Modeling Relevance in Statistical Machine Translation: Scoring Alignment, Context, and Annotations of Translation Instances

Aaron B. Phillips

CMU-LTI-12-004

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee

Ralf D. Brown (Chair)
Stephan Vogel
Noah A. Smith
Chris Callison-Burch

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Language and Information Technologies

Abstract

Machine translation has advanced considerably in recent years, primarily due to the availability of larger datasets. However, one cannot rely on the availability of copious, high-quality bilingual training data. In this work, we improve upon the state-of-the-art in machine translation with an instance-based model that scores each instance of translation in the corpus. A translation instance reflects a source and target correspondence at *one specific location in the corpus.* The significance of this approach is that our model is able to capture that some instances of translation are more relevant than others.

We have implemented this approach in Cunei, a new platform for machine translation that permits the scoring of instance-specific features. Leveraging per-instance alignment features, we demonstrate that Cunei can outperform Moses, a widely-used machine translation system. We then expand on this baseline system in three principal directions, each of which shows further gains. First, we score the source context of a translation instance in order to favor those that are most similar to the input sentence. Second, we apply similar techniques to score the target context of a translation instance and favor those that are most similar to the target hypothesis. Third, we provide a mechanism to mark-up the corpus with annotations (e.g. statistical word clustering, part-of-speech labels, and parse trees) and then exploit this information to create additional per-instance similarity features. Each of these techniques explicitly takes advantage of the fact that our approach scores each instance of translation on demand after the input sentence is provided and while the target hypothesis is being generated; similar extensions would be impossible or quite difficult in existing machine translation systems.

Ultimately, this approach provides a more flexible framework for integration of novel features that adapts better to new data. In our experiments with German-English and Czech-English translation, the addition of instance-specific features consistently shows improvement.

Acknowledgments

I am indebted to the community of researchers at Carnegie Mellon University. Foremost, I thank my thesis committee for your counsel and guidance. You pushed me to think more carefully and to write more clearly; this dissertation is stronger because of you. I thank my professors for imparting knowledge and understanding. You provided the foundation in linguistics, statistics, and machine translation that was instrumental to this work. And to my fellow Ph.D students, you too have provided clarity and insights, but you have also made life fun. Several of you have preceded me in graduating, but to those who remain: best wishes! To all–my committee, professors, and fellow students, you have made the Language Technologies Institute a nurturing and rewarding environment; I consider myself fortunate to have been part of this community.

I am also indebted to my family for their unconditional support. To my wife, Lindsay, I thank you for your grace and commitment through the last seven years. You have also provided me with the crucial technical expertise of where to place commas; somehow that grammar lesson has escaped me through all my years of education. And to my daughter, Evangeline Raine: in the last few months I have spent many more long days at the office writing than I care to acknowledge, but returning home to your smiles has always warmed my heart.

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 datasets. When provided ample, in-domain bilingual training data, state-of-the-art machine translation can be quite good. However, data collection is still costly and time-consuming. For many language pairs, we have at most moderate amounts of training data. Much of the available data is collected from peculiar genres, such as legislation, that is easier to obtain but often at a disconnect from the type of documents one wants to translate. Furthermore, within these collections of text, the quality of translation is not consistent and can vary substantially. These issues pose challenges to the standard methods for data-driven machine translation. We improve upon the state-of-the-art in machine translation with an instance-based approach that selects relevant, high-quality translations.

1.1 An Instance-Based Approach to Machine Translation

General purpose, fully-automated, high-quality machine translation necessitates modeling the translation task as a decomposable process. Our approach employs the same framework as current state-of-the art machine translation systems that score smaller fragments of translation known as translation units which are then combined together to generate a target hypothesis. Within this framework, we propose scoring each translation unit based on its similarity to relevant instances of a translation in the training data. Each instance of translation–the occurrence of a translation at one specific location in the corpus–occurs within a unique linguistic context. A reasonably large corpus will contain many such instances of translation, but they are not all equally suitable for the translation task at hand. For example, each translation instance–even if it predicts the same target hypothesis–may be associated with a different parse tree or morpho-syntactic information. What makes our approach unique is that it is able to score this information (by comparing it to the input sentence or target hypothesis) individually for each instance of translation. We exploit this per-instance information to distinguish when some translation instances are more relevant than others. The goal is to generate a target hypothesis from a combination of translation units that is maximally similar to instances of translation present in a training corpus.

Our method for scoring each translation unit from instances of translation is a form of instance-based learning. To construct a new translation unit, the system searches the training data for one or more relevant training instances. The 'nearest neighbors' are identified with a distance function that scores the relative importance of each translation instance. Possible translation units are then scored by summing over the score the model assigns to translation instances (i.e. 'neighbors'). Instance-based learning will be discussed in more detail in Chapter 2, but an advantage, in general,

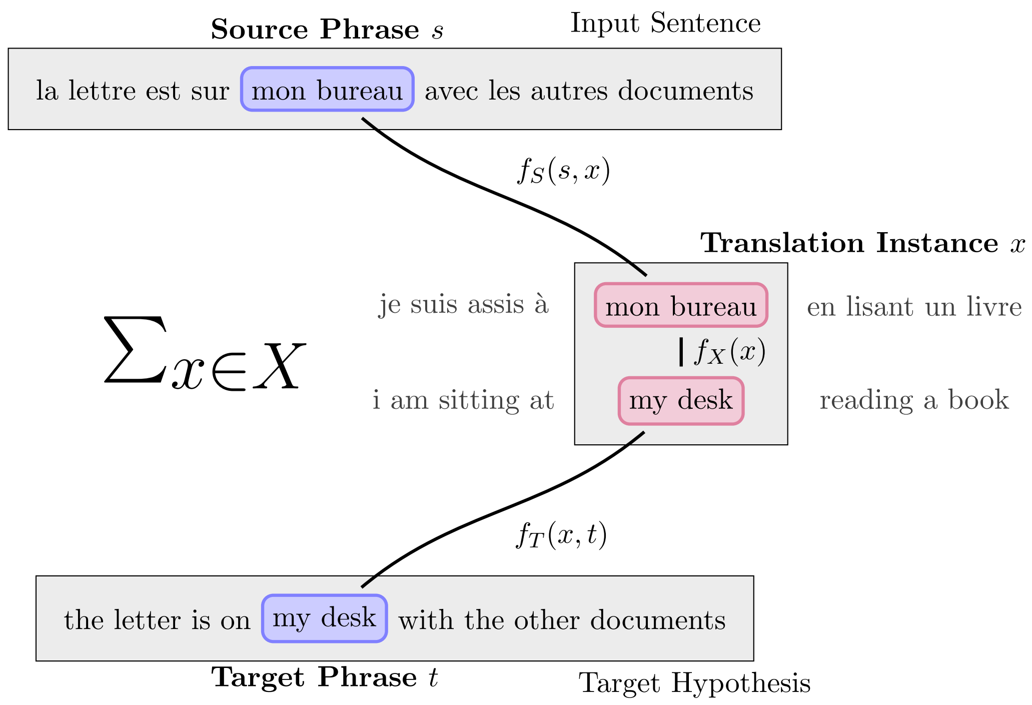


Figure 1.1: Scoring an Instance of Translation

of instance-based learning is that it inherently allows for generalization. If a highly-specialized instance of translation exists in the corpus that exactly matches the context of the text being translated, it will be leveraged by the model. Likewise, if little information exists in the training data and the most similar instance of translation is only a paraphrase, that too will be scored and leveraged by the model.

The formalism and implementation of our model will be presented in Chapter 3. Our implementation uses phrase pairs as the translation units. Each phrase pair represents a source phrase and a target phrase that are presumed to be translations of each other. Our instance-based approach could be applied to other types of translation units, but phrase pairs are simple and intuitive. We have illustrated the process for scoring one instance of translation in Figure 1.1. Here the instance of translation is used to score the correspondence between a source phrase s and a target phrase t within the broader context of the input sentence and the target hypothesis. In this case, the instance of translation x has the same source and target as the phrase pair $\langle s, t \rangle$, but this is not required. The instance of translation is scored with a parametric distance function defined with multiple feature functions. In the illustration, we present this distance function conceptually as three functions: $f_S(s,x)$, $f_X(x)$, and $f_T(x,t)$. The similarity functions $f_S(s,x)$ and $f_T(x,t)$ measure the distance between what one desires to model and what information is available in the corpus. $f_S(s,x)$ calculates the distance between the input sentence and the source of the translation instance while $f_T(x,t)$ measures the distance between the target of the instance and the target hypothesis. Finally, this breakdown provides us the ability to score the translation with $f_N(x)$ based on its location, and thereby surrounding context, in the corpus. For example, one is permitted to generate features that use the document in which the instance is located or that analyze the alignment probability of an instance with respect to the effect of the phrase alignment on the remainder of the sentence. These functions assign preference to high-quality instances of translation that are minimally divergent from the input sentence and the target hypothesis. But, when data is scarce,

