data and align these tokens across multiple indexes.

We have attempted to carefully engineer this code to minimize the size of the index and permit efficient access. One such optimization is that Cunei is able to represent the index as a bit array. The bit array is dynamically adjusted to use the minimum number of bytes that are capable of representing the total number of types and tokens present in the corpus. This allows for a much smaller data structure than just representing everything with an integer, and has no upper bound.[2] However, accessing memory that is not integer aligned can significantly decrease performance. It has been our experience that this drop in performance is most sensitive to writes and, in particular, the sorting required to construct the suffix array. The default settings, therefore, maintain an integer-aligned array during construction, but then convert the data structure to a bit array prior to writing it to disk. This reduces the disk space required to store the model and also reduces the size of the data structure that Cunei must load the next time it runs. Furthermore, we memory map the bit array which dynamically loads only those regions of the data structure that are actively used. In addition, the memory mapping permits Cunei to use data structures that are larger than resident memory by relying on the operating system to swap in the required pages.

By storing multiple indexes, we can efficiently search additional types of metadata such as those shown in Figure 3.1. Due to our separate storage of positional information, this metadata does not not have to have a 1:1 mapping with the original lexical sequence. However, we are only able to index metadata that is sequential in nature and covers the entire sentence. For example, we can currently store base-phrase chunks, but not parse trees.

For each type of sequence present in the input, the respective suffix array for that type of sequence is queried. The corpus is scoured for all partial matches of the input; a match may contain as few as one of the tokens from the input or exactly match the entire input. We start searching for the smallest possible match–one token–and incrementally add one more token to the sequence. By storing the bounds in the suffix array from each match, we can limit the search for additional tokens when we extend the sequence. Furthermore, once we are no longer able to locate a particular sequence of tokens in the corpus, we can also rule out searching for any sequences that subsume it. The collection of corpus matches is stored as a lattice with each element indexed by the span of the input it covers.

The corpus instances are not required to be exact matches of the input. For example, a source phrase retrieved by matching only a part-of-speech sequence may be structurally similar to the

[1] By storing an additional data structure for the longest common prefix between neighboring rows in the suffix array, it is possible to reduce the search time to $O(m + \log_2 n)$ (Manber and Myers, 1990), but this is not currently implemented in our system.

[2] We have indexed corpora that must address more than $2^{31} - 1$ bits, the largest value of an `Integer` in Java, but have not yet encountered a situation that exceeds $2^{63} - 1$ bits, the maximum value of a `Long`.

input, but it is likely to be semantically unrelated. Matches such as these do not as-is provide valid translations, but they do still contain useful information about the translation process. When we locate a match according to one sequence type, we are able to use the positional information we stored in the corpus to retrieve all other sequence types present in the corpus at that position.

It is also possible to locate matches with gaps without completely matching one of the sequence types. The suffix array does not permit an efficient mechanism for locating matches with insertions or deletions. However, we store all of our partial matches in a lattice. Therefore, we can iterate over all the matches in the lattice and determine if any two matches come from the same sentence. If we find two matches that are correctly ordered and occur with a few extra tokens between them, then we form a new translation instance with a gap. Due to sampling this technique is not guaranteed to find all possible discontiguous phrases. A more robust (and also more complicated) alternative is described by Lopez (2007). We have not implemented this approach as we have focused on enabling indexing and retrieval of multiple sequential types which can be searched much more efficiently. Furthermore, the types of discontiguous phrases that our naïve method will not locate are those that are extremely frequent. When the translation occurs frequently it is both likely to be modeled well by its components and be located via a more general type sequence.

For each token in a match that does not lexically match the input, a gap is formed. The gaps are projected to the target during phrase alignment in order to form translation templates. Metadata about the gap, such as part-of-speech tags, is preserved in order to aid selection of a valid replacement. The translation templates permit modeling discontiguous phrases, but the gaps are not restricted and may not always follow linguistically-motivated paradigms. As such, these templates allow for the formation of novel phrases, but also add risk. Generally, we will want to prefer exact phrasal translations if they are present. At this stage, the instance of translation is modeled as if the gap did not exist. In §3.3.4 we will learn how the decoder fills these gaps and appropriately scores the combination.

The process of locating all matches in the corpus is relatively cheap, but what we do with each match–notably perform alignment and generate a translation instance–is expensive. Furthermore, we have found in practice that using every last match is overkill and an adequate model can be constructed using a much smaller sample. In order to select an appropriate sample, we perform two levels of sampling. The size of each sample is a parameterized option. Typically, we will first load information from the corpus for one thousand matches. These matches are sampled uniformly over the entire range with the exception that complete matches (i.e. those whose prefix is the start-of-sentence marker and whose suffix is the end-of-sentence marker) are selected first. Each of these matches are then scored according to those feature functions that can be calculated based only on the source. Currently, this stage is merely a place holder, but in the future we will apply features that determine how similar the match is to the input, such as the sentential and document context described in §4.1.1. Using these partial scores, the sample is then resampled. A few hundred matches are selected as the most promising and retained for alignment.

3.3.2 Alignment

After a match is found on the source-side of the corpus, Cunei must determine the target phrase to which it aligns. The alignment is treated as a hidden variable and not specified during training. Instead, Cunei uses statistical methods to induce a phrase alignment. Ideally, the full alignment process would be carried out dynamically at run-time. Unfortunately, even a simple word alignment such as IBM Model-1 is too expensive. Instead, we run a word-aligner offline and use the word alignments as features for an on-line phrase alignment. While the calculations are not exactly the same, conceptually this work is modeled after Vogel (2005).

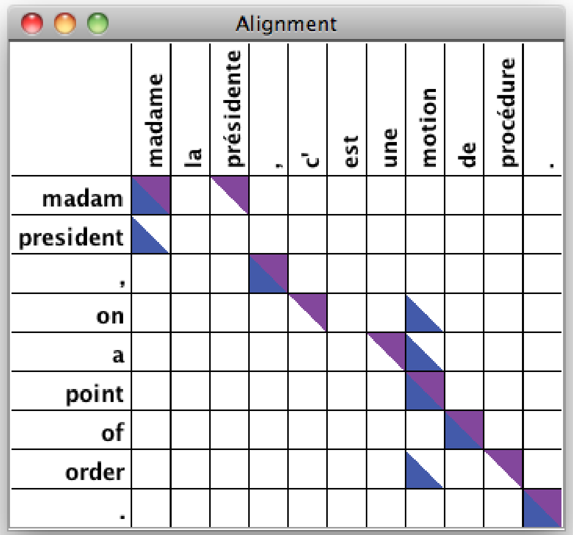


Figure 3.3: Alignment visualization

For each source-side match in the corpus, an alignment matrix is loaded for the complete sentence in which the match resides. The alignment matrix, as visually depicted in Figure 3.3, contains scores for all possible word correspondences in the sentence-pair. Each word-alignment link maintains two scores: α_s and α_t. When using GIZA++ (Och and Ney, 2003) to generate the initial word alignments, $P(s_i|t_j)$ as modeled by GIZA++ is saved as $\alpha_s(i,j)$ and $P(t_j|s_i)$ as $\alpha_t(i,j)$. Cunei also supports initializing the alignment matrix using the Berkeley aligner (Liang et al., 2006). In this case $\alpha_s(i,j)$ and $\alpha_t(i,j)$ will both be set to $P(s_i,t_j)$ as the Berkeley aligner model symmetrizes the probability model. While both GIZA++ and Berkeley model probability distributions, the α scores need not be normalized for our calculations.

Each source phrase is modeled as having some probability of aligning to every possible target phrase within a given sentence. When a source phrase is aligned to a target phrase, it implies that the remainder of the source sentence that is not specified by the source phrase is aligned to the remainder of the target sentence not specified by the target phrase. As detailed in Table 3.3, separate features model the probability that the word alignments for tokens within the phrase are concentrated within the phrase boundaries and that the word alignments for tokens outside the phrase are concentrated outside the phrase boundaries. In addition, words with no alignment links or weak alignments links demonstrate uncertainty in modeling. To capture this effect, we incorporate two more features to measure the uncertainty present in a given phrase alignment.

For each match in the corpus, Cunei uses these feature functions to extract a scored n-best list of phrase alignments. The size of the n-best list is controlled by two user-defined pruning parameters: a maximum number of elements and a maximum ratio between the best and worst score. In practice, 3 to 6 phrase alignments are typically selected. Each possible alignment forms a new instance of the source phrase translating as the target phrase.

Outside Probability
The set of positions in the source phrase and target phrase that are outside the phrase alignment under consideration are, respectively, s_{out} and t_{out}.

`Alignment.Weights.Outside.Source.Probability`	$\sum_{i \in s_{out}} \log \frac{\epsilon + \sum_{j \in t_{out}} \alpha_t(i,j)}{\epsilon + \sum_j \alpha_t(i,j)}$
`Alignment.Weights.Outside.Target.Probability`	$\sum_{j \in t_{out}} \log \frac{\epsilon + \sum_{i \in s_{out}} \alpha_s(i,j)}{\epsilon + \sum_i \alpha_s(i,j)}$

Inside Probability
The set of positions in the source phrase and target phrase that are inside the phrase alignment under consideration are, respectively, s_{in} and t_{in}.