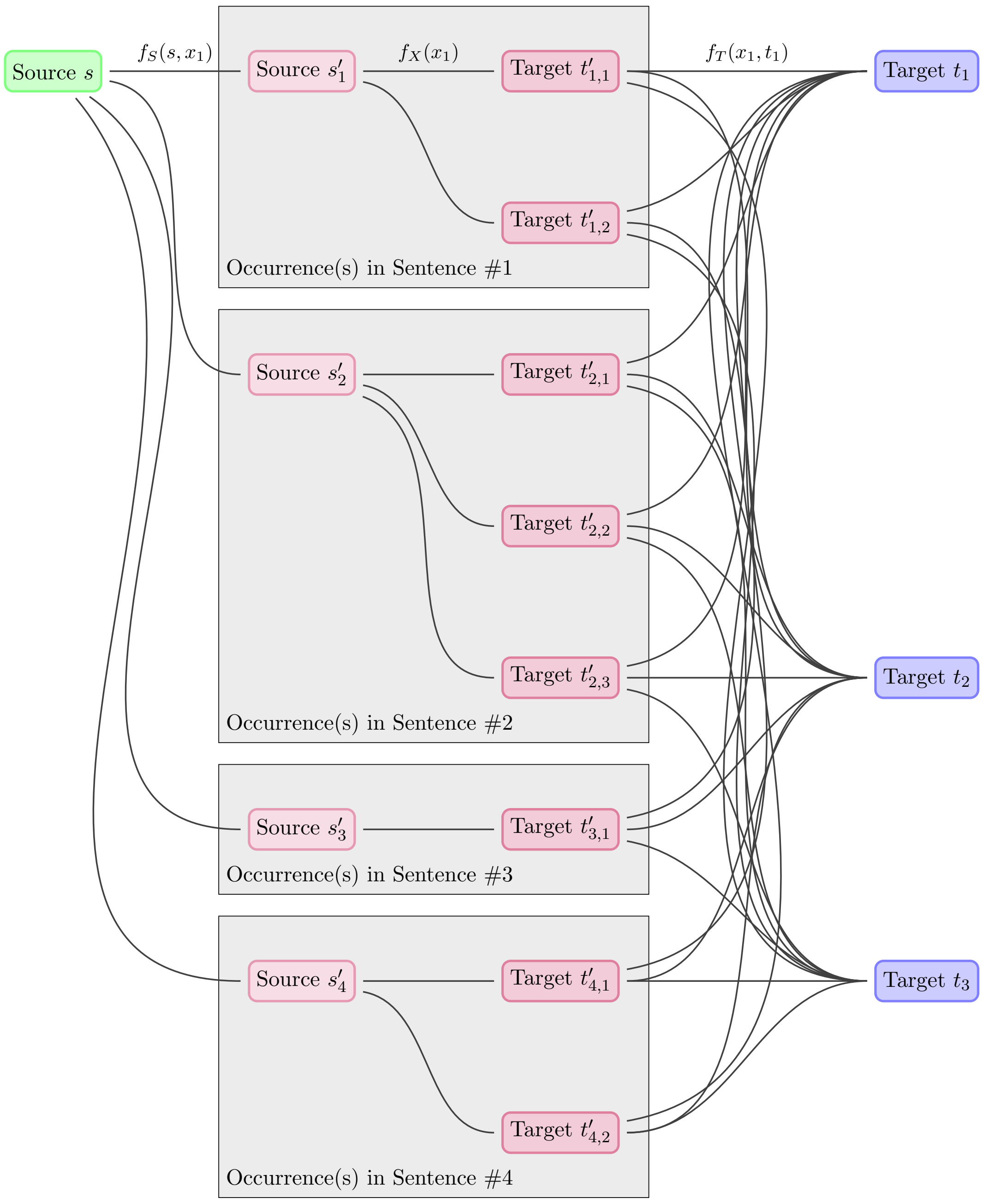


Figure 1.2: Scoring Translation Units with Instances of Translation

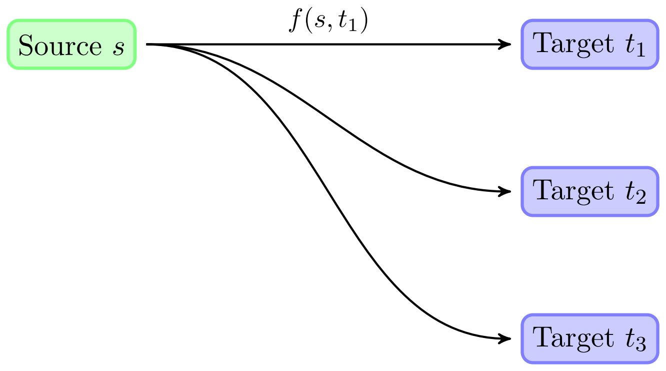


Figure 1.3: The Standard Statistical Translation Model

this framework allows the model to capture a broader perspective of information from the corpus.

The score for a phrase pair is the summation over all instances of translation as shown in Figure 1.2.[1] In this illustration, the occurrences of a source phrase s' in one sentence can correspond to multiple target phrases t' due to uncertainty in the alignment. Our model considers each pair $\langle s', t' \rangle$ at one location in the corpus as a unique instance of translation x. *All* instances of translation are used by the model, albeit some instances of translation contribute more to the overall score for a phrase pair. The greater the score for a translation instance, the more confident the model is that it represents a valid translation. We acknowledge that instance-based modeling is more complex; while at one point this approach may not have been practical, today it is, thanks to Moore's law.

Ultimately, this approach provides a more flexible framework for integration of novel features that adapts better to new data. To substantiate this claim, we propose injecting novel sources of information that exploit instance-specific features. Therefore, our thesis is that

> **Thesis Statement**: Scoring each instance of a translation in the corpus will improve machine translation quality and facilitate the integration of non-local context and similarity features.

1.2 Statistical Machine Translation

Statistical machine translation (SMT) models translation units as discrete, countable events. Depending on the architecture, these translation units take on different forms: phrase pairs, grammar rules, or factors. The accepted generative story in statistical machine translation describes a bilingual corpus as being formed by a combination of those translation units. There exists a 'true' distribution for each translation unit; the training corpus provides observations that one uses to estimate this distribution. Each occurrence of a translation unit in the training corpus–what we call an instance of translation–is a countable event. The standard SMT model scores each translation unit with a combination of several feature functions. The model is constructed by processing the training corpus and estimating the value for these feature functions based on the occurrences of translation units. Generally all events (occurrences of a translation unit) in the training corpus are considered to be of equal utility. The most common features are relative frequencies of conditional events, such as the likelihood of a source phrase given a target phrase. As our implementation

[1] It is conceptually simpler to describe the collection of phrase pairs as distinct from the instances of translation. However, the model cannot create a target hypothesis that does not occur in the corpus. Therefore, the set of possible phrase pairs is determined by the translation instances found in the corpus.

uses phrase pairs, when we refer to the "SMT model" throughout this dissertation we are, in fact, referring to phrase-based SMT.

SMT has some nice properties: it is easy to extend the model with more features, and the feature calculations are usually simple and scalable to large amounts of data. Given enough data, one can even leverage very fine-grained, specialized phrase pairs. Indeed, as available bilingual datasets have grown, so too has performance of these systems.

In comparison to Figure 1.2, a traditional SMT model scores the correspondence between a source phrase and target phrase *in abstract* as shown in Figure 1.3. The model, consisting of the features θ for each phrase pair, is estimated by processing the training data once. Once the phrase pairs are identified and their corresponding features are calculated, the training corpus can be discarded. The phrase pairs for the model can be highly specialized, but they represent abstract units in that no specific instances of translation are retained. Only the information encoded initially within a phrase pair is available during translation. Likewise, the features may be extensive, but they are calculated without knowledge of what dimensions of the data are most relevant for the input document and target hypothesis are not yet known.

1.3 Thesis Contributions

The fundamental advantage of our model is that it provides a simple and flexible mechanism for integrating new sources of information into machine translation with instance-based features. In the traditional SMT model, a translation unit is an abstraction over many translation instances. However, a wealth of information–such as domain, alignment, context, or parse trees–is all specified on *each occurrence* of a translation unit. This information is not constant and varies between instances of translation in the corpus–even those that predict the same target hypothesis. In Chapter 2 we will walk through several different ways that traditional SMT models have been extended to incorporate individual aspects of this information, but there is no simple, straightforward method for integrating per-instance information. The system designer has to make explicit decisions of how to model the additional information and what features to use. An illustrative example is the paper by (Shen et al., 2009) which states clearly that their unique contribution is that "feature functions defined in this way are robust and ideal for practical translation tasks." The types of linguistic and contextual information their features capture is not new, but as they demonstrate, other approaches have been imperfect and have struggled to incorporate it. In a complex system like machine translation, we do not always know *a priori* what information is useful and how it should be expressed; we would prefer if the model took care of this automatically.

Our model identifies the most relevant instances of translation with a simple distance function consisting of multiple features. As a result, the model is capable of discriminating between individual instances of translation. The traditional SMT model can also be highly-discriminative by incorporating numerous, specialized features. In the SMT model, these features must individually identify each complex event or phenomena the system builder believes to be useful. This can quickly expand to a very, very large number of features; generally, only a select number of complex events are modeled. In addition, these specialized features, by definition, identify complex conditioning events that occur rarely in the training data. In the traditional SMT model, this necessitates the use of complex smoothing techniques to properly estimate the discriminative features. In our instance-based approach, each dimension of the data can be represented as a single feature in the distance function. Furthermore, instance-based learning inherently provides a smooth estimate by automatically adapting to the available data. With our approach, new sources of information can easily be added to discriminate between instances of translation, and the system engineer does not have to determine which complex conditioning events to model.

Our instance-based approach also inherently provides generalization capabilities. The SMT model is parametric; there is a fixed distribution specified by the parameters that the data must fit. In our approach, the complexity of the model will grow with the amount of training data. The more training data, the more often our model will be able to leverage highly-specialized instances of translation. The SMT model can also leverage highly-specialized phrase pairs, but the granularity of these must be decided on *a priori*. Selecting appropriate phrase pairs is a delicate balancing act. With too little data, broader, more generic phrase pairs must be used or the model will overfit. Selecting the 'right' phrase pairs is somewhat of a black art and is often dependent both on the amount of data and the language pair. In our approach, the distance function provides an ordering over all translation instances from most general to most specific. This allows us to leverage the most specific instances of translation where available and automatically use more generic instances of translation as needed.

A common theme here is that there are many modeling decision our model postpones and does not set *a priori*. This is because, like all instance-based methods, our model is not constructed until the input is provided. Traditional SMT systems perform a massive, upfront estimation of their model by processing the training data. Our approach avoids this and scores each phrase pair *on demand*. This trade-off does come at the cost of more computation during translation.

Finally, we find instance-based learning to be an attractive paradigm because it also provides explanative capabilities. When the system produces a target hypothesis, it is the combination of several phrase pairs. For each of these phrase pairs, the system can identify the most relevant translation instances. Our approach still selects the target hypothesis with the best score according to the statistical model, but the calculation for this score is tied to specific instances of translation in the corpus that can be analyzed by a human.

1.4 Roadmap

The early chapters provide background and describe our approach in detail. We also present an implementation of our model that we show to improve upon state-of-the-art SMT.

Chapter 2: Prior and Related Work
We present a more detailed description of instance-based learning and approaches to machine translation. We then survey prior extensions to the traditional SMT model and discuss how they relate to our approach.

Chapter 3: Overview of the Cunei Machine Translation Platform
We formally describe our model and its implementation.

Chapter 4: Learning Model Weights
We describe a method for learning the weights of our model such that hypotheses generated by the model for a development set maximize an objective function.

Chapter 5: Baseline Evaluation
We describe a Czech-English and a German-English dataset that we use to compare our implementation to Moses, a widely-used SMT implementation. We demonstrate that the Cunei baseline configuration, which includes per-instance alignment features, outperforms Moses trained on the same data.

The latter chapters extend our model in three principal directions incorporating new sources of information:

Chapter 6: Incorporating Source Similarity
We introduce non-local features that identify the relevant source context of a translation instance in order to favor those that are most similar to the input sentence.

Chapter 7: Incorporating Target Similarity
We apply some of the same techniques from the previous chapter to score the target context of a translation instance and favor those that are most similar to the target hypothesis.

Chapter 8: Incorporating Corpus Annotations
We provide a mechanism to mark-up the corpus with annotations from multiple external sources (statistical word clustering, part-of-speech labels, and parse trees) and then score the similarity of translation instances with respect to these annotations.

Each of these techniques explicitly takes advantage of the fact that our approach scores each instance of translation on demand after the input sentence is provided and while the target hypothesis is being generated; similar extensions would be impossible or quite difficult with a traditional SMT model. We conclude with Chapter 9, which presents a summary of our findings and discusses possible future directions.

Chapter 2

Prior and Related Work

Machine translation is an enormously complex task that has spawned an entire field of research with its own approaches, algorithms, and techniques. We will focus our survey of prior and related work on core research that directly applies to machine translation. Within this discussion we will address the broader field of natural language processing and related areas such as language modeling and machine learning where appropriate. Central to this dissertation is the concept of instance-based modeling, but nearly all research in machine translation relies on parametric models. As a result, before diving into the field of machine translation, we will begin the discussion by exploring instance-based learning.

2.1 Instance-Based Learning

Instance-based learning is a non-parametric method for machine learning that performs classification, regression, or density estimation by comparing an input query to known, labeled data. This method is also referenced in the literature as similarity-based, example-based, memory-based, exemplar-based, case-based, analogical, nearest neighbor, and lazy learning (Daelemans et al., 1996). It is referred to as instance-based because this technique constructs hypotheses directly from the training instances themselves and the complexity of the hypothesis grows with the training data (Russell and Norvig, 2003). By definition, it is a *lazy* approach in that the training data is not processed until there is an input query. When an input query is presented to the system, the training data is searched for one or more relevant training instances which are used to construct a hypothesis.

2.1.1 The Distance Function

A core ingredient of instance-based learning is the distance function which is used to identify relevant training instances. The distance function $d(q, x)$ scores how 'far away' the query q is from each instance in the training data x. The data is usually represented as containing multiple dimensions (or features) and we measure the distance along each of these. Suppose there are n such dimensions and let the function $a_i(x)$ represent the value of instance x in dimension i. The Euclidean distance is both simple and frequently used:

$$d(q, x) = \sqrt{\sum_{i=0}^{n} (a_i(q) - a_i(x))^2}$$

Common alternatives are the Mahalanobis distance (Mahalanobis, 1936), which integrates covariance and the Hamming distance (Hamming, 1950), which can be applied to symbolic representations.

A more flexible option is to use a parameterized distance function with feature weights:

$$d(q,x) = \sum_{i=0}^{n} \lambda_i g_i(q,x)$$

When we present the formalism for our model in Chapter 3, we will show that our approach employs a parametric distance function. Parametric distance functions abstract the distance calculation in each dimension i to a function $g_i(x,q)$. We can score each $g_i(q,x)$ with the Hamming distance, Euclidean distance, or something entirely different. In fact, we write $g_i(q,x)$ instead of $g_i(a_i(q), a_i(x))$ as the function could compute a score that simultaneously assesses multiple dimensions of the data. This allows the distance function to score many different representations of the data and weight each of these functions with a parameter λ_i. Daelemans and van den Bosch (1992) describe using the above parametric distance function where λ is set by calculating information gain. In the field of machine translation, there has been much effort in learning the weights of parametric models to maximize an objective function that measures the end-to-end quality of translation. These existing methods for learning λ for a SMT model will be discussed in Section 2.4, and later in Chapter 4 we will apply similar techniques to learn λ for our distance function.

2.1.2 Nearest Neighbors and Kernel Density Estimation

Perhaps the simplest and most well-known form of instance-based learning is the nearest neighbor algorithm first introduced in a technical report by Fix (1951). In this algorithm, an input query is classified by selecting the label from the instance in the training data with the shortest distance (i.e. most similar) to the input. Similarly, this approach can be generalized to k-nearest neighbors in which a majority vote of the k most-relevant training instances $x_1, x_2, x_3 ... x_k$ is used to select the label:

$$\hat{f}(q) = \operatorname*{argmax}_{v \in V} \sum_{i=1}^{k} \delta(v, f(x_i))$$

The function $f(x_i)$ returns the label associated with the training instance x_i. The delta function returns one when the labels v and $f(x_i)$ are equal and zero otherwise. Alternatively, if the data is real-valued, regression can be computed with:

$$\hat{f}(q) = \frac{1}{k} \sum_{i=1}^{k} f(x_i)$$

The distance function is used to select the k-nearest neighbors, but the value of the distance function is not used in the above equations. Each of the k-nearest neighbors is considered equally relevant. As a result, the hypotheses may vary substantially based on the size of k. This behavior is especially evident at the extremes. If k=1, we may be fitting an outlier. If k encompasses the entire training data, the hypothesis will be the same for all queries. Shepard's method improves on the standard k-nearest neighbors algorithm by weighting each instance in the training data by the inverse of its distance to the input query (Shepard, 1968):

$$\hat{f}(q) = \frac{\sum_{i=1}^{k} \frac{1}{d(q,x_i)} f(x_i)}{\sum_{j=1}^{k} \frac{1}{d(q,x_j)}}$$

In this algorithm, training instances far away from the query have little impact on the hypothesis. The value of k can be conveniently increased to the size of the entire dataset, in which case regression yields a continuous function.

Similarly, one can use training instances to perform density estimation. Modeling the probability density with weighted training instances can be generalized as a kernel model. This technique views each training instance as generating a density function of its own that is defined by the kernel function $K(q, x_i)$ (Russell and Norvig, 2003):

$$P(q) = \frac{1}{k}\sum_{i=1}^{k} K(q, x_i)$$

$K(q, x_i)$ is a probability density function that is evaluated at distance $d(q, x_i)$. The kernel function is usually defined as a Gaussian, but one may select other probability density functions. For example, Shepard's method can be obtained by defining the kernel as:

$$K(q, x_i) = \frac{\frac{1}{d(q,x_i)}}{\sum_{j=1}^{k} \frac{1}{d(q,x_j)}}$$

2.1.3 Non-parametric vs Parametric Learning

Most learning methods use parametric models that attempt to fit all the training data to a global set of parameters. Instance-based learning is non-parametric; it uses relevant training instances (or 'neighbors') to build a local model for each query. A key advantage of non-parametric learning is that there is no fixed type of distribution that the data must fit. In fact, the complexity of the model grows with the data. In general, parametric learning can produce better estimates if the parametric assumptions concerning the distribution of the data are correct. On the other hand, non-parametric learning is usually superior when the distribution of the data is unknown or varied.

In non-parametric learning, the entire set of training instances must always be available. A naïve implementation of non-parametric learning requires both space and time complexity of $O(NF)$ where N is the number of training instances and F is the number of features associated with each training instance. However, instance-based learning can commence as soon as training instances are available, and the set of training instances can be modified at any time. This is in contrast to parametric learning which must first process the training data to learn a set of parameters. Initially learning the parameters in a parametric model can be quite expensive (the actual cost varies based on the learning algorithm). However, parametric models are usually quite efficient once the model has been built. For example, a linear model will have space and time complexity of just $O(F)$.

Zavrel and Daelemans (1997) explore the relationship between instance-based learning and back-off smoothing with parametric models. A common problem when learning parametric models is that many possible conditioning events are not present in the training data. Smoothing methods address this issue by redistributing some of the probability mass in a distribution to unseen events. Back-off smoothing is one such technique that linearly interpolates the probabilities from multiple conditioning events. For example, consider a 3-gram language model that estimates the probability of word w_i given the history $w_{i-3}w_{i-2}w_{i-1}$ as:

$$\hat{P}(w_i|w_{i-3}, w_{i-2}, w_{i-1}) = \alpha_1 P(w_i|w_{i-3}, w_{i-2}, w_{i-1}) + \alpha_2 P(w_i|w_{i-2}, w_{i-1}) + \alpha_3 P(w_i|w_{i-1})$$

The weights α define the back-off sequence and the relative importance of each conditioning event. The similarity to instance-based learning is that both techniques use estimates from more general patterns when specific patterns are absent in the training data. In instance-based learning, a distant training instance will be used if no nearby training instance is available. The distance function defines the back-off sequence; it provides an ordering over all training instances from most general to most specific. An advantage of instance-based learning is that it requires only one parameter in the distance function per dimension of the data whereas a back-off model may require an exponential number of parameters to capture combinations of conditioning events.

In the words of Daelemans et al. (1996), instance-based learning "is a statistical approach, but it uses a different kind of statistics than Markov model-based approaches." Instance-based learning constructs a hypothesis by selecting training instances that are most similar or relevant to a query. It is distribution free and inherently provides generalization capabilities when the query does not match instances in the training data. Furthermore, instance-based learning also provides explanation capabilities in that it can identify the training instances that were most informative in generating the hypothesis.

2.2 Paradigms of Machine Translation

In the next chapter we will detail the implementation of Cunei, our instance-based system for machine translation. Before plunging into the technical details, it is important to understand the broader context of machine translation that has shaped this research. Machine translation has grown into a diverse field and we do not attempt to cover all varieties of it (such as rule-based or knowledge-based methods). Our focus here will be limited to data-driven methods, as these are currently the dominant paradigm and they directly relate to our approach.

2.2.1 Statistical Machine Translation

The advent of Statistical Machine Translation (SMT) defined a probabilistic approach to machine translation that was not quickly adopted by researchers, but it has since become the dominant paradigm. 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 maximize an objective function.

Warren Weaver in a 1949 memorandum first suggested that a computer might be capable of performing translation between two natural languages (Locke and Booth, 1955). He even postulated that it could be fruitful to apply cryptographic methods based on Claude Shannon's developing work in information theory. While Weaver is credited with inspiring research in the field of machine translation, it took forty years for his ideas to be fully realized. Based on Shannon's noisy channel model (Shannon, 1948), researchers at IBM's T. J. Watson Research Center introduced the first statistical machine translation system in 1990 (Brown et al., 1990). They described translation as the process of attempting to recover the input sentence s which was transmitted through a noisy channel from which one observed the altered output t. More simply, this work modeled the likelihood that the original source sentence was s when a translator produced the target sentence t. Applying Bayes' rule, this likelihood $P(s|t)$ can be decomposed into:

$$P(s|t) = \frac{P(s)P(t|s)}{P(t)}$$

For translation we are only concerned with selecting the most likely value of s. The decision rule can, thus, ignore the denominator $P(t)$ because it is independent of s:

$$\operatorname*{argmax}_s P(t|s)P(s)$$

In this initial work, Brown et al. (1990) modeled the likelihood of recovering the source sentence given a target sentence, but the same principle can be applied in reverse. It is now more common to refer to translation as selecting the most likely target sentence given a source sentence:

$$\operatorname*{argmax}_t P(s|t)P(t)$$

In this equation, $P(s|t)$ represents the translation model and $P(t)$ represents the language model. Thus, the goal is to select a sentence t that is both fluent and corresponds to the source sentence s.

The preferred model for statistical machine translation is now the log-linear model as introduced to SMT by Berger et al. (1996) and popularized by Och and Ney (2002). This approach throws away the *a priori* structural dependencies of the generative model and models $P(t|s)$ with a set of n features θ and weights λ:

$$P(t|s) = \frac{e^{\sum_{i=0}^{n} \lambda_i \cdot \theta_i(s,t)}}{\sum_{t'} e^{\sum_{i=0}^{n} \lambda_i \cdot \theta_i(s,t')}} \tag{2.1}$$

Once again, we are only interested in selecting the most likely translation t for which it is unnecessary to compute the denominator:

$$\begin{aligned} \tilde{t} &= \operatorname*{argmax}_t e^{\sum_{i=0}^{n} \lambda_i \cdot \theta_i(s,t)} \\ &= \operatorname*{argmax}_t \sum_{i=0}^{n} \lambda_i \cdot \theta_i(s,t) \end{aligned} \tag{2.2}$$

It should be noted that this decision rule becomes a simple linear model. However, the SMT community still refers to it as a log-linear model (due to its origins) and we will as well.

Log-linear models incorporate all the information of their generative counterparts. We can represent the original generative model as a log-linear model with the following two features:

$$\begin{aligned} \theta_0 &= P(s|t) \\ \theta_1 &= P(t) \end{aligned}$$

The clear advantage of a log-linear model is that it 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. Furthermore, learning these optimal weights is straightforward with the log-linear model (and will be discussed in Section 2.4).

The feature functions for each translation unit are calculated over occurrences in the training corpus. Generally, they are relative frequencies that count each instance of translation equivalently. Once the model is constructed, the training corpus can be “thrown away” as it provides no further information.