`Alignment.Weights.Inside.Source.Probability`	$\sum_{i \in s_{in}} \log \frac{\epsilon + \sum_{j \in t_{in}} \alpha_t(i,j)}{\epsilon + \sum_j \alpha_t(i,j)}$
`Alignment.Weights.Inside.Target.Probability`	$\sum_{j \in t_{in}} \log \frac{\epsilon + \sum_{i \in s_{in}} \alpha_s(i,j)}{\epsilon + \sum_i \alpha_s(i,j)}$

Inside Unknown
The score threshold below which an alignment link is considered uncertain is θ.

`Alignment.Weights.Inside.Source.Unknown`	$\sum_{i \in s_{in}} \max(0, \frac{\theta - (\epsilon + \sum_j \alpha_t(i,j))}{\theta})$
`Alignment.Weights.Inside.Target.Unknown`	$\sum_{j \in t_{in}} \max(0, \frac{\theta - (\epsilon + \sum_i \alpha_s(i,j))}{\theta})$

Table 3.3: Phrase alignment features

Phrase Frequency
The number of occurrences of the source phrase and the target phrase in the corpus are, respectively, c_s and c_t.

`Phrase.Weights.Frequency.Correlation`	$\frac{(c_s-c_t)^2}{(c_s+c_t+1)^2}$
`Phrase.Weights.Frequency.Source`	$-\log(c_s)$
`Phrase.Weights.Frequency.Target`	$-\log(c_t)$

Lexical Probability
The conditional probabilities of the source words s and target words t are relative frequency counts using the word alignments over the entire corpus.

`Lexicon.Weights.Lexical.Source`	$\sum_{i\in s}\max_{j\in t}\log P(s_i\|t_j)$
`Lexicon.Weights.Lexical.Target`	$\sum_{i\in t}\max_{j\in s}\log P(t_i\|s_j)$

Length Ratios
The mean, μ, and variance, σ^2, of the lengths are calculated over the entire corpus.

`Phrase.Weights.Ratio.Word`	$-\frac{(\|s\|_{word}*\mu_{word}-\|t\|_{word})^2}{\sigma^2(\|s\|_{word}*\mu_{word}+\|t\|)}$
`Phrase.Weights.Ratio.Character`	$-\frac{(\|s\|_{char}*\mu_{char}-\|t\|_{char})^2}{\sigma^2(\|s\|_{char}*\mu_{char}+\|t\|)}$

Table 3.4: Static SMT-like features

3.3.3 Scoring

As described previously each translation instance is scored with a log-linear model incorporating many features. The model for each translation instance is unnormalized and reflects a score proportional to $P(s,t)$. When two translation instances predict the same hypothesis, their scores are summed. The scoring framework is intentionally flexible such that features are not hard-coded and any number of features can be added to the model at run-time. The model is progressively updated and new features are added as the necessary information becomes available. Recall that during matching we had the option of applying features that measure the source-side similarity or relevance of each translation instance. This direction will be further explored in our thesis work (§4.1.1 and §4.1.2) to generate dependencies based on the input and the particular sentence or document in the corpus in which the instance was found. Next, during phrase alignment we introduced several features that were dependent on the complete sentential alignment for each translation instance. After the translation instances are retrieved from the corpus, we can also apply more general SMT-like features. We currently add features representing a translation's overall frequency in the corpus, lexical probabilities, and length distribution as detailed in Table 3.4. These particular features are static in the sense that they do not change from instance to instance if they share the same source and target phrases. They are knock-offs of traditional SMT features and useful to ensure our model has no less discriminative power. While the emphasis is placed on Cunei's ability to use per-instance features, in this manner, Cunei can also take advantage of features computed over sets of instances or loaded from external models.

3.3.4 Decoding

Cunei synthetically combines the partial translations into a complete sentence using a statistical decoder. Recall that the set of matches retrieved from the corpus particular to the input sentence are stored in a lattice. These matches are aligned to form partial translations which are in turn stored in a lattice as illustrated in Figure 3.4. The decoder then searches this latter lattice for a set of partial translations with the minimum score that completely cover the input.

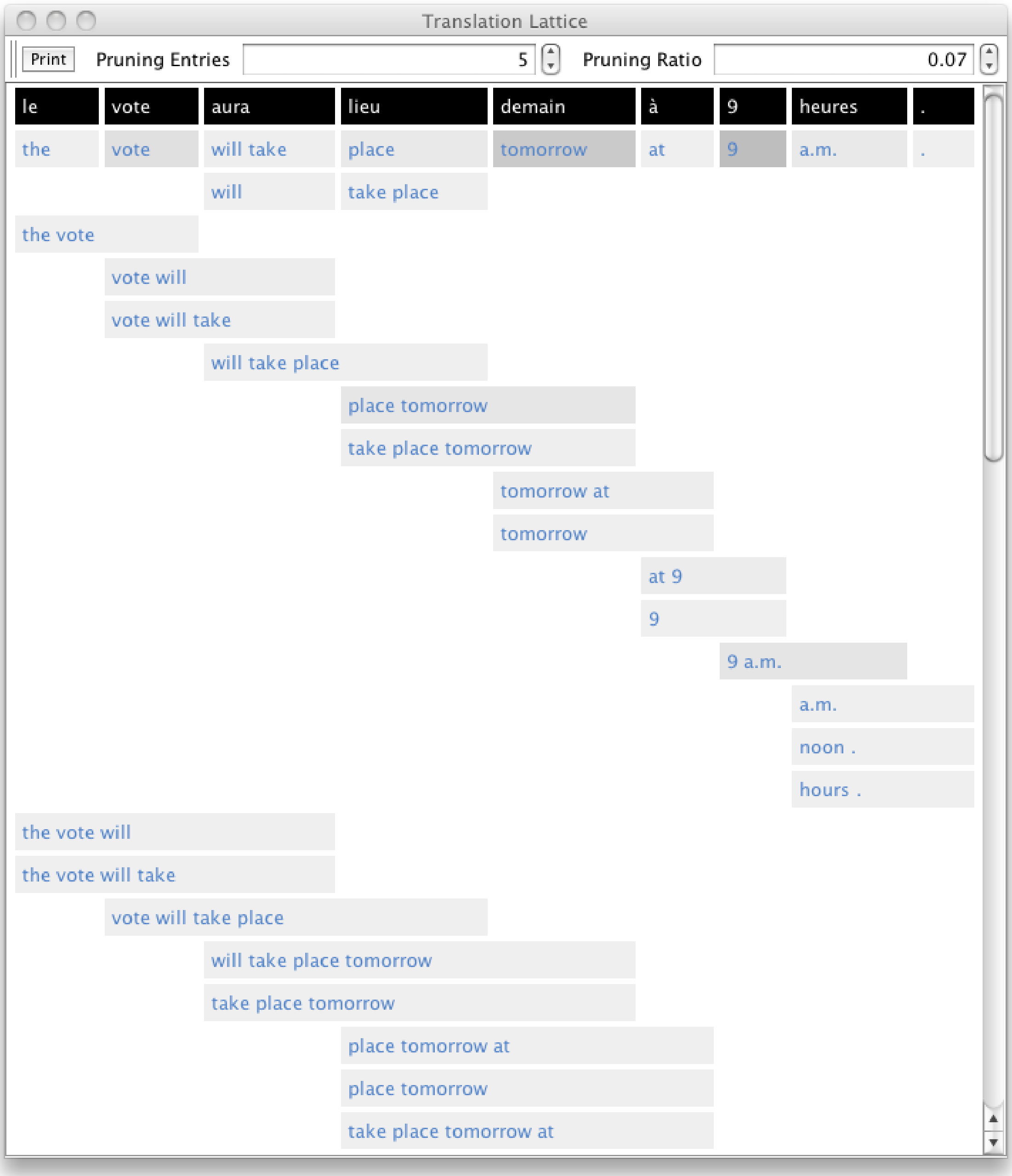
The decision of which partial translations to combine is informed by the scores of the partial translations and the decoding features detailed in Table 3.5. We would generally prefer to use fewer, longer partial translations. This is measured by including a feature that counts the number of nodes that were combined. Occasionally no partial translation exists for a span of the input or a partial translation exists with a very low score. In these situations the decoder may introduce an epsilon arc. A binary feature is set on these arcs so that they can receive a low score and are only used in extreme circumstances. In order to compensate for divergences between the source and target language, Cunei may need to reorder the partial translations. The reordering is modeled by counting the total number of re-orderings and by keeping track of the total distance that words have been moved. Additionally, the probability of the complete target sequence can be estimated with a statistical language model.[3] Any number of language models can be used simultaneously during decoding and each will generate its own set of features. However, the language model probabilities decrease as the sentence length increases. In order to offset this tendency toward short output, we balance it with one feature that counts the number of words present in the target. The complete sentence length is also modeled based on the ratio of source words to target words like we did with each partial translation.

In addition to partial translations that were retrieved from the corpus, the lattice provided to the decoder also contains translation templates. When translation templates occur within a particular input span, the decoder must fill the gap to form a valid partial translation. In the early stages, the decoder acts like a chart parser composing all partial translations with one source word before those with two source words, which in turn are calculated before those with three source words, and so on. For each coverage span, a separate (pruned) beam is stored of hypotheses. Each hypothesis is the composition (possibly reordered) of one or more partial translations. This guarantees that by the time we encounter a translation template with a gap, there will already exist a beam of hypotheses that can be used to fill the gap. The process of combining the translation template with a previously composed hypothesis is shown in Figure 3.5. Currently, Cunei supports the gap-filling process, but does not additionally score this form of composition which is a point we will address in §4.1.2. Once sufficiently-long arcs are formed and no gaps remain in the lattice, the decoder switches to a beam-search strategy which is much more efficient.