A popular implementation of SMT is the open-source software package Moses (Koehn et al., 2007). Moses creates a log-linear model for translation that was originally described with just eight features. While this represents more information than the original generative model, the number of features needed to represent the translation process is still quite small. In Moses, translation units are represented as phrase pairs consisting of a source phrase s and a corresponding target phrase t. Each phrase pair is assigned the following five features:

- $lex(s|t)$ and $lex(t|s)$[1] that estimate the phrase probability using the word alignment
- $p(s|t)$ and $p(t|s)$ that estimate the phrase probability with relative frequencies using the phrase alignment
- A constant phrase penalty to prefer the use of longer phrases

The following three features are applied when phrase pairs are combined together during decoding:

- A language model score
- A word penalty to balance the language model's bias toward short sentences
- A distortion penalty to limit reordering

A more detailed description of these features is present in Koehn et al. (2003). Moses has been extended over the years, and to be clear, this is not an exhaustive list of possible features. 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 that are now also included in the default configuration of Moses. However, nearly all extant SMT systems have the eight features listed above at their core because their log-linear combination has been found to perform quite well. The point is that by understanding these features, the reader should be able to grasp the foundation of most SMT systems.

SMT's simplicity, scalability, and training procedures have 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.

Our model is best described as an extension of the SMT model that integrates ideas from instance-based learning. As the log-linear model expanded on the generative model, our model expands on the log-linear model for SMT. The new source of information that our model facilitates is that features can individually score instances of translation and not just translation units. It is as an extension of the traditional SMT model in that it is also capable of exactly representing the SMT model.

2.2.2 Example-Based Machine Translation

Before the rise in popularity of SMT, the focus in data-driven machine translation was on Example-Based Machine Translation (EBMT). IBM's introduction of the first statistical machine translation system in 1990 (Brown et al., 1990) disrupted the field of machine translation. Statistical models were capable of producing good results, but their methods were not well understood. Unlike rule-based systems, when a translation error occurred, it was difficult to pinpoint the cause or correct the translation. One way to view EBMT is that it represented a middle-ground between data-driven and mainstream linguistic approaches to translation. EBMT is a corpus-based approach, but one in which there is a desire to make the translation process intuitive to the end-user. While statistics

[1] Given a word alignment a and lexical weights w:

$$lex(s|t) = \max_a \prod_{i=1}^{|s|} \frac{1}{|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(s_i|t_j)$$

$$lex(t|s) = \max_a \prod_{i=1}^{|t|} \frac{1}{|(i,j) \in a|} \sum_{\forall (i,j) \in a} w(t_i|s_j)$$

may play a role in constructing the translation, EBMT systems are often also guided by human behavior or linguistic principles. Somers (1999) provides a detailed survey of EBMT's history.

EBMT systems share the belief that the translation process is best described using previous examples of translation. They store a set of example translations and propose new translations that can be constructed from some combination of the examples. A core requirement is that the system is capable of retrieving example translations that have a high degree of similarity to the input. Thus, the simplest representation or model of the translation process is most often the training corpus itself. Whether a system can reproduce the training corpus has been suggested as a possible litmus test for EBMT.

EBMT was initially proposed by Makato Nagao (Nagao, 1984) as a method for translation by analogy. He credits the way humans process language as the inspiration for this approach. He argues that by comparing similar training examples, the system should be able to learn correspondences between the structure of sentences and the words therein. On receiving a new sentence to translate, the first step of an EBMT system is to decompose the sentence into smaller phrases (he suggests case frame units). The system will then translate each of these smaller phrases by analogy. For each phrase, the system selects a similar training example to use as a template. Within this training example, the system will replace words that differ from the input sentence. The substitution may be performed by using other training examples or an external lexicon.

Gaijin (Veale and Way, 1997) represents one implementation of this approach. Gaijin is designed to leverage the Marker Hypothesis (Green, 1979) which posits that a closed set of words in every language can be used to identify the syntactic structure of a sentence. These markers are typically conjunctions, prepositions, determiners, and quantifiers. Each marker indicates the start of a constituent phrase. Gaijin selects one example from the training corpus that exhibits the same ordering of markers as the input sentence to serve as a master template for translation. The input sentence and master template will, thus, have the same number and type of constituent phrases. The master template is then adapted to the the input through the *high-level grafting* and *keyhole surgery* operations. *High-level grafting* replaces an entire constituent phrase in the master template with a more similar phrasal constituent from the training corpus. *Keyhole surgery* substitutes individual words within a phrase constituent to match the input sentence. These operations preserve the structure of the master template by only permitting the substitution of phrase constituents that have the same head-of-phrase marker and the substitution of words that have the same part-of-speech label. The particular sequence of constituent phrases in the master template, rather than a language model, dictates the phrasal selection and reordering.

Alternatively, Panlite (Brown, 1996) is a shallow EBMT implementation that it is guided simply by lexical matches to the input sentence. Instead of matching a deeper linguistic structure, any combination of examples that covers the input sentence is allowed. Each example that matches a subsequence of the input sentence is scored according to its surrounding context, alignment probability, genre, and location in the corpus. Panlite then searches for a combination of examples that is maximally similar to the input sentence. Brown (2000) extended this system to permit generalized examples. In this later work, the examples were allowed to contain equivalence classes such as a person's name or date. The generalization of examples enabled broader coverage and the use of longer examples.

Gaijin and Panlite illustrate two very different implementations of EBMT. Other implementations exist as well, some of which employ even deeper structures, such as the syntax or semantics of a phrase (Nakazawa et al., 2006; Way, 2003). The point is that EBMT systems differ widely in their methods of matching examples, allowed generalization, and combination of examples. Scoring is performed on an example-by-example basis and often includes several heuristics. Unlike SMT, there is no well-defined model common to all EBMT systems. EBMT defines an approach, but not a particular model or method. The research community has now generally moved away from EBMT as SMT systems have performed better in competitive evaluations.

To be clear, EBMT is an early application of instance-based modeling to machine translation. Like EBMT, the modeling approach we propose constructs a hypothesis based on instances of translation. However, our contribution is novel in that it embeds the modeling of each translation instance within a larger statistical framework. Furthermore, our approach significantly outperforms existing EBMT systems (Phillips, 2011).

2.2.3 Lazy Machine Translation

Lazy machine translation systems are characterized by their implementation and not their model. EBMT systems necessitate the extraction of examples during translation and are, thus, commonly implemented with on-demand phrase alignment and scoring algorithms. SMT systems have no such restriction, and the advantage of a parametric model is that it can be learned once and stored on disk. However, SMT models have become very large. In order to reduce the memory-footprint of large phrase tables, lazy SMT systems have also been developed that extract phrase pairs or grammar rules on demand.

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 longer phrase pairs can 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.[2] This is made possible by the fact that the features used in a standard SMT model are very simple and that the nature of suffix arrays makes counting the occurrences 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 dataset and perform Hiero-style (Chiang, 2005) 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. 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 in a lazy manner) 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.

Cunei's implementation is also lazy. We build upon the work outlined above and use many of the same techniques (such as suffix arrays). While lazy machine translation systems are not the norm, the fact that our implementation is lazy is not in and of itself that significant. The importance of the lazy implementation in Cunei is that it enables on-demand retrieval of related instances of translation that can be scored individually with contextual, feature-rich models.

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

2.3 Extending and Adapting Machine Translation

Each section below surveys a different way in which researchers have extended the traditional SMT model. While none of the work below leverages instance-based learning, the information they use and the problem they address is similar. Most often these improvements are captured by adding new features to the model or building multiple, independent models. We will expand on these ideas and sometimes even generate similar features, but we will apply them within our modeling approach that scores the relevance of each translation instance. In Chapters 6, 7, and 8, we will present how our approach leverages instance-based learning to capture source and target context, perform adaptation, and measure similarity. In doing so, our approach unifies these threads of research together within a single instance-based model.

2.3.1 Source-Side Context

The MT community generally agrees that source 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 research in machine translation with richer context models 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 (Chiang, 2005). 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 allowing weights to be learned for these context features that maximized end-to-end translation performance. Their baseline Hiero system trained on the Chinese-English FBIS corpus scored 29.73 BLEU when evaluated on NIST MT03. The same system extended with 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 machine translation. Like Chan et al. (2007), they now modeled context with features in the log-linear model. 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 part-of-speech, and basic dependency features. These context features were integrated in a phrase-based SMT system and appended to entries in the phrase table. This process significantly enlarged the phrase table because each entry in the phrase table identified a unique context.

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 sentence. Some of these features, such as using colocations of lexical or part-of-speech 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 whether the phrase is (exactly) a constituent, the nonterminal label of the lowest node in the parse tree that covers the phrase, whether it occurs at the beginning of the sentence, or the fraction of the words in the input sentence that are covered by the phrase. The result was minor gains on in-domain experiments in Chinese-English and German-English, but large improvement

(36.15 to 37.72 BLEU) on an out-of-domain Chinese-English experiment that included additional United Nations data.

We present how our model incorporates source context in Chapter 6. In fact, the types of information we will use, such as local collocations, are quite similar. The difference is that we are not merely adding new features to a parametric model, but we are using a different type of model that scores phrase pairs with instance-based learning. The additional contextual information is used to identify relevant instances of translation instead of fragmenting phrase pairs into more specialized units.

2.3.2 Target-Side Context

In the methods above, translation matching on the source-side was extended using knowledge from similar phrases. Context-Based Machine Translation (CBMT) (Carbonell et al., 2006) focuses on the other side of a translation–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 context 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.

In Chapter 7 we will present our technique for leveraging target context within our model. Essentially, we use the same methods we explored for modeling context on the source and apply them to the target. Our approach scores instances of translation as being more relevant if they share context with the target hypothesis. This technique will prefer translations that are embedded in longer target n-grams which captures a similar effect to CBMT's decoding with overlapping n-grams. However, CBMT is a more aggressive approach that alters the decoding paradigm to enable target hypotheses that have not been seen in a bilingual corpus. Our approach models target context, but it does so within the standard SMT framework for decoding.

2.3.3 Source-Side Paraphrases

Building upon research in paraphrasing, researchers have augmented the standard SMT model 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. The probability of the paraphrase, $p(s_2|s_1)$ for paraphrases or 1 for original entries, was also added as a new feature to the phrase table. The primary finding of this work was that paraphrases could be used to significantly improve translation coverage. In experiments using up to 320,000 sentence pairs from the Spanish-English and French-English Europarl corpora, they also report an improvement in BLEU.

Marton et al. (2009) describe another approach that uses 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 feature that scores the similarity between the input sentence and the source of the original phrase pair. The purpose of this feature is to discount the score of phrase pairs generated via paraphrasing. For phrase pairs original to the model (not generated via paraphrasing), the value of this feature is zero and the score for these phrase pairs remains unchanged. The authors artificially limit their approach to unknown words and unfortunately do not report any results when their approach is applied to all entries in the 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.

We mention this work because it is a natural fit with our model, but it is not an area we specifically pursue in this work. Normally, Cunei selects instances of translation that lexically match the desired source phrase. However, since we score each instance of translation with a distance function, it is straightforward to extend the distance function with similarity features and allow instances of translation that are paraphrases. The closest we come to this is in Chapter 8 where we allow instances of translation that match a sequence of annotations, but may lexically diverge from the input sentence.

2.3.4 Continuous Space Modeling

One of the beautiful, but troubling, characteristics of natural language is that there are often many different ways to express the same concept. Modeling the similarity of translations, such as the use of paraphrases, allows for generalization 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, 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 dissertation intends to address. 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*. These issues with LSA may be addressed in the future by constructing a continuous model with a different method, but there is currently very little work in this area for machine translation.

Our model is not continuous, but it does exhibit some continuous characteristics when scoring a phrase pair from multiple instances of translation. In particular, our distance function allows us to glean information from similar instances of translation that do not exactly match the source or target. By scoring the relevance of each translation instance, we do not have to discretely bin those with the same source and target phrase; rather, each instance of translation can contribute to multiple hypotheses.

2.3.5 Weighting and Filtering Corpora

Data-driven statistical models are merely a reflection of the text they are trained on. Instead of adding new features to the model, one can also change the hypotheses a machine translation system generates by altering the data from which the model is built. Filtering or re-weighting the training data is effective at globally skewing the translation model by altering the model's parameters. This technique is frequently used for domain adaptation to guide the translation toward a particular target.

When dealing with corpora in multiple domains, perhaps the most natural approach is to build multiple SMT models. Foster and Kuhn (2007) and Lu et al. (2007) describe mixture-model approaches in which the corpus is partitioned and traditional SMT models are built on each component. Lu et al. (2007) weight each component based on its TF-IDF similarity to the test set. Foster and Kuhn (2007) explore multiple distance metrics and find that an EM approach maximizing the likelihood of the test set provides the best mixture weights.

Hildebrand et al. (2005) is an early approach that applies information retrieval 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. They evaluated this approach on Chinese-English tourism dialogue while the training data was dominated by newswire and speeches. 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. While Hildebrand et al. (2005) filter the training corpus so that it is maximally similar to the test set, Lu et al. (2007) extend this idea to re-weight the training corpus based on each sentence's TF-IDF similarity with the input document.

Similar techniques that filter a large dataset for relevant, in-domain sentences have also been applied to language modeling. For example, Moore and Lewis (2010) use cross-entropy to filter a large collection of text in order to build a high-quality, in-domain language model. They build one language model on a small set of known in-domain text and one general-purpose language model from a random sample of text. A score is assigned to each sentence by computing the difference of the cross-entropy of the text according to these two language models. A larger domain-specific language model is then created from all sentences with a score greater than a threshold value which is determined on a held-out set of in-domain data. Moore and Lewis (2010) compare this technique to earlier work and show that they achieve lower language model perplexity on a held-out test set. However, they do not show the effect of using their adapted language model on end-to-end performance during translation.

More recent work in translation modeling has focused on learning weights for components of the corpus instead of using a pre-defined similarity metric. Matsoukas et al. (2009) assigned weights to sentence pairs in the training corpus so that the resulting SMT model maximized an objective function. The weight for each sentence was calculated with a perceptron model that used several simple feature functions. The perceptron was trained by minimizing the expected Translation Error Rate (Snover et al., 2006) over an n-best list of target hypotheses generated by the SMT model on a development set. As the sentence weights were updated, the phrase and lexical translation probabilities in the SMT model also needed to be re-estimated. To account for this divergence, the n-best list was regenerated every 30 iterations. The researchers found that when the development set was similar to the test set, their approach led to significant gains across multiple evaluation metrics. Furthermore, the baseline Arabic-to-English system that they improved upon is one of the best in the world.

The idea of weighting components in the corpus captures the essence of what we are trying to achieve with instance-based learning. Our work is most similar to Matsoukas et al. (2009) in that we use multiple features in our distance function and learn weights for them based on a development set. Nonetheless, our method is quite different from all of these efforts. First, all of the methods described above weight documents or sentences within the training data, whereas we go one step more specific. Our features score instances of translation instead of sentences or documents. Translation instances are embedded within a particular sentence, but they represent smaller units. Second, in all of the related work, either implicitly or explicitly there two separate models: the corpus model and the SMT model. The SMT model is built on top of the corpus model in that the score for a sentence according to the corpus model is used in the feature function calculations of the SMT model. The weights applied to the corpus are separate from the weights applied to the feature functions of the SMT model. Our approach uses instance-based learning to construct a single unified model with one set of weights.

2.4 Learning Model Weights

A lure of the log-linear model is that it allows for numerous, arbitrary features. 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. While theoretically *any* feature can be used, in practice not all features are equally useful and careful feature engineering can significantly improve a system's performance.

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 objective function is convex with a 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 1-best translation hypothesis. MERT uses an efficient linear programming solution to select weights that favor the optimal translation 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.

It is desirable to be able to employ many features and allow the learning algorithm to sort the good features from the bad features with appropriate model weights. However, MERT only works well when the number of parameters is relatively small. Liang et al. (2006a) first exploit large feature sets in machine translation by applying a perceptron-based learning algorithm. Perceptron learning uses a very simple update rule to learn weights by computing the difference between the feature values assigned by the model to the hypothesis translation and to the reference translation. A problem with applying perceptron learning to machine translation is that the reference translation is not always reachable by the translation model, and even when the reference translation is reachable it may not be the result of a valid alignment. The simple solution is to score the reference if it is reachable and otherwise skip that sentence during training. However, Liang et al. (2006a) report better results by selecting the hypothesis translation with the highest BLEU score in the n-best list to act as the reference translation when performing the update.

Liang et al. (2006a) applied this discriminative approach to a phrase-based SMT system with limited reordering capabilities. In addition to the translation probability and language model probability, they included alignment constellation features, part-of-speech features, and lexical features. The lexical features were particularly numerous and allowed fine-grained control of the translation process. Each lexical feature denoted the presence of either a specific translation or a specific output phrase (independent of the source). In total, the researchers augmented the translation model with over one million features and achieved an improvement of 1.3 BLEU on a French-English test set.

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 when using a large number of features.

Chiang et al. (2009) use MIRA to augment a Hiero-based MT system with over 10,000 features and a syntax-based MT system with more than 250 features. Most of the features were linguistically informed to catch specific situations where grammar rules failed. In addition, the authors added features to identify source-side context and to apply discount weights to low-frequency translations. 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 from MIRA with a 1.1 BLEU gain with the syntax-based system and a 1.5 BLEU gain with the Heiro-based system on a Chinese-English test set.

Smith and Eisner (2006) propose as an alternative to MERT 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 model's probability of the translation given the input sentence. Unlike MERT, it is no longer sufficient to only identify the translation with maximum score. To compute the probability, one must divide the score of a translation by the summation of all translation scores (we must compute the denominator in Equation 2.1). 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.

Chiang (2012) recently compared the difference between MIRA and minimum risk as the objective function for learning. MIRA came out slightly ahead (44.9 vs 44.8 BLEU) on a small Arabic-English translation system with 13 features. However, in this scenario MERT actually achieved the best score of 45.2 BLEU on the held out test set. The real advantage of both Watanabe et al. (2007) and Smith and Eisner (2006) are that they are capable of scaling to many more features than Och (2003). Adding a large number of simple, discriminative features to a model can achieve larger gains than selecting a specific learning algorithm (Chiang, 2012; Chiang et al., 2009; Liang et al., 2006a).

The idea of using a large feature space is itself orthogonal to the modeling approach. 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. However, the fact that we score translation instances with a distance function allows us to incorporate similarity features that are not so rigidly lexicalized and can be used in multiple situations. We expect such similarity features (which are only possible due to our instance-based scoring) to be of greater utility than a large number of very specific lexical features. That being said, all of the features described in Liang et al. (2006a) and Chiang et al. (2009) could be added to our model as well; it is simply not an area we explore in this dissertation.

As far as the learning algorithm, we will present in Chapter 4 that our implementation uses the minimum risk approach of Smith and Eisner (2006). This decision was simply a matter of practicality and based on what methods were known at the time and were straightforward to implement with our model. Overall, the thesis is neutral toward the objective function and the algorithm used to learn the model weights.

Chapter 3

Overview of the 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 modify an existing system. Much of the preparatory work for this dissertation went into building Cunei, a new platform for machine translation, that is state-of-the-art and supports instance-based modeling. We have released Cunei as open-source software that is available for download at `http://www.cunei.org`. Much of this chapter was first published in (Phillips, 2011).

3.1 Approach

Due to ever-increasing bilingual corpora, data-driven machine translation is able to locate many potential translation instances that are similar to the input sentence in whole or in part. Yet these translation instances are not all equally suitable for the translation task at hand. Each translation instance in the corpus occurs within a unique linguistic context. As mentioned in the introduction, each translation instance–even if it predicts the same target hypothesis–may be associated with a different parse tree or morpho-syntactic information. What makes Cunei unique is that it is able to score this information (by comparing it to the input sentence or target hypothesis) *individually* for each instance of translation. We exploit this per-instance information to distinguish when some translation instances are more relevant than others.

Generally we do not expect to find a translation in the corpus that is identical to the input sentence. Instead, data-driven machine translation identifies possible translations for subsections of the input sentence which are then combined together to generate a complete target hypothesis. Each of these smaller units of translation represents the correspondence between a particular source and target. In phrase-based implementations such as ours, these units are referred to as phrase pairs. A typical SMT system will score these phrase pairs with a series of feature functions. As described in Section 2.2.3 of the last chapter, these feature functions can be computed on demand, but more often they are computed once over the entire training data and the system stores a large correspondence table between possible phrase pairs and feature values. The feature functions calculate the count or relative frequency of various conditioning events in the training data. In the traditional SMT model, each instance of translation represents an event and generally all events in the training data are counted equally.

In Cunei, a distance function measures the relevance of each translation instance. The distance function scores each instance of translation with respect to the input sentence, the target hypothesis, and the phrase pair being scored. This process was illustrated in Figures 1.1 and 1.2 of the introduction. The score for each instance of translation is calculated with a series of feature functions similar to how the traditional SMT model scores a phrase pair. The difference is that these feature functions are computed for each instance of translation. The score assigned to a phrase pair then is the summation of the score for all instances of translation. The effect of this calculation is similar to selecting the best instance of translation except that the presence of multiple high scoring translation instances will result in an even greater score. The intent is that the distance features can be maximally discriminative and score each instance individually. However, any traditional SMT feature can also be incorporated in our distance function.

Conceptually, this approach bears similarity to weighted k-nearest neighbors as described in Section 2.1 of the previous chapter. We score a phrase pair by identifying instances of translation that are 'nearest' to the input sentence and target hypothesis. In this sense, our model is instance-based. However, this instance-based score for each phrase pair is still situated within the broader SMT framework for decoding. As presented next in Section 3.1.1, Cunei's combination of the scores for the phrase pairs used to generate a complete target hypothesis is the same as in a traditional SMT system.

The translation process begins by identifying all possible contiguous phrases in the input sentence. For each of these phrases in the input sentence, Cunei locates relevant source phrases in the training corpus and performs alignment to identify the corresponding target phrases. Each *occurrence* of an aligned source and target phrase is extracted as a unique translation instance. These instances of translation are scored individually with a series of feature functions. Cunei then generates a phrase pair by summing the score of translation instances that predict the same target hypothesis. The target hypothesis is generated by finding the combination of phrase pairs with highest score.