3.3.5 Optimization

Because–at the end of the day–Cunei uses a log-linear model, we can now optimize it using the same approaches developed for SMT. The most common approach, MERT, was described in §2.4. MERT iteratively generates an n-best list of translations from the current set of weights and finds a new set of weights that maximize an objective function (typically BLEU). Due to pruning and the beam search within the decoder, the new weights may yield different translations in the n-best list. This is traditionally remedied by merging the n-best lists after each iteration to obtain a

[3] Cunei supports using a language model during decoding but currently relies on external software to build the language model. Typically, we build a 5-gram language model with modified Knesser-Ney smoothing using the SRILM toolkit.


Translation Lattice

Print Pruning Entries 5 Pruning Ratio 0.07

le	vote	aura	lieu	demain	à	9	heures	.
the	vote	will take	place	tomorrow	at	9	a.m.	.
		will	take place					
the vote								
	vote will							
	vote will take							
		will take place						
			place tomorrow					
			take place tomorrow					
				tomorrow at				
				tomorrow				
					at 9			
					9			
						9 a.m.		
							a.m.	
							noon .	
							hours .	
the vote will								
the vote will take								
	vote will take place							
		will take place tomorrow						
		take place tomorrow						
			place tomorrow at					
			place tomorrow					
			take place tomorrow at					

Figure 3.4: A lattice of translations

Hypotheses

`Hypothesis.Weights.Nodes`	Number of partial translations used to construct a hypothesis
`Hypothesis.Weights.Epsilon`	Number of epsilon translations used to construct the hypothesis

Reordering

`Hypothesis.Weights.Reorder.Count`	Number of times target phrases were reordered
`Hypothesis.Weights.Reorder.Distance`	Absolute difference between the first position of the current partial translation and one greater than the last position of the previous partial translation

Language Model

Multiple language models can be used–these refer to the model identified as `Default`.

`LanguageModel.Default.Weights.Probability`	Log probability of the target sequence $w_0 w_1 w_2 ... w_n$, generally calculated as $\sum_i \log P(w_i \mid w_{i-i} w_{i-2} w_{i-3} w_{i-4})$
`LanguageModel.Default.Weights.Unknown`	Number of unknown words

Sentence Length

The mean, μ, and variance, σ^2, of the lengths are calculated over the entire corpus.

`Sentence.Weights.Length.Words`	$\|t\|_{word}$
`Sentence.Weights.Ratio.Word`	$-\frac{(\|s\|_{word} * \mu_{word} - \|t\|_{word})^2}{\sigma^2(\|s\|_{word} * \mu_{word} + \|t\|)}$
`Sentence.Weights.Ratio.Character`	$-\frac{(\|s\|_{char} * \mu_{char} - \|t\|_{char})^2}{\sigma^2(\|s\|_{char} * \mu_{char} + \|t\|)}$

Table 3.5: Decoder features

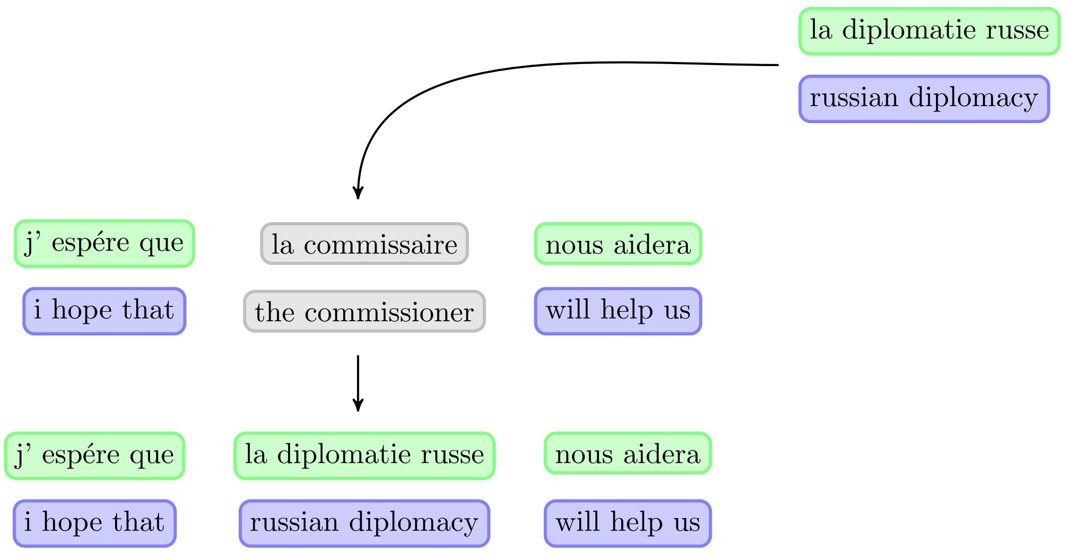


Figure 3.5: Combining two translations to fill a gap

larger representation of the search space. In Cunei, new weights can also change what translations are found in the corpus and affect the feature calculations. Recall that Cunei's model is only an approximation once the parameters change. In order to manage this variability, we merge the n-best lists after each iteration, but always select the most recent model. Furthermore, stale models–translations that appeared in an n-best list several iterations back that we have not seen recently–are progressively down-weighted based on their age and eventually pruned. Cunei must also manage a larger feature space, and as discussed previously, in practice MERT is not an ideal choice when the feature-set becomes large. As such, Cunei's optimization code closely follows the approach of Smith and Eisner (2006), which uses an objective function that minimizes the expected loss over the distribution of translations present in the entire n-best list.

Cunei optimizes toward BLEU using the following objective function that in log-space sums the expected value of BLEU's brevity penalty and precision score:

$$(1+e^{\mu(h)-\mu(r)})(\frac{\mu(|r|)}{\mu(h)}e^{\frac{\sigma(h)}{2\mu(h)^2}-\frac{\sigma(r)}{2\mu(r)^2}}-1)+\frac{\sum_{n=1}^{4}\log(\mu(t_n))-\frac{\sigma(t_n)}{2\mu(t_n)^2}-\log(\mu(c_n))+\frac{\sigma(c_n)}{2\mu(c_n)^2}}{4}$$

such that,

$$\begin{aligned}\mu(x) &= \sum_i p_i x_i\\ \sigma(x) &= \sum_i p_i(x_i-\mu(x))^2\\ p_i &= \frac{e^{\gamma\phi(i)}}{\sum_k e^{\gamma\phi(k)}}\end{aligned}$$

$\phi(i)$	is the log-score of hypothesis i in the n-best list
c_n	is BLEU's "modified count" of n-grams common to the hypothesis and references
t_n	is the total number of n-grams present in the hypothesis
h	is the length of the hypothesis
r	is the length of the selected (shortest or closest) reference

Following Smith and Eisner (2006), in order to avoid local minima, the distribution of the n-best list is slowly annealed. This is carried out by multiplying the log-score of each translation by a γ parameter; our default setting initializes gamma to 0.25 and continues to double it after convergence until it reaches a value of 8. Hence, we begin with a nearly flat distribution and mildly peak the distribution each time the objective function converges. Eventually this process reaches a distribution where, for each sentence, nearly all of the probability mass resides on one translation. We also followed Smith and Eisner (2006) in adding support to maximize entropy prior to minimizing the objective function. However, we found that this resulted in much longer run times with no significant benefit and it is disabled by default. In the early stages, sharpening the distribution is often the quickest way to minimize the expected loss. While γ is fixed until convergence, the same effect is seen by all the other weights uniformly increasing in magnitude. To address this weight creep, Cunei augments the objective function with an L2 normalization term. Last, Cunei supports the ability to decode sentences *toward* a particular set of references. Obviously, under normal circumstances this is cheating, but initializing the n-best list with these translations does help guide the process toward high-scoring, obtainable translations.

3.4 Experiments

	Finnish-to-English		**French-to-English**		**German-to-English**	
Types	499,770	84,257	106,862	87,083	273,960	86,671
Tokens	21,492,772	29,744,581	34,979,287	32,001,553	29,730,317	31,156,576
Sentences	1,121,312		1,207,184		1,165,545	

Table 3.6: Corpora statistics

To compare Cunei with Moses on several languages, we chose the freely-available Europarl corpus (Koehn, 2005), and evaluated the performance on translating from Finnish, French, and German into English. We followed the normal practice of reserving the proceedings from the fourth quarter of 2000 for evaluation and trained each system on the remainder of the parallel corpora. Statistics for each of these three parallel corpora are shown in Table 3.6. Cunei applied light pre-processing, filtering, and tokenization suitable for Western languages to each corpus; the corpora were subsequently word-aligned by GIZA++ in both directions. As all of the systems translated into English, we built one language model consisting of 243 million words from the Europarl and a portion of the English newswire released by the 2009 Workshop on Statistical Machine Translation.[4] An identical tokenized, word-aligned parallel corpus and language model were provided to Cunei and Moses for each language pair, and the translation systems were trained using their default configurations. Specifically, Cunei only used the features that were outlined in Tables 3.3, 3.4, and 3.5.