3.1.1 Formalism

The SMT formalism as typically presented (and as described in the last chapter in Section 2.2.1) presupposes we are capable of generating the entire target hypothesis. This makes the model easier to understand, but it is also not very practical. In fact, the scoring function for the SMT model is generally decomposable. This is accomplished by identifying an alignment, a, that segments the input sentence and target hypothesis into smaller units of translation. In the formalism below, these units of translation are phrase pairs, an abstract representation of one or more source words that align to zero or more target words. The model scores a collection of phrase pairs by summing over the score of each phrase pair that is identified in the alignment. The SMT system then searches all possible alignments, A, and selects the hypothesis with the highest score according to the model. Given a function m to score each phrase pair, we can abstractly describe the decision rule for selecting the target sentence $\tilde{t}$ given the source sequence $s_1, s_2 ... s_n$ as:

$$\tilde{t} = \underset{t_1,t_2...t_n}{\operatorname{argmax}} \sum_{\langle s_i,t_i\rangle \in a} m(s_i, t_i, \lambda) \quad \forall a \in A \tag{3.1}$$

This decision rule selects the highest scoring target hypothesis *for any one alignment*. However, it is possible for the same target hypothesis to be reached through multiple alignments. While less often used in SMT systems, one can alternatively specify a decision rule that selects the highest scoring target hypothesis *over all alignments*:

$$\tilde{t} = \underset{t_1,t_2...t_n}{\operatorname{argmax}} \sum_{a \in A} \sum_{\langle s_n,t_n\rangle \in a} m(s_i, t_i, \lambda) \tag{3.2}$$

In fact, Cunei implements this latter decision rule, but both Equations 3.1 and 3.2 are representative of the broader SMT framework for decoding. In either of the above decision rules, the function m scores each phrase pair which consists of a target phrase t_i and a corresponding source span s_i.[1] The target hypothesis $\tilde{t}$ represents the sequence of target phrases $t_i, t_2, ...t_n$ from the selected collection of phrase pairs.

Within this framework, a typical log-linear SMT model with features θ and weights λ is represented as:

$$\begin{aligned} m(s_i, t_i, \lambda) &= \ln e^{\sum_k \lambda_k \cdot \theta_k(s_i, t_i)} \\ &= \sum_k \lambda_k \cdot \theta_k(s_i, t_i) \end{aligned} \tag{3.3}$$

The linear combination of feature functions and weights in Equation 3.3 is used by the model to assign a score to each phrase pair with source s_i and target t_i. Note that this score does not form a valid probability distribution. Substituting Equation 3.3 into Equation 3.1 results in the SMT model presented previously as Equation 2.2 on page 21. Equation 2.2 scores an entire sentence, whereas Equations 3.1 and 3.3 illustrate the decomposition of the sentence into phrase pairs.

Evidence for a phrase pair is present at multiple locations within the training data. We refer to each occurrence of a phrase pair in the training data as an instance of translation. The feature functions in the SMT model, θ, operate over this set of translation instances. However, the instances of translation are simply viewed as countable events where each event generally contributes the same weight. A common feature, for example, is the number of times s_i and t_i are aligned divided by the total occurrences of s_i. Once these features are estimated, the traditional SMT model no longer requires the training data.

Cunei's model for translation is fundamentally different in that we score a phrase pair by directly comparing it to relevant instances of translation in the training data. We calculate the distance from an instance of translation x in the training data to the phrase pair $\langle s_i, t_i \rangle$ as:

$$d(s_i, t_i, x) = e^{-\sum_k \lambda_k \cdot \phi_k(s_i, t_i, x)}$$

The distance function, $d(s_i, t_i, x)$, evaluates not only whether $\langle s_i, t_i \rangle$ is a good translation in general, but also specifically if it should be used for a particular span of the input sentence and if it should be used to extend the current target hypothesis. We will demonstrate in the following chapters how to discriminate between translation instances by scoring their alignment, source context, target context, and annotations. Similar to Equation 3.3, this function is feature-based, which allows the system engineer to easily integrate new sources of knowledge. In both the distance function and Equation 3.3, the feature functions θ and ϕ are calculated with the training data and the weights λ are learned. The difference is that ϕ evaluates one specific instance of translation instead of scoring the entire set of translation instances.

The distance function assesses how 'far away' each translation instance is from the ideal candidate. We use the distance function to relate our approach to instance-based learning, but the score assigned to a translation instance is the inverse: $\frac{1}{d(s_i, t_i, x)}$. Our model scores a phrase pair by

[1] For simplicity we only include the source span s_i, but both the SMT model and Cunei's model can be extended to include the entire input sentence as a component of the model.

summing over the set X of translation instances:

$$\begin{aligned} m(s_i, t_i, \lambda) &= \ln\left(\frac{1}{|X|}\sum_{x\in X}\frac{1}{d(s_i, t_i, x)}\right) \\ &= \ln\left(\frac{1}{|X|}\sum_{x\in X} e^{\sum_k \lambda_k \cdot \phi_k(s_i, t_i, x)}\right) \end{aligned} \tag{3.4}$$

Here x represents an instance of translation which identifies a unique location within the training data where a source phrase s' translates as a target phrase t'. The first two arguments of the distance function, s_i and t_i, refer to the candidate source and target phrase that we are trying to construct and not necessarily to the source phrase s' or target phrase t' at this position in the corpus. The summation is conceptually over all possible instances of translation, but in practice we sample a set of the most relevant translation instances (this will be discussed further in Section 3.2.2). The feature function ϕ informs the model how relevant the translation instance x is for modeling the phrase pair $\langle s_i, t_i\rangle$. The feature function ϕ may include information such as the alignment probability between s' and t', the similarity between s_i and s', the location of x in the corpus, or other contextual knowledge.

For illustration, consider that the translation instances for a given phrase pair occur in a variety of sentences within the training data. Some instances may include an inconsistent word alignment from within the selected phrase pair to a word in the remainder of the sentence. Cunei's model allows us to learn from these translation instances, but discount them by including a feature in ϕ which scores the phrasal alignment given the words *outside* the phrase pair. This differs from the standard SMT approach where phrase alignment is a binary decision. The same principle also applies if we want to include additional non-local information such as genre or context within the model. A traditional SMT model requires specialized phrase pairs conditioned on the extra information, whereas Cunei models the non-local information as features of ϕ and calculates a score over *all* instances of translation. As discussed in Section 2.1 of the last chapter, instance-based methods provide smooth estimates and inherent generalization.

The scoring function presented in Equation 3.4 is similar to k-nearest neighbors with inverse-distance weighting (Shepard's method) and density estimation with a kernel model. Recall, a more detailed description of these instance-based methods was presented in Section 2.1 of the previous chapter. The difference is that Cunei does not compute a valid probability distribution. In this respect, Cunei is no different from a standard SMT system. Note that both Equations 3.3 and 3.4 assign unnormalized scores to each phrase pair.

While our approach is instance-based, it is also a feature-based model situated in the broader SMT framework. The result is that the standard log-linear model used in SMT is a special case of Cunei's model. When the features for all instances of a translation are constant such that $\phi_k(s, t, x) = \theta_k(s, t) \quad \forall x\, \forall k$, then Equation 3.4 is equivalent to Equation 3.3. In fact, we will show in Section 3.2.4 that Cunei's model can incorporate any standard SMT feature. Thus, Cunei can represent the traditional SMT model and expand beyond it with per-instance features. Like other SMT systems, Cunei is also able to learn model weights that minimize error on a development set (this will be the focus of Chapter 4).

Cunei's approach differs from SMT only in how each phrase pair is modeled. This is illustrated above in the different definitions for m (Equations 3.3 and 3.4). Both approaches apply the same framework (with either Equation 3.1 or 3.2 as the decision rule) to combine phrase pairs together and select the target hypothesis.

3.2 Runtime Execution

Sufficient statistics for the features of each translation instance could be computed offline, but the space requirements would be quite large. Furthermore, most of the model is unnecessary for a particular input. As a result, Cunei delays as much computation as possible until an input sentence is available for translation. Translations are not retrieved from a pre-built phrase table, but rather generated on demand. Scoring translations in a lazy manner has two key advantages:

1. On-demand feature extraction makes it easy to score non-local features dependent on the particular input or surrounding context. This is a key strength of Cunei that we will leverage in later chapters.
2. Retrieving translation instances on demand allows Cunei to explore a larger search space. We are not required to generate all possible generalizations or compute similarity features for all possible inputs *a priori*, which would be expensive and necessitate significant pruning. Instead, Cunei can limit the search space to translation instances that are relevant to the input sentence.

The remainder of this section will describe in detail how translations are constructed on demand.

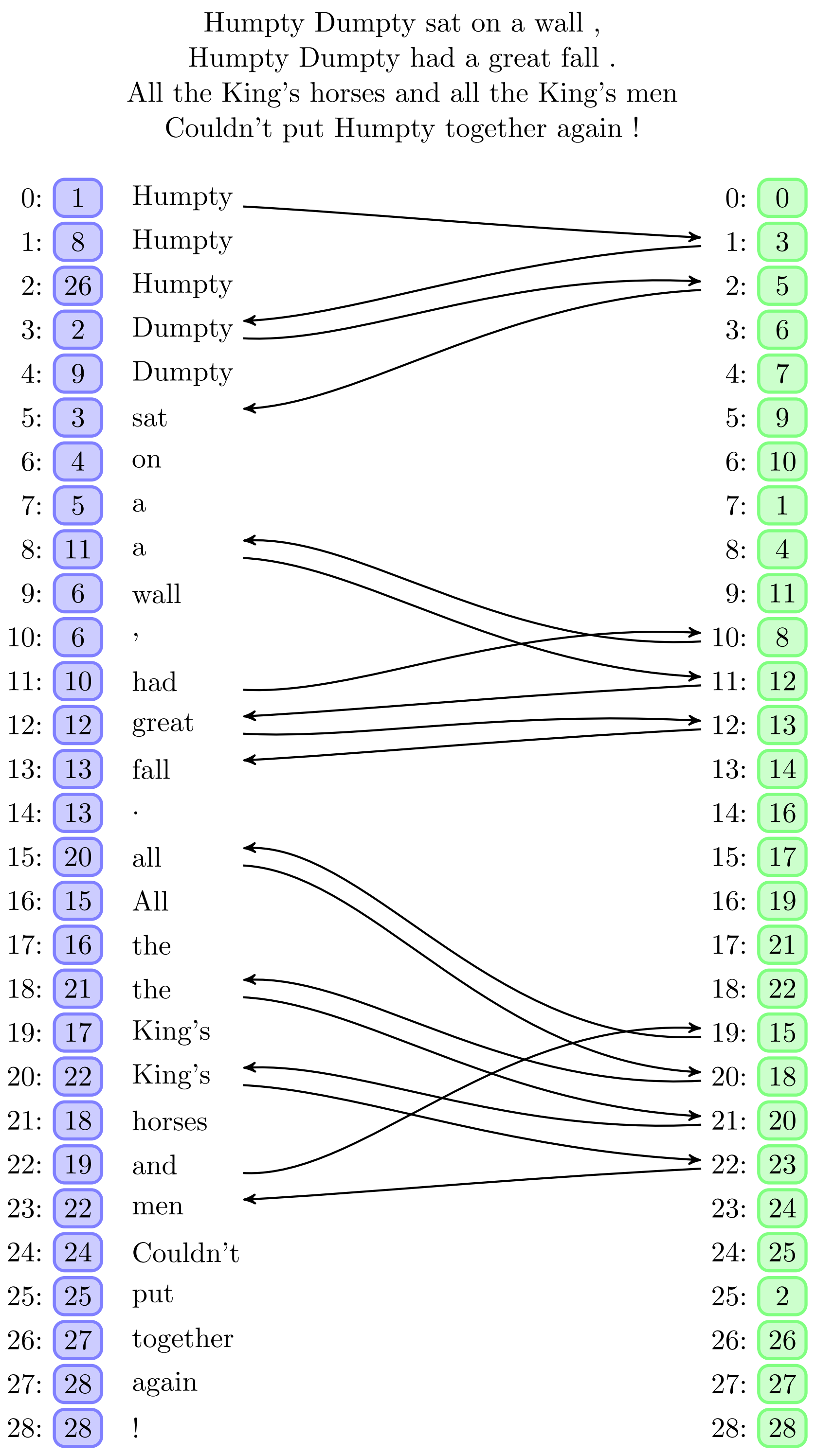
3.2.1 Locating Matches

To support locating translations on demand, Cunei constructs a suffix array for each side of the parallel corpus. Suffix arrays provide a compact and efficient data structure for locating arbitrary sequences of tokens within a large corpus (Yamamoto and 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.[2] As evidenced by the work of Brown (2004), Zhang and Vogel (2005), Callison-Burch et al. (2005), and Lopez (2008), suffix arrays are also increasingly popular in machine translation.

Finite state machines are often used to represent sequences of tokens, but they are insufficient for this task. A finite state machine could encode what phrases are present in the corpus, but unlike a suffix array, the corpus cannot actually be reconstructed from the finite state machine. Several of the features we calculate, such as those related to context, require reconstructing a portion of the corpus. In order to continue supporting these features, we would have to store both the corpus and a finite state machine that points to specific positions within the corpus. The suffix array allows us to store a single data structure that is both compact and efficient.

We minimally alter the standard suffix array data structure to store the position of each token in the corpus. 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. Adding position information allows us to query both *where specific tokens occur* and *what tokens occur at a specific position*. A suffix array augmented with an additional position array as implemented in Cunei is shown in Figure 3.1. The standard suffix array includes only the array on the left. In addition, the pointers in a standard suffix array would point directly to the next suffix in the corpus. Our modification simply pivots the pointers through the position array on the right. This provides a mapping from both suffixes to positions and positions to suffixes. Note that the keys

[2] 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 Cunei.



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 3.1: A Suffix Array with Position Information.

Algorithm 1: Lookup Corpus Matches

```
for $p = 0$ to $|\text{input}| - 1$ do
    for $q = p + 1$ to $|\text{input}|$ do
        matches ← SearchCorpus(input[$p...q$])
        if list matches is empty then break
        matches ← SampleMatches(matches)
        Append list matches to matchLattice at span $\langle p, q \rangle$
    end
end
```

are not sorted according to their natural order but rather based on when the type first appeared in the input. This is reflective of how the suffix array is actually constructed by Cunei as the entire vocabulary is not known beforehand.

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 bits per entry 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.[3] 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. The memory-mapping actually permits Cunei to use data structures that are larger than resident memory by relying on the operating system to swap in the required pages.

When provided a new sentence to translate, Cunei searches the source-side of the corpus for phrases that match any subsequence of the input sentence. A match may contain as few as one of the tokens from the input sentence or represent the entire input sentence. We start searching the suffix array for the smallest possible match–one token–and incrementally add one more token to the sequence every time a match is found. Once we are no longer able to locate a particular sequence of tokens in the corpus, we can also stop searching for any sequences that subsume it. When the suffix array does locate the sequence of tokens, we are provided a list of positions in the corpus where they occur. This collection of corpus matches is sampled (discussed next in Section 3.2.2) and stored in a lattice where each entry is indexed by the span of the input sentence it covers. The procedure for constructing the lattice of corpus matches is defined in Algorithm 1.

Generally, we only retrieve instances of translation that are exact lexical matches with some subsequence of the input sentence. The suffix array does not permit an efficient mechanism for locating matches that contain insertions or deletions. However, we store all of Cunei's partial matches in a lattice. Therefore, we could 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 occur in the same sentence with only a few extra tokens between them, then we could identify a discontiguous match. Due to sampling, the lattice may not contain all corpus matches and this technique will not be

[3] 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`.

able to find all discontiguous matches. That being said, the discontiguous matches that this naïve method will not locate are those where the individual components of the discontiguous phrase occur frequently (so as to saturate the sampling). As they occur frequently, the individual components of the discontiguous phrase may also translate well independently. A more robust (and also more complicated) alternative is described by Lopez (2007). We have implemented the naïve method, but it is not enabled by default. We mention this to note some of the restrictions of working with suffix arrays. We will, however, leverage multiple suffix arrays in Chapter 8 to locate divergent, albeit not discontiguous, matches.

3.2.2 Sampling Matches

As mentioned previously, the score for a phrase pair is formally defined as the summation over all instances of translation. However, it is not practical to actually calculate the score for every single instance of translation in the corpus. Many instances of translation, such as those that represent a different source phrase, would receive a score that is infinitesimally small. Instead, we calculate the score for a phrase pair by summing over a small sample of relevant translation instances.

The set of translation instances Cunei scores is initially restricted by the matches we find in the suffix array. Since our suffix array only locates exact matches, this means that the source phrase for all translation instances will be equivalent to the source phrase of the candidate phrase pair ($s_i = s'$ in Equation 3.4). In Chapter 8, we will will explore loosening this restriction by introducing multiple suffix arrays.

The suffix array enables Cunei to efficiently identify all matching positions in the corpus, but what we do with each match–notably, perform alignment and generate a translation instance–is too expensive to perform on this entire set. Some high frequency words, such as stop words, can occur tens of thousands of times in the corpus. However, this behavior is rare and content words usually occur less than a few hundred times in the training data. This distribution in natural language is known as Zipf's law (Zipf, 1932). The number of matches we process is the single largest factor in determining the overall speed of translation because it limits how many instances of translation are scored. While there is a trade-off between speed and accuracy, a reasonable balance can be obtained with a small sample of translation instances. We have found in practice that after a certain threshold, scoring additional translation instances has minimal impact on the score for a phrase pair. In order to select an appropriate set, we perform two levels of sampling on the collection of corpus matches identified by the suffix array.

The first level of sampling as shown in Algorithm 2 simply reduces the number of matches to a manageable size for further analysis. From the suffix array, we obtain the entire range of matches in the corpus, but this may include several thousand positions. 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.[4] The default settings will initially sample up to 750 matches. These matches are then scored with the feature functions that can be calculated when only the source phrase is available.

The second sample as shown in Algorithm 3 selects an even smaller subset of matches to pass on for alignment and complete scoring. The default settings only extract 150 matches as this latter stage is more expensive. This second sample is also uniform in the baseline configuration, as described and evaluated in Chapter 5. With the baseline configuration, the only information we have about a match is its position the corpus. While more recent matches may be preferable

[4] The matches are sampled uniformly over their location in the suffix array. This will select matches that are followed by different suffixes, but it does not necessarily guarantee a uniform distribution over their position in the corpus.

Algorithm 2: Sampling Algorithm #1

```
$m \leftarrow$ value from parameter Matcher.Lookup.Sample
if $|\mathsf{matches}| \leq m$ then return matches
$p \leftarrow 0$
while $p < |\mathsf{matches}|$ and $p < m$ do
    if match matches[$p$] is not complete then break
    Append matches[$p$] to list result
    $p \leftarrow p + 1$
end
while $|\mathsf{result}| < m$ do
    Append matches[$p$] to list result
    $p \leftarrow p + \max(1, \frac{|\mathsf{matches}| - p}{m - |\mathsf{result}|})$
end
return result
```

Algorithm 3: Sampling Algorithm #2

```
$m \leftarrow$ value from parameter Matcher.Lookup.Entries
if $|\mathsf{matches}| \leq m$ then return matches
if parameter Matcher.Sampling is uniform then
    $p \leftarrow 0$
    while $|\mathsf{result}| < m$ do
        Append matches[$p$] to list result
        $p \leftarrow p + \max(1, \frac{|\mathsf{matches}| - p}{m - |\mathsf{result}|})$
    end
end
else if parameter Matcher.Sampling is best score then
    for $p = 0$ to $|\mathsf{matches}|$ do
        scores[$p$] $\leftarrow$ ScoreMatch(matches[$p$])
    end
    order $\leftarrow$ positions in scores array from highest to lowest value
    for $p = 0$ to $m$ do
        Append matches[order[$p$]] to list result
    end
end
return result
```

(those occurring near the end of the corpus) we do not expect the position alone to be strongly correlated with the complete score for translation instance. However, we maintain two sampling stages, because in later chapters we will explore re-sampling the matches based on their partial scores. In Chapters 6 and 8 we will sample matches using new context and annotation features. With these source-side features, we will calculate a partial score and select the highest scoring matches. In either case, when there are fewer matches in the corpus than the desired sample size, Cunei will select all of them. Due to this minimum threshold, only high-frequency words and phrases are actually sampled.

An interesting effect of sampling based on the partial model score is that when the model weights are modified, the sample will generally change as well. This is the case when we explore new model weights in Chapter 4 to maximize an objective function. Theoretically, it is a desirable trait–our sample is selecting the most relevant matches according to the model. However, in practice modifying the sample introduces some instability to the scores of phrase pairs. This instability can complicate learning the model weights, and so a uniform sample is preferable under most conditions.

3.2.3 Phrase Alignment

After a match is found on the source-side of the corpus, Cunei must determine the target phrase to which it aligns. While the training corpus is sentence-aligned, the internal alignment of words and phrases within the sentence pair is unknown. Cunei uses per-instance features to score possible phrase alignments.

The IBM models (Brown et al., 1993) are capable of inducing a statistical word alignment from bilingual text. 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. While the calculations are not exactly the same, Cunei's approach is conceptually modeled after the work of Vogel (2005). A word alignment is computed once and then stored as part of the indexed corpus. Possible phrase alignments are evaluated with a series of features that operate over all word alignments in a sentence pair. These phrase alignment features are calculated on demand and incorporated as part of the score for each translation instance.

For each source-side match in the corpus, we load the alignment matrix for the complete sentence in which the match resides. The alignment matrix, as visually depicted in Figure 3.2, contains scores for all possible correspondences between source word s_i and target word t_j 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, $P(s_i|t_j)$ will be stored 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., 2006b) which symmetrizes the probability model. In this case, $\alpha_S(i,j)$ and $\alpha_T(i,j)$ will both be set to $P(s_i,t_j)$. While both GIZA++ and Berkeley model probability distributions, the α_S and α_T scores need not be normalized.

Cunei uses this matrix of words alignments to compute phrase alignment scores. When a source phrase is aligned to a target phrase, it not only implies that words within the source phrase are aligned to words within the target phrase, but also that the remainder of the source sentence not specified by the source phrase is aligned to the remainder of the target sentence not specified by the target phrase. Scoring the alignment links outside the phrase boundaries helps ensure that the algorithm has not greedily selected an alignment that forces the remainder of the sentence to align poorly. As detailed in Table 3.1, we define separate features to model the probability that the alignment links for words within the phrase are concentrated within the phrase boundaries and that the alignment links for words outside the phrase are concentrated outside the phrase boundaries. In addition, words within the phrase that are not aligned or have weak alignments demonstrate uncertainty in modeling. To capture this effect, we incorporate two more features that measure the number of uncertain alignment links included within the phrase alignment boundaries.