The results of the evaluation are shown in Table 3.7. Each system was evaluated using BLEU (Papineni et al., 2002), since both Moses and Cunei have built-in support to optimize their weights

[4] `http://www.statmt.org/wmt09/`

	Finnish-to-English		**French-to-English**		**German-to-English**	
	Dev	Test	Dev	Test	Dev	Test
Moses	0.2445	0.2361	0.3207	0.3219	0.2746	0.2546
Cunei	0.2456	**0.2369**	0.3215	**0.3225**	0.2813	**0.2634**

Table 3.7: Evaluation of Cunei and Moses on Europarl using BLEU (Phillips and Brown, 2009)

Finnish -to- English	*Moses*	mr president, indeed himaren orgy of violence and electoral fraud in local elections in the province, who were living in the region kreikkalaisvähemmistöön.
	Cunei	mr president, indeed in the province of violence and electoral fraud in local elections, which were against the greek minority in the area.
	Reference	madam president, it is quite right that the municipal elections in himara were marked by violence and fraud at the expense of the greek minority living there.
French -to- English	*Moses*	for some reason, i know that my name is not on the attendance register.
	Cunei	for some reason i do not know, my name is not on the attendance register.
	Reference	for some strange reason, my name is missing from the register of attendance.
German -to- English	*Moses*	i would like to criticise the lack of initiatives on the new challenges in the safety of employee participation and in industrial relations.
	Cunei	i share the criticism of initiatives to the new challenges in the field of health and safety, and worker participation in the labour relations.
	Reference	i would like to raise a criticism in connection with the lack of initiatives produced in response to the new challenges we face in health and safety at work, employee participation and labour relations.

Table 3.8: Comparison of translation output

toward BLEU using a development set. The development set and a blind test were extracted from the evaluation portion of the Europarl collection. Documents were selected using the same set of dates across all corpora, resulting in the English references consisting of roughly the same text. The development set contained close to 3,000 sentences and the evaluation set was nearly 4,000 sentences, with no overlap.

Cunei tops Moses ever so slightly in French-to-English, which is the language pair with the fewest divergences and highest overall scores. Interestingly, performance is also nearly identical on Finnish-to-English even though word order in Finnish is much freer and can require substantial reordering during decoding. This result is even more surprising given that reordering has not (yet) been a significant focus in Cunei and our reordering model is only informed by how frequently and how far phrases are moved during decoding. On the other hand, Moses is using a lexicalized distortion model that is much more sophisticated. Only in German-to-English do we see a distinction between Moses and Cunei. We suspect that the key linguistic divergence affecting translation is German compounding. This results in a greater prevalence of one-to-many alignments which makes phrase alignment and extraction more difficult. Cunei shines due to its run-time phrase-extraction which was able to adapt during training and learn appropriate weights for aligning and extracting German-English phrase pairs from less-than-perfect word alignments.

Table 3.8 provides one example of translation output for each language pair. The most noticeable difference in the output is that Cunei is more likely to find translations of ‘unknown’ words and tends to exhibit better word selection. However, in comparison to Moses, this at times comes at the cost of less faithful word ordering.

3.5 Summary

As shown in Table 3.7 and published in Phillips and Brown (2009), Cunei is state-of-the-art across several language-pairs. In recent years SMT has dominated the field of machine translation research. Even though SMT in many respects grew from the data-driven focus of EBMT, the concept of modeling each translation instance individually has been lost. While we do not (yet) claim to outperform Moses, our approach opens the door to new sources of knowledge that are permitted to describe each instance of translation differently. The development of Cunei has laid the groundwork for this thesis and provides a comprehensive framework for further research and experimentation. Ultimately, Cunei should be able to use any available information, be it lexical, syntactic, semantic, grammatical, pragmatic, contextual, etc., to make a case-by-case selection of the best possible instance of translation in its corpus–fulfilling the goal of true ‘data-driven’ machine translation.

Chapter 4

Thesis Work

What has been presented so far is a novel platform for machine translation, but we have not yet taken full advantage of all it has to offer. In the thesis work outlined below we intend to push the boundaries. The majority of our work will focus on extending the translation model with new instance-specific features that our model is in a unique position to exploit. In addition, we hope to investigate the generality of our approach by applying it to another type of data-driven model: language modeling.

4.1 Translation Modeling

As illustrated originally in Figure 1.3 from the introduction, we can view our model of each translation instance as a composition of three functions: $\phi_1(source, source')$, $\phi_2(source', target')$, and $\phi_2(target', target)$. Recall that ϕ_2 models the likelihood of translation while ϕ_1 and ϕ_3 model the distance of our translation instance to either the input or the output. The description of Cunei so far has been limited to features that fall into the category of ϕ_2. First, we will incorporate ϕ_1 by modeling non-local context features. Then we will extend our modeling of ϕ_1 and ϕ_2 with a large quantity of metadata. Last, we will utilize ϕ_3 by modeling similarity among the set of predicted translations.

4.1.1 Contextual Features

One of the reasons we have argued for separately modeling each instance of a translation is that it allows for a more nuanced differentiation between each possible translation present in the corpus. Translations that occur within the same topic as the input, have the same genre as the input, or are simply from a specific collection of documents may prove to be more relevant. We would like to exploit this non-local information that is embedded within the surrounding context of each translation instance, but not directly contained within the translation's phrase pair. Our approach extends well to this situation as our model permits some instances to have a greater impact on the translation model.

If the corpus is relatively homogenous and composed of high-quality, in-domain translations, then this distinction may not be important. However, the problem we typically face is having additional out-of-domain corpora or low-quality comparable corpora that we do not know how to use. Often, when dealing with data of varying quality, training a SMT system on all of the data *degrades performance*. A common work-around is to perform some sort of heuristic sub-sampling that selects a small quantity of novel phrase pairs from the large out-of-domain corpus such that they do not overwhelm the number of phrase pairs extracted from the smaller in-domain corpus. Our approach seeks to avoid this situation, permitting the system to train on *all* available data by dynamically weighting each instance of a translation. We weight each instance by adding features to ϕ_1 which measures the distance between the input and the source of the instance. In this fashion, we move from a heuristic process to one that explicitly models the relevance of each translation instance. Since the distance metric is incorporated into our log-linear translation model, we can automatically learn weights that maximize BLEU (or any other objective function).

Conceptually, the terms context, topic, and genre all refer to the same idea at different levels of granularity. We will capture this effect by modeling documents as a bags of words and comparing each input document to each corpus document. To this end we represent each document with a frequency vector of types. Extremely frequent words, such as 'the' in English, provide little information and may be discarded. A wide range of document-level distance metrics have been invented and explored within information retrieval. Currently, we have implemented two such metrics: cosine distance and Jensen-Shannon distance (Lin, 1991). Both of these are relatively simple to compute and generally perform well. Table 4.1 shows the calculation for each of these new features.

Some corpora have well-defined document boundaries, but this information is not available in other corpora. Also, if we are modeling local-context we may want to use sections of text smaller than a document, while modeling genre may require sections of text larger than a document. Thus, in addition to using actual document boundaries, we permit the use of arbitrarily-sized windows of sentences. Each window size permits the capture of contextual clues at varying levels of granularity. The number of sentences per window is specified by the user and multiple windows may be used simultaneously. Each window size can be viewed as creating a new collection of pseudo-documents. For each context window we replicate the document-level context features described above and compute them over these pseudo-documents. Due to performance considerations, the context windows only apply to the corpus and not the input.

As the size of the window becomes smaller, the frequency vectors will become sparse and may no longer provide adequate information for comparison. In order to still capture local, intra-sentential context, we additionally favor instances of translations in which the words immediately to the left and/or right also match the input. For example, if we wish to translate "Je bois" we would likely retrieve an instance from the corpus in which we would be able to align both "Je bois" to "I drink" and "bois" to "drink". Normally, the decoder will prefer to select the longest possible match, but let's pretend this phrase is part of a longer sentence and based on the translations available for the rest of the input we need to produce a translation solely for "bois". The problem is that when translating "bois" we will find many instances in the corpus that align to "wood". This is a valid translation of "bois" in general, but not for this context. Our solution is to exploit the fact that the translation "drink" came from an instance in the corpus that also matched one additional word of the input ("Je"). We achieve this by storing the longest possible match that covers the same position as each match we locate in the corpus. Then, when scoring each translation instance, we add features that prefer having long contextual matches with the input. Specifically, we calculate the two new local context features described in Table 4.1 using our knowledge of the longest match.

A related but orthogonal concept is the style of a text. This refers to the speaker, writer, or organization which was the key creative contributor behind the text. For example, The New York Times has a very different writing style than The Associated Press, regardless of what subject is being discussed. We support this by storing the origin of each sentence in the corpus. Then, when an instance of a translation is retrieved from the corpus, we generate a set of binary features indicating where the instance originated. The weights for the origin features can be conceived as mixture weights specifying the relevance of the different types of sources. Currently only a single origin can be stored per sentence, so the user must decide what information is most pertinent.

One direction we hope to expand this work is by supporting the storage of multiple origin labels which would enable more than one of the binary origin features to be set per translation instance. For example, this would permit modeling the speaker and organization simultaneously. It would also provide a broader framework to model additional contextual information. The time period could be represented as one such source if it permits better discrimination among possible translations (e.g. "9/11"). Sections of text could also be differentiated based on their position in the document. For example, the headline of a newswire article varies greatly in style from the body text. Additionally, letters or emails may have specific endings or closing tags. Each of these categories could receive a separate origin label. Thus, one instance of translation could be labeled as spoken by the *British Prime Minister* as an *opening* address and recorded in the *Europarl* proceedings in the year *2008*.

We carried out initial experiments with these context features by training multiple systems on the German-to-English and Czech-to-English parallel data released for the 2010 Workshop on Statistical Machine Translation[1]. The development and test sets were newswire and there were about two million words of parallel news commentary for training. In addition, we trained the German-to-English systems with an updated version of the Europarl[2] that contains approximately 45 million words. The Czech-to-English systems were augmented with a 90 million word parallel corpus collected by Charles University[3] that contained a wide range of genres. We used one billion words from newswire text made available through the workshop and portions of the English Gigaword corpus to construct a single English 5-gram language model with Kneser-Ney smoothing.

The baseline system used the same features and configuration as described in the previous chapter. In our first experiment, we augmented the baseline system by adding the origin and local context features. Second, we applied the document context features, constructing a system that includes all the instance-specific context features listed in Table 4.1. This system used three levels of document context features: the original document boundaries, a window of 7 sentences, and a window of 15 sentences. Results from our experiments are recorded in Table 4.2. The document context features clearly did not perform as well as we had hoped. However, the origin and local context features do show a strong improvement over the baseline.

As discussed in 2.3.1 there is precedent for incorporating context features like these in machine translation. While we know of no existing work that combines these particular features together, the features themselves are not the key contribution. Instead, the importance of this work is the simplicity with which they can be jointly modeled and integrated into our system. We are encouraged by the results of our initial experiments using origin and local context features as they further confirm our proposed paradigm shift in modeling. While we need to run a few more experiments and investigate the issues with the document context features, this component of the thesis is largely complete.

4.1.2 Metadata

Our approach to modeling also makes it convenient to include arbitrary metadata that further expands the system's knowledge of the corpus and its ability to select relevant translations. As discussed in §3.3.1 Cunei views a phrase as being composed of multiple types of sequences. Non-lexical type sequences are one form of metadata. As part of the work below we will extend the notion of metadata to encompass information that is not sequential in nature, such as annotations over words or phrases in the corpus. Fundamentally, metadata is extra knowledge that augments

[1] `http://www.statmt.org/wmt10/`
[2] `http://www.statmt.org/europarl/`
[3] `http://ufal.mff.cuni.cz/czeng/czeng09/`

Document Context

$tf(t,d)$ is the count of type t in corpus document d or the input document d'. $df(t)$ is the log count of documents in the corpus and the input that contain the type t.

$$\alpha = \frac{tf(t_i,d)}{df(t_i)\sum_{t_j}\frac{tf(t_j,d)}{df(t_j)}}$$
$$\beta = \frac{tf(t_i,d')}{df(t_i)\sum_{t_j}\frac{tf(t_j,d')}{df(t_j)}}$$

`Context.Origin.Weights.Cosine`	$\frac{\sum_{t_i}\alpha\beta}{\sqrt{\sum_{t_i}\alpha^2}\sqrt{\sum_{t_i}\beta^2}}$
`Context.Origin.Weights.JensenShannon`	$\sum_{t_i}\frac{\alpha\left(\log(\alpha)-\log(\frac{\alpha+\beta}{2})\right)+\beta\left(\log(\beta)-\log(\frac{\alpha+\beta}{2})\right)}{2}$

Local Context

The longest match is from position p_s to p_e while the current instance being scored covers position m_s to m_e.

`Match.Weights.Context.Skew`	$2\frac{(p_e-m_e)^2+(p_s-m_s)^2}{(p_e-m_e+p_s-m_s)^2}-1$
`Match.Weights.Context.Total`	$p_e-m_e+p_s-m_s$

Origin

The identity of the corpora from which an instance is retrieved is marked with 1 and all others receive 0. Below is an example when there are two corpora, `Europarl` and `News`, and the instance being scored came from the `Europarl` corpus.

`Origin.Weights.Europarl`	1
`Origin.Weights.News`	0

Table 4.1: Context features

German-to-English	**Sampled WMT09 (Dev)**				**WMT08 (Test)**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Baseline	0.1912	5.9010	0.5241	0.6489	0.2060	6.2838	0.5291	0.6543
+ Origin & Local Context	**0.1971**	5.9423	0.5288	0.6435	**0.2115**	**6.3745**	0.5328	**0.6447**
+ Document Context	0.1970	**5.9510**	**0.5293**	**0.6424**	0.2097	6.3397	**0.5342**	0.6467

Czech-to-English	**Sampled WMT09 (Dev)**				**WMT08 (Test)**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Baseline	0.2159	6.2195	**0.5531**	0.6189	0.2005	6.3250	0.5338	0.6472
+ Origin & Local Context	0.2160	**6.2595**	0.5527	**0.6151**	**0.2043**	6.3717	**0.5358**	**0.6413**
+ Document Context	**0.2172**	6.2389	0.5525	0.6220	0.2031	**6.3864**	0.5350	0.6446

Table 4.2: Evaluation of non-local features

the lexical types present in the corpus. Similar to context, we utilize metadata to better model each translation instance, but it is not a requirement for translation. Thus, metadata may be present on the source-side and/or the target-side of the parallel corpus as available.

Recall in §3.3.1 we described Cunei's ability to perform fuzzy matching. By indexing multiple forms of metadata, one is able to retrieve related phrases from the corpus that exactly match one type sequence of metadata (e.g. part-of-speech), but not completely match the lexical type sequence present in the input. Thus, an additional benefit of utilizing metadata is that the system will be better able to retrieve fuzzy matches. Furthermore, when these matches contain gaps that need to be filled by other translations, the metadata will be used by the decoder to guide the replacement process and favor substitutions with compatible metadata.

An early incarnation of our system that used multiple type sequences was presented in Phillips (2007). In addition to standard lexical matching, we used part-of-speech tags to retrieve instances from the corpus that structurally matched the input. Each match was aligned by the system to predict a target sequence. When a word or phrase in the source did not lexically match the input, a gap was formed in the lexical target sequence. Using the alignment links, lexical translations from the corpus were substituted into these gaps to form a synthetic lexical translation. In addition to respecting the word alignment, the substitution was also required to match the target part-of-speech sequence. In this manner, the structural matches identified templates that guided local re-ordering.

To evaluate this setup Cunei was trained on approximately 100,000 sentence pairs of Arabic-English newswire text. At the time this represented all available Arabic-English newswire text from the Linguistic Data Consortium with sentences containing fewer than 50 words. System parameters were tuned using part of the 2003 NIST MT Evaluation data set (MT03), and the 2004 NIST MT Evaluation data set (MT04) was held-out for evaluation. MT04 contains editorial, speech, and news genres, but nearly half of it is news. In addition to reporting the BLEU score for all of MT04, we split MT04 by genre and divided the news genre into two parts. Document boundaries were preserved in all the splits and the chunks range in size from 278 sentences to 387 sentences. Splitting the data in this fashion allowed multiple evaluations while maintaining enough sentences to have meaningful results.

Table 4.3 illustrates the gain in performance from annotating the corpus with part-of-speech tags. It is clear that the structural matching improves translation quality as BLEU scores improved in all testing conditions. While the relative improvement is smallest for "News A", this is still a respectable gain in performance considering the high baseline. "News B", "Editorial", and "Speech", which all have lower baselines, show stronger gains from the structural matching. This suggests

			Partial MT04 by Genre			
	MT03 (Dev)	**MT04 (Test)**	**News A**	**News B**	**Editorial**	**Speech**
Baseline	0.444	0.397	0.483	0.455	0.321	0.339
Structural	0.452	0.412	0.490	0.475	0.329	0.364
Baseline NR	0.419	0.385	0.461	0.434	0.320	0.333
Structural NR	0.446	0.411	0.490	0.470	0.333	0.363

Table 4.3: Evaluation with part-of-speech metadata using BLEU (Phillips, 2007). NR denotes 'No Reordering'.

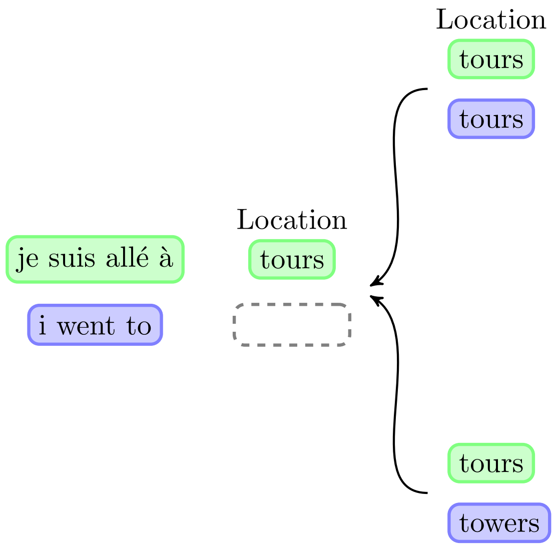


Figure 4.1: Source-language metadata

that the use of additional metadata makes the system more robust and allows it to degrade more gracefully. In addition to evaluating the use of structural matching, we also experimented with reordering disabled. As expected, when reordering is disabled, the performance of the system with only lexical matching drops. Interestingly, this was not true for the system with structural matching enabled, signifying that the structural matching captured most (if not all) of the reordering.

The work described above allowed for gaps but was limited to one form of metadata (part-of-speech tags) that had a 1:1 correspondence with the sequence of lexical tokens. Furthermore, unlike how we have described our current model, this early work did not model the substitution with distance features and, therefore, placed hard constraints on the gap-filling process. Our thesis work seeks to address both of these limitations. First, we will investigate features that compute the distance between divergent metadata annotations on the source and target. Second, we will extend the notion of metadata and model multiple forms of metadata simultaneously.