For each match in the corpus, Cunei uses these alignment features to extract a scored n-best list of phrase alignments. Theoretically, every possible target phrase within a sentence is a candidate for alignment. In order to reduce the search space, Cunei calculates the expected location of the phrase alignment in the target sentence. Each location in the target sentence is weighted by the

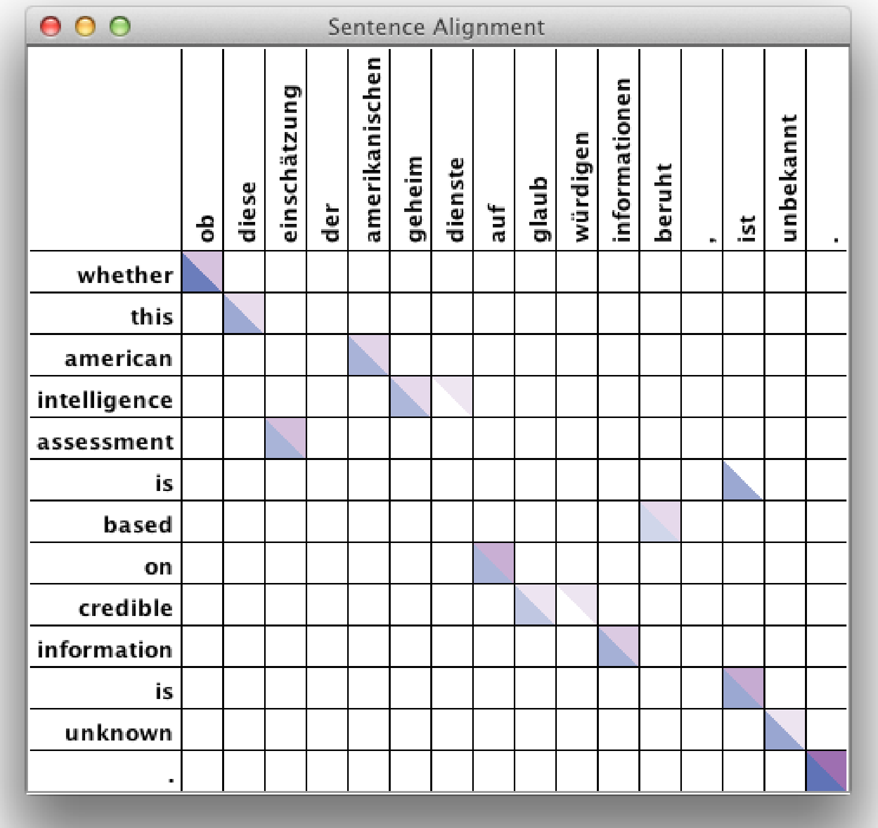


Figure 3.2: Word Alignment Visualization

Alignment in a single direction is represented by a triangle; when an entire cell is shaded, then both directions of the GIZA++ alignments agree.

value of the word alignments that link it to the source match. The expected location for the phrase alignment is determined by taking the weighted average of the possible target locations. Cunei then explores a region within the target sentence that is centered on the expected location of the phrase alignment. The size of this region is guided by a hyperparameter, and it is roughly proportional to the length of the source match. Each target phrase within this region is scored as a candidate for alignment and the results are added to an n-best list. The size of the n-best list is controlled by two 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 per source match. Each of these phrase alignments represent an instance of translation. They identify a correspondence between a source phrase and a target phrase at one particular location in the corpus. From the lattice of source matches, Cunei generates a lattice of translation instances with Algorithm 4.

By contrast, Moses uses a heuristic to identify valid phrase alignments (Koehn et al., 2003). Like Cunei, Moses identifies phrase alignments based on the word alignments, but it does not maintain a score for each word alignment link. The default heuristic `grow-diag-final` symmetrizes the $P(s|t)$ and $P(t|s)$ word alignment models computed by GIZA++. Word alignments that are predicted by

Let $\alpha_S(i,j)$ and $\alpha_T(i,j)$ be the alignment score between the source word at position i and target word at position j (from the external word aligner).

Outside Probability
Let the set of positions in the source phrase and target phrase that are outside the phrase alignment be, 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
Let the set of positions in the source phrase and target phrase that are inside the phrase alignment be, 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 configurable threshold β identifies the value below which an alignment score is considered uncertain.

`Alignment.Weights.Inside.Source.Unknown`	$\sum_{i \in s_{in}} \max(0, \frac{\beta - \left(\epsilon + \sum_j \alpha_T(i,j)\right)}{\beta})$
`Alignment.Weights.Inside.Target.Unknown`	$\sum_{j \in t_{in}} \max(0, \frac{\beta - \left(\epsilon + \sum_i \alpha_S(i,j)\right)}{\beta})$

Table 3.1: Description of Phrase Alignment Features

both models are always selected. This intersection results in high-precision word alignments, but it often does not cover all regions of the sentence pair. Word alignments identified by only one model are added when they do not conflict with an existing alignment link and they grow the existing phrase alignments in a predictable manner.

Moses then extracts all phrase pairs that are consistent with the symmetrized word alignment. This process will extract units of various sizes; the units may overlap in the sense that smaller units may combine together to form the larger units. However, the words in a phrase pair must only align to other words within the phrase pair. A phrase pair is not extracted if any word of the phrase pair aligns to a word elsewhere in the sentence. A sentence with a fully-specified word alignment will have one deterministic set of phrases and alignments.

Moses selects phrase alignments with this heuristic algorithm instead of scoring each phrase

Algorithm 4: Build Translation Lattice

```
foreach span $\langle p, q \rangle$ in matchLattice do
    $s \leftarrow$ input[$p...q$]
    matches $\leftarrow$ list in matchLattice at span $\langle p, q \rangle$
    foreach match $m$ in list matches do
        instances $\leftarrow$ PhraseAlignment($m$)
        foreach instance $i$ in instances do
            $t \leftarrow$ target phrase of instance $i$
            $v \leftarrow$ ScoreInstance($i, s, t$)
            Add score $v$ to phrase pair $\langle s, t \rangle$ in translationLattice at span $\langle p, q \rangle$
        end
    end
end
```

alignment. The algorithm makes a binary decision as to whether or not a source and target phrase should be aligned. This is in contrast to Cunei's approach where an alignment cannot strictly fail, but only result in a very low score. This is helpful for rare phrases where one prefers to predict *something* even if it has a high degree of uncertainty. In addition, Moses's binary heuristic cannot differentiate between the quality of phrase alignments. When Moses computes the features for its translation model, all of the extracted phrase pairs are considered equally likely.

3.2.4 Scoring Translation Units

Translation units are generated by pairing the candidate source phrase from the input sentence with each unique target phrase identified by the set of translation instances. The score for a phrase pair is conceptually the sum over all instances of translation, but to reduce the necessary computation, Cunei limits this set. As already described in Section 3.2.2, Cunei samples the corpus matches, and they are typically restricted to those which are equivalent to the candidate source phrase. When scoring phrase pairs, we also restrict the sum to translation instances with the same candidate target phrase. We will show in Section 7.5 of Chapter 7 that we tried loosening this restriction, but saw no improvement. Formally, this means that $s = s'$ and $t = t'$ when scoring a phrase pair with Equation 3.4 on page 35. However, this is a restriction imposed by the implementation and not the model.

Per-Instance Features

Per-instance features are the features in the distances function that score each instance of translation. As described previously, Cunei scores phrase pairs by summing over these instances of translation. The scoring framework is intentionally flexible such that features are not hard-coded and any number of features can be added to the model and computed on demand. In the most basic configuration of a per-instance model (evaluated in Chapter 5), we only require the phrase alignment features described in the previous section. More per-instance features will be introduced and evaluated in Chapters 6, 7, and 8.

Static Features

In addition to per-instance features, we can also apply more general, static, SMT-like features to a phrase pair. We integrate these features in our model by applying them to every instance of translation. This is possible because Cunei's score for each translation instance takes the same form as the score for a phrase pair in the standard SMT log-linear model. We simply perform the standard SMT feature calculation once over the set of retrieved translation instances and then apply the same feature to all instances. This type of feature is 'static' in that the value of the feature is independent of where in the training corpus the instance of translation is located. Formally, we extend Equation 3.4 to include the the SMT feature function θ in the score for a translation instance:

$$\begin{aligned} m(s_i, t_i, \lambda) &= \ln\left(\frac{1}{|X|}\sum_{x\in X} e^{\sum_j \lambda_j\cdot\phi_j(s_i,t_i,x)+\sum_k \lambda_k\cdot\theta_k(s_i,t_i)}\right) \\ &= \ln\left(\frac{1}{|X|}\sum_{x\in X} e^{\sum_j \lambda_j\cdot\phi_j(s_i,t_i,x)}\right)+\sum_k \lambda_k\cdot\theta_k(s_i,t_i) \end{aligned} \tag{3.5}$$

Note that including these independent features within the summation of translation instances is not necessary. As shown on the second line of Equation 3.5, the calculation of per-instance features ϕ can be separated from the calculation of static features θ. Recall, we mentioned in Section 3.1.1 that if we only include features that are independent of a translation instance, then Equation 3.4 is equivalent to Equation 3.3. This equality is apparent in Equation 3.5 because the log of the summation over translation instances will equal zero if there are no per-instance features ϕ.

Cunei constructs a phrase pair by summing over all instances of translation, but the separation described above permits static features to be applied to phrase pairs at any time. Cunei currently includes the SMT-like features described in Table 3.2 which model a translation's overall frequency in the corpus, lexical probability, and coverage. We calculate these features to ensure that Cunei's model has no less discriminative power than a standard SMT system. 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.

3.2.5 Combining Translation Units

Thus far we have constructed phrase pairs that correspond to some part of the input sentence, but in order to generate a complete target hypothesis, Cunei must explore possible combinations and reorderings of these phrase pairs. The decision rule presented in Equation 3.2 dictates which target hypothesis Cunei ultimately selects. To identify this hypothesis, Cunei must explore many possible alignments and score many possible orderings of phrase pairs.

We calculate the score for each phrase pair on demand, but re-scoring the phrase pair for each possible alignment is both expensive and inefficient. Instead, Cunei constructs a lattice of translation units, illustrated in Figure 3.3, as an intermediary data structure. In this lattice Cunei stores lists of phrase pairs and the span of the input sentence to which they correspond. Unlike a phrase table, the lattice structure permits the phrase pairs to correspond to a specific span of the input sentence. The phrase pairs that Cunei stores in the lattice are scored with respect to the input sentence and conditioned on their surrounding source context.

When phrase pairs are selected from the lattice during decoding, Cunei then augments their score with additional features. For example, language model and reordering features cannot be included in the score of a phrase pair in the lattice. This additional information is available only

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

`Translation.Weights.Frequency.Correlation`	$\frac{(c_s - c_t)^2}{(c_s + c_t + 1)^2}$
`Translation.Weights.Frequency.Source`	$-\log(c_s)$
`Translation.Weights.Frequency.Target`	$-\log(c_t)$
`Translation.Weights.Frequency