On the source-side, the metadata can be used to determine how similar a retrieved instance is to the input. An example of how this is useful in the selection process can be found in Figure 4.1. Restricting the metadata of the input to exactly match the metadata for each translation instance will produce high precision translations, but significantly reduce the number of possible translations we are able to retrieve from the corpus. Instead, we will need to introduce features for the distance function ϕ_1 to reflect the divergence between the metadata on the input and the metadata on the source of each translation instance.

Source-side metadata is also relevant when combining translation instances to fill gaps. This process will also utilize distance features but apply them to the metadata present on the two translation instances being combined (instead of between the input and a translation instance). Furthermore, this distance calculation is feasible even when there is no metadata present on the input since it depends solely on the metadata present in the corpus.

We anticipate using the same set of metadata distance features for both the selection process and gap substitution process. Initially we will introduce source-side features based on the F_1 score of 1-grams, 2-grams, 3-grams, and 4-grams. F_1 is computed as the harmonic mean of precision and recall. This should provide a reasonable base, but we expect additional features to be necessary to adequately measure variation across metadata. At this time we do not know what those additional features will be. Our direction will be more clear after we begin using the F_1 scores and analyze their effect on the translation output.

On the target side, the metadata can be used ensure the coherency of the target hypothesis. The obvious situation is when the decoder is filling a gap. Just like on the source-side, the translation template may contain some metadata describing the gap. We would therefore prefer that the translations we insert into the gap have the same or similar metadata. However, the target metadata is also relevant when reordering or concatenating two partial translations. A partial translation may contain the beginning but not the end of an annotation. During decoding we would prefer to pair this partial translation with another partial annotation that has the end of the annotation. Thus, when combining the two partial translations we would be able to compose a complete annotation that spans the joint edge. An illustration of this situation is provided in Figure 4.2.

Here too we will need to properly investigate what features are most appropriate. For the target-side comparisons where there is known target metadata to match (i.e. during gap-filling), we will use the same distance features we develop for the source metadata and initially begin with F_1 scores over 1-gram, 2-gram, 3-gram, and 4-gram metadata matches. Additionally, we may find that it is helpful to separate the source and the target distance features concerning metadata so that they can receive different weights. However, these features will not suffice for modeling the combination of two abutting partial translations (with or without reordering). Sequential metadata can be modeled with a statistical language model in the same way we model the lexical sequence. However, we are also interested in modeling non-sequential metadata. For this situation our initial implementation will penalize hypotheses based on the number of incomplete annotations that remain after combination. As before, additional features will be explored as necessary.

After investigating what features to use to model changes in metadata, we plan to expand on the type and quantity of metadata used within the system. As alluded to earlier, the first step will be modeling multiple sources of metadata that are not sequential in nature. Each source of metadata will maintain its own collection of annotations and the metadata distance features

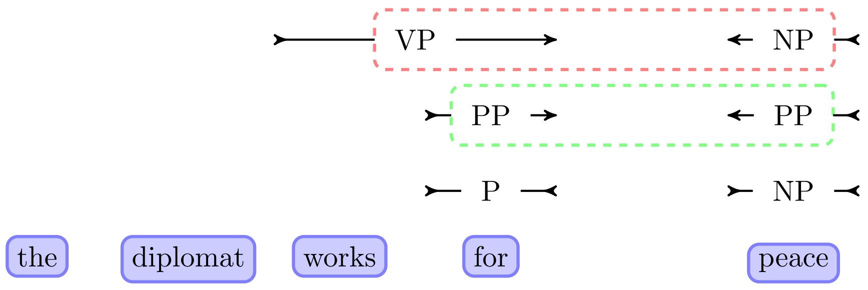


Figure 4.2: Target-language metadata

will be calculated separately over each collection. For example, we might include part-of-speech (sequential) and named-entity annotations (non-sequential) as two different sources of metadata each with their own set of features. Possible forms of non-sequential metadata we are interested in exploring are syntactic trees, co-reference resolution, named entities, subject/object identification, and sentiment detection. Adding additional metadata should be a relatively painless process as we will use off-the-shelf statistical taggers to annotate the corpus. Limited support is currently available for unordered and overlapping metadata spanning multiple tokens. We have started to use name entity tags but have not yet evaluated this setup carefully.

Next, we will extend the mechanism of metadata to permit hierarchical annotations. In particular, to compare this approach with synchronous context-free grammars, we would like to be able to index complete parse trees. It is important to point out that unlike a synchronous context-free grammar, we are not explicitly modeling the generation of the parse tree. Instead, like all of our other metadata the parse trees assist in measuring similarity. The selection of each translation instance and the decoder's combination process will leverage these distance features. Once the capability for hierarchical metadata is in place, it would also be worthwhile to explore the use of dependency trees. Implementation-wise, the key addition is that each annotation will need to be extended to support the concept of a parent node.

SMT has attempted to move beyond purely lexical translation by incorporating factors or synchronous context-free grammars with non-terminal labels. While some improvement has been reported, both of these approaches eventually suffer from fragmenting the hypothesis space. The structure of the model may vary, but the basic idea is that each phrase pair is conditioned on the additional information (factors or a parse tree). As a result, many of these conditional phrase pairs occur only a few times in the corpus and exhibit unreliable relative frequency estimates (which form the basis for many of the features). Smoothing does help, but it makes modeling much more complicated.

In one sense, our approach that models each instance of a translation with varying levels of metadata maximally fragments the translation-space. The crucial difference is that we do not rely on coarse relative frequency estimates. Instead, our model incorporates smooth distance models ϕ_1 and ϕ_3 alongside the assessment of the quality of each translation instance, ϕ_2. Essentially our model provides built-in interpolation as the score for a translation is calculated by summing over the distance scores of all instances that are consistent with a hypothesis. However, because this interpolation is part of the model, it too is learned during optimization. In the same way that instances with the same context as the input contribute more to the translation model, our approach will prefer instances with similar metadata to the input. Instead of fragmenting the translation-space based on each type of metadata, we interpolate knowledge across all forms of metadata. This leads to the additional benefit that, unlike most approaches in SMT, we do not have to specify the structure of the model.

4.1.3 Relations

Thus far we have described several different ways to measure how relevant an instance of a translation is based on its context and additional metadata. However, we have assumed until now that each lexical sequence of words is distinct and should be modeled independently. This naïve assumption significantly simplified our calculations, but it is not accurate. While the collection of examples we extract from the corpus will propose a wide range of possible translations, some of these translations will be only subtly divergent. A subset of the predicted translations will be related in the sense that they share significant semantics, context, or structure. As part of the thesis work, we propose exploiting these relations among the target-side hypotheses. We will model this with the distance function ϕ_3 to measure the similarity of one hypothesis to all the other predicted hypotheses. Instead of simply summing over all instances that have the same target hypothesis, we will sum over all instances weighted by their target hypothesis similarity. The use of target relations, ϕ_3, complements our source-side similarity, ϕ_1, as illustrated in Figure 1.3 of the introduction. The basic framework for this is supported in Cunei, but we have not yet run any experiments with it.

The initial distance features will be what we used for metadata–namely, the F_1 scores of 1-grams, 2-grams, 3-grams, and 4-grams. As we explore additional distance metrics for metadata comparison, many of them should also be suitable for detecting divergences within the lexical type sequence. However, all of these metrics will be type-driven and unable to capture shared semantics between multiple types. To model the semantic relations we prefer methods that are statistical, language independent, and corpus driven. Thus, while a knowledge base such as WordNet may provide great utility for English translations, we will focus our efforts elsewhere. In particular, we are interested in utilizing recent research in paraphrasing.

The idea from paraphrasing that we propose exploring is the use of target-side distributional profiles to predict semantic similarity. Since all of our translations are derived from the same source phrase, we need to be careful how the paraphrasing is modeled. Data-driven methods using the bilingual alignment links will arrive at the conclusion that *all* of our translations are strong paraphrases of each other. Using target-side distributional profiles conveniently avoids knowledge of the shared source alignment. Furthermore, it is relatively efficient to compute within Cunei. When each instance of a translation is retrieved from the corpus, the entire target sentence in which it resides is available. We can use the complete target sentence to form a distributional profile which is then associated with the target hypothesis. To determine whether two target hypotheses are paraphrases of each other, we compute the distance between their distributional profiles. This calculation represents ϕ_3 and will be modeled with a set of distance features. We expect to store the distributional profiles as vectors similar in representation to the source-side context. Thus, we will also start with the same two features, cosine distance and Jensen-Shannon distance, which we described previously and which are known to operate well over vectors.

In addition, once the set of translation instances that match the input are loaded from the corpus, we can model the likelihood that translation instances come from a particular sentence or document in the corpus. The underlying concept is that when forming a translation we would prefer to use a collection of partial translations that occur within the same vicinity of each other in the corpus. Therefore, in measuring ϕ_3, we will also add a distance feature that reflects the density of each position in the corpus with respect to the collection of translation instances retrieved for the current input (as stored in the lattice). We will initially calculate the density per sentence, but as appropriate investigate the merits of incorporating document-level densities.

In both of these cases, we will be updating the translation models after retrieving the instances of translation from the corpus and prior to decoding. Thus, the ϕ_3 distance features are sensitive to the frequency and variety of other translation instances present in the lattice. However, the features are not re-calculated based on which partial translations are actually selected by the decoder and combined together. This is a deliberate decision to make these calculations efficient and limit the scope of our thesis work.

4.2 Language Modeling

All the work described up to this point has been focused on improving machine translation and, in particular, the translation model. However, we believe that this approach is generalizable to other data-driven models and would like to take one step in this direction by applying the same approach to language modeling. Language models have relevance beyond the scope of machine translation, but we can conveniently also explore them within the context of machine translation. Furthermore, while language models share the same domain of natural language, the structure that is modeled is quite different. In particular, language models depend heavily on using simple n-gram relative frequencies and sophisticated interpolation. In recent years we have seen a push in language modeling to use ever larger corpora. The main motivation behind this, and fundamental problem with current language models, is that they do not generalize well. Most 5-gram or 6-gram matches will not occur frequently in a corpus. Reliable statistics can only be gathered in very large corpora and even then the model is quite sensitive to the method of smoothing.

The novelty of our translation model is that it interpolates among translation instances using distance features that assess divergences in context, metadata, and semantics. By incorporating a measure of how similar a particular instance is to what we are attempting to model we can leverage instances that do not exactly represent the input. This same concept can equally be applied to language modeling. One of the perceived benefits is that our smooth distance function will mitigate the quantization error that is normally present in a standard language model's maximum likelihood estimates.

Like a typical n-gram language model we will score w_n given the history $w_1 w_2 w_3 ... w_{n-1}$. Our lazy search will query a large monolingual corpus and retrieve partial matches for the sequence of words. These matches will minimally cover w_n, but may also include some of the history. As in translation modeling, multiple sequence types and forms of metadata may be incorporated into this search. Each partial match represents an instance of an n-gram in the corpus that is weighted by our model according to a set of distance features. Conveniently, many of the same features we have proposed for translation modeling, with the notable exception of alignment features, should be relevant here too. In particular, we expect most of the features that operate over context and metadata as described in §4.1.1 and §4.1.2 to be pertinent. In addition, we will augment the model with 7 new binary features. The first five of these relate to the length of the match. Each instance of an n-gram will have the first n features (or first five features if n is greater than five) set to 1. The sixth feature is set for all n-grams for which n is greater than five and the seventh feature is set for out-of-vocabulary words. The score for w_n is then computed by summing over all instances that contain it. While it is dependent on the weights of the model, generally speaking, the presence of $w_{n-2} w_{n-1} w_n$ will result in a greater score for w_n than the presence of $w_{n-1} w_n$ alone. Fully exploring this concept could be a separate thesis in its own right, so we merely propose to report preliminary results that (hopefully) suggest this is fruitful and should be pursued in future work.

Our instance-based approach does not lend itself neatly to conditional probabilities as any single instance will always report a conditional probability of 1. Thus, we have replaced the conditional probability standard in language modeling with an instance-based density estimator. While we have seen this approach work well in translation modeling, it is entirely possible that it will fail in language modeling. This in and of itself would not undermine our thesis as the primary concern of this work is in regard to translation modeling. Generality to language modeling is desirable, but minimally we will still gain a better understanding of the limits of our approach.

4.3 Summary

As described in the introduction and then investigated in §2, over the years researchers have introduced several different ways to augment and extend the traditional SMT model. Our approach is able to build upon and integrate work in context, paraphrasing, adaptation, and the like into one unified model. By transitioning from modeling a mixture of independent models, to modeling each instance in the corpus, we unlock new potential in machine translation. The significance of our model is that it assesses the relevance of each translation instance from the corpus that is used to produce the complete translation hypothesis with a distance measure. This distance measure allows for easy integration of instance-specific features, and we have outlined our plans to exploit this in three key directions: non-local features (§4.1.1), metadata (§4.1.2), and related translations (§4.1.3). Furthermore, we described in §3.2.1 how our model synthesizes the approaches of SMT and EBMT. Much work remains, but we are hopeful based on what has been completed that this proposal will lead to significant improvements in machine translation quality.

Appendix A

Timeline

2010	
March	Thesis proposal
April	Extend non-local feature experiments
May	Write-up results from non-local features
June-July	Experiment with multiple unsupervised clustering algorithms for flat metadata
August-September	Experiment with dependency graphs to generate hierarchical metadata
October-November	Experiment with parse-trees to generate hierarchical metadata
December	Write-up results from incorporating metadata

2011	
January-February	Identify useful similarity metrics for related phrases
March-April	Experiment with modeling related translations
May	Write-up results from related translations
June-July	Experiment with language modeling
August	Write-up results from language modeling
September	Complete first draft of thesis
October	Respond to feedback/comments
November	Prepare presentation
December	Thesis defense

Appendix B

A Mathematical Description

Recall the standard SMT model with features ϕ and weights λ can be represented as:

$$\underset{\bar{t}}{\operatorname{argmax}} \prod_{\langle s,t\rangle \in \langle \bar{s},\bar{t}\rangle} e^{\lambda\phi(s,t)}$$

The target sentence $\bar{t}$ is the composition of one or more phrase pairs $\langle s,t\rangle$. The set of phrase pairs are constrained in that they must also compose the source sentence $\bar{s}$. Historically, ϕ decomposed according to the source-channel paradigm:

$$\phi(s,t) = \ln P(s,t) = \ln P(s|t) + \ln P(t|t_{i-1}...t_{i-n})$$

However, modern SMT decoders do not worry about this and model ϕ as a log-linear model of many–potentially overlapping–features.

The key distinction of our approach is that instead of modeling generic phrase pairs, we individually model each instance of a translation $\langle s',t'\rangle$ in the corpus. The score for a phrase pair $\langle s,t\rangle$, which is proportional to the probability of the translation $P(s,t)$, is achieved by summing over the model of each instance in the corpus C with respect to the phrase pair being modeled:

$$P(s,t) = \sum_{\langle s',t'\rangle \in C} P(s,s',t',t) \propto \sum_{\langle s',t'\rangle \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)}$$

The instance-specific features $\phi(s,s',t',t)$ can be split into three components:

$$\lambda\phi(s,s',t',t) = \lambda_1\phi_1(s,s') + \lambda_2\phi_2(t',s') + \lambda_3\phi_3(t',t)$$

These components correspond, in order, to the similarity between the source of the input and the source of the instance being modeled, the likelihood of translation, and the similarity between the target of the instance and the target hypothesis begin modeled. The introduction of instance-specific features results in the full translation model being:

$$\underset{\bar{t}}{\operatorname{argmax}} \prod_{\langle s,t\rangle \in \langle \bar{s},\bar{t}\rangle} \sum_{\langle s',t'\rangle \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)}$$

In addition, we introduce γ as a way to (optionally) anneal the distribution during training (for standard decoding $\gamma = 1$):

$$\operatorname*{argmax}_{\bar{t}} \prod_{\langle s,t\rangle \in \langle \bar{s},\bar{t}\rangle} \left(\sum_{\langle s',t'\rangle \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma}$$

Calculating the value of this function is tractable, but calculating the gradient requires maintaining the feature vectors for all instances of all translations and becomes prohibitively costly. Instead we approximate this summation with a log-linear model ψ and an extra variable δ, such that:

$$e^{\gamma(\delta+\sum_i \lambda_i \psi_i)} \approx \left(\sum_{(s',t') \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma}$$

Note that the same parameters λ are used in both models. The goal is that by learning parameters that minimize an objective function for the approximation ψ, the same parameters will minimize an objective function for the 'true' model ϕ. Thus, in addition to resulting in a score as close as possible to the summation, the gradient of the parameters should be approximately equal:

$$\frac{d}{d\lambda_k} e^{\gamma(\delta+\sum_i \lambda_i \psi_i)} \approx \frac{d}{d\lambda_k} \left(\sum_{(s',t') \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma}$$

For feature k, the derivative of the weight λ_k with respect to ψ is:

$$\begin{aligned} \frac{d}{d\lambda_k} e^{\gamma(\delta+\sum_i \lambda_i \psi_i)} &= \left(\frac{d}{d\lambda_k} \gamma \sum_i \lambda_i \psi_i \right) e^{\gamma(\delta+\sum_i \lambda_i \psi_i)} \\ &= \gamma \psi_k e^{\gamma(\delta+\sum_i \lambda_i \psi_i)} \end{aligned}$$

For feature k, the derivative of the weight λ_k with respect to ϕ is:

$$\begin{aligned} \frac{d}{d\lambda_k} \left(\sum_{(s',t') \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma} &= \gamma \left(\sum_{(s',t') \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma-1} \sum_{(s',t') \in C} \frac{d}{d\lambda_k} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \\ &= \gamma \left(\sum_{(s',t') \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma-1} \sum_{(s',t') \in C} \left(\frac{d}{d\lambda_k} \sum_i \lambda_i \phi_i(s,s',t',t) \right) e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \\ &= \gamma \left(\sum_{(s',t') \in C} e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \right)^{\gamma-1} \sum_{(s',t') \in C} \phi_k(s,s',t',t) e^{\sum_i \lambda_i \phi_i(s,s',t',t)} \end{aligned}$$

Setting the the two solutions to the derivative of λ equal:

$$
\begin{aligned}
\gamma\psi_k e^{\gamma(\delta+\sum_i \lambda_i\psi_i)} &= \gamma\left(\sum_{(s',t')\in C} e^{\sum_i \lambda_i\phi_i(s,s',t',t)}\right)^{\gamma-1} \sum_{(s',t')\in C} \phi_k(s,s',t',t)e^{\sum_i \lambda_i\phi_i(s,s',t',t)} \\
\psi_k &= \frac{\left(\sum_{(s',t')\in C} e^{\sum_i \lambda_i\phi_i(s,s',t',t)}\right)^{\gamma-1} \sum_{(s',t')\in C} \phi_k(s,s',t',t)e^{\sum_i \lambda_i\phi_i(s,s',t',t)}}{e^{\gamma(\delta+\sum_i \lambda_i\psi_i)}} \\
&= \frac{\left(\sum_{(s',t')\in C} e^{\sum_i \lambda_i\phi_i(s,s',t',t)}\right)^{\gamma-1} \sum_{(s',t')\in C} \phi_k(s,s',t',t)e^{\sum_i \lambda_i\phi_i(s,s',t',t)}}{\left(\sum_{(s',t')\in C} e^{\sum_i \lambda_i\phi_i(s,s',t',t)}\right)^{\gamma}} \\
&= \frac{\sum_{(s',t')\in C} \phi_k(s,s',t',t)e^{\sum_i \lambda_i\phi_i(s,s',t',t)}}{\sum_{(s',t')\in C} e^{\sum_i \lambda_i\phi_i(s,s',t',t)}}
\end{aligned}
$$

Essentially by setting each feature ψ to the expectation of ϕ according to the probability distribution of each translation instance, the gradient of λ under both models is equivalent. This calculation is dependent on the parameters λ and, therefore, only valid for that particular point in space, but should form a reasonable approximation as long as the parameters do not drastically change. The variable δ, hereto unconstrained, adjusts for any discrepancies between the scores of the two models. Finally, solving for δ:

$$
\begin{aligned}
e^{\gamma(\delta+\sum_k \lambda_k\psi_k)} &= \left(\sum_{(s',t')\in C} e^{\sum_k \lambda_k\phi_k(s,s',t',t)}\right)^{\gamma} \\
e^{\gamma\delta}e^{\gamma\sum_k \lambda_k\psi_k} &= \left(\sum_{(s',t')\in C} e^{\sum_k \lambda_k\phi_k(s,s',t',t)}\right)^{\gamma} \\
e^{\gamma\delta} &= \frac{\left(\sum_{(s',t')\in C} e^{\sum_k \lambda_k\phi_k(s,s',t',t)}\right)^{\gamma}}{e^{\gamma\sum_k \lambda_k\psi_k}} \\
\gamma\delta &= \gamma\ln\left(\sum_{(s',t')\in C} e^{\sum_k \lambda_k\phi_k(s,s',t',t)}\right) - \gamma\sum_k \lambda_k\psi_k \\
\delta &= \ln\left(\sum_{(s',t')\in C} e^{\sum_k \lambda_k\phi_k(s,s',t',t)}\right) - \sum_k \lambda_k\psi_k \\
&= \ln\left(\sum_{(s',t')\in C} e^{\sum_k \lambda_k\phi_k(s,s',t',t)}\right) - \frac{\sum_k \lambda_k \sum_{(s',t')\in C} \phi_k(s,s',t',t)e^{\sum_i \lambda_i\phi_i(s,s',t',t)}}{\sum_{(s',t')\in C} e^{\sum_i \lambda_i\phi_i(s,s',t',t)}}
\end{aligned}
$$

Appendix C

Bibliography

Bannard, C. and Callison-Burch, C. (2005). Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 597–604, Ann Arbor, USA.

Berger, A. L., Pietra, S. A. D., and Pietra, V. J. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Blunsom, P., Cohn, T., Dyer, C., and Osborne, M. (2009). A gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 782–790, Suntec, Singapore.

Brown, P. E., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Brown, R. D. (1996). Example-based machine translation in the pangloss system. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 169–174, Copenhagen, Denmark.

Brown, R. D. (2004). A modified burrows-wheeler transform for highly scalable example-based translation. In Frederking, R. E. and Taylor, K., editors, *Machine Translation: From Real Users to Research, 6th Conference of the Association for Machine Translation in the Americas*, pages 27–36, Washington, DC, USA. Springer.

Brown, R. D., Hutchinson, R., Bennett, P. N., Carbonell, J. G., and Jansen, P. (2003). Reducing boundary friction using translation-fragment overlap. In *Machine Translation Summit IX Proceedings*, pages 24–31, New Orleans, USA.

Callison-Burch, C., Bannard, C., and Schroeder, J. (2005). Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 255–262, Ann Arbor, USA.

Callison-Burch, C., Koehn, P., and Osborne, M. (2006). Improved statistical machine translation using paraphrases. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 17–24, New York City, USA.

Carbonell, J., Klein, S., Miller, D., Steinbaum, M., Grassiany, T., and Frei, J. (2004). Context-based machine translation. In Frederking, R. E. and Taylor, K., editors, *Machine Translation: From Real Users to Research, 6th Conference of the Association for Machine Translation in the Americas*, pages 19–28, Washington, DC, USA. Springer.

Carpuat, M. and Wu, D. (2005). Word sense disambiguation vs. statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 387–394, Ann Arbor, USA.

Carpuat, M. and Wu, D. (2007). Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 61–72, Prague, Czech Republic.

Chan, Y. S., Ng, H. T., and Chiang, D. (2007). Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 33–40, Prague, Czech Republic.

Cherry, C. and Lin, D. (2007). Inversion transduction grammar for joint phrasal translation modeling. In *Proceedings of the NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 17–24, Rochester, NY, USA.

Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 218–226, Boulder, USA.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Gimpel, K. and Smith, N. A. (2008). Rich source-side context for statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 9–17, Columbus, USA.

Hildebrand, A. S., Eck, M., Vogel, S., and Waibel, A. (2005). Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the Tenth Annual Conference of the European Assocation for Machine Translation*, pages 133–142, Budapest, Hungary.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Machine Translation Summit X Proceedings*, pages 79–86, Phuket, Thailand.

Koehn, P., Axelrod, A., Mayne, R. B., Callison-Burch, C., Osborne, M., and Talbot, D. (2005). Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*, Pittsburgh, USA.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133, Edmonton, Canada.

Kumar, S. and Byrne, W. (2004). Minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 169–176, Boston, USA.

Liang, P., Taskar, B., and Klein, D. (2006). Alignment by agreement. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 104–111, New York City, USA.

Lin, J. (1991). Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.

Lopez, A. (2007). Hierarchical phrase-based translation with suffix arrays. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 976–985, Prague, Czech Republic.

Lopez, A. (2008). Tera-scale translation models via pattern matching. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 505–512, Manchester, United Kingdom.

Manber, U. and Myers, G. (1990). Suffix arrays: a new method for on-line string searches. In *SODA '90: Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pages 319–327, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Marcu, D. and Wong, W. (2002). A phrase-based, joint probability model for statistical machine translation. In *2002 Conference on Empirical Methods in Natural Language Processing*, pages 133–139, Philadelphia, PA, USA.

Marton, Y., Callison-Burch, C., and Resnik, P. (2009). Improved statistical machine translation using monolingually-derived paraphrases. In *2009 Conference on Empirical Methods in Natural Language Processing*, pages 381–390, Suntec, Singapore.

Matsoukas, S., Rosti, A.-V. I., and Zhang, B. (2009). Discriminative corpus weight estimation for machine translation. In *2009 Conference on Empirical Methods in Natural Language Processing*, pages 708–717, Suntec, Singapore.

Nakazawa, T., Yu, K., Kawahara, D., and Kurohashi, S. (2006). Example-based machine translation based on deeper nlp. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 64–70, Kyoto, Japan.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.

Och, F. J. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Philadelphia, USA.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, USA.

Phillips, A. B. (2007). Sub-phrasal matching and structural templates in example-based mt. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 163–170, Skövde, Sweden.

Phillips, A. B. and Brown, R. D. (2009). Cunei machine translation platform: System description. In Forcada, M. L. and Way, A., editors, *Proceedings of the 3rd Workshop on Example-Based Machine Translation*, pages 29–36, Dublin, Ireland.

Sarikaya, R., Deng, Y., Afify, M., Kingsbury, B., and Gao, Y. (2008). Machine translation in continuous space. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association*, pages 2350–2353, Brisbane, Australia.

Schwenk, H., Dchelotte, D., and Gauvain, J.-L. (2006). Continuous space language models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 723–730, Sydney, Australia.

Shen, L., Xu, J., Zhang, B., Matsoukas, S., and Weischedel, R. (2009). Effective use of linguistic and contextual information for statistical machine translation. In *2009 Conference on Empirical Methods in Natural Language Processing*, pages 72–80, Suntec, Singapore.

Smith, D. A. and Eisner, J. (2006). Minimum risk annealing for training log-linear models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 787–794, Sydney, Australia.

Vogel, S. (2005). Pesa: Phrase pair extraction as sentence splitting. In *Machine Translation Summit X Proceedings*, pages 251–258, Phuket, Thailand.

Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. (2007). Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 764–773, Prague, Czech Republic.

Way, A. (2003). Machine translation using lfg-dop. In Bod, R., Scha, R., , and Sima'an, K., editors, *Data-Oriented Parsing*, pages 359–384. CSLI.

Yamamoto, M. and Church, K. W. (2001). Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. *Computational Linguistics*, 27(1):1–30.

Zhang, Y. and Vogel, S. (2005). An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora. In *Proceedings of the Tenth Annual Conference of the European Assocation for Machine Translation*, pages 294–301, Budapest, Hungary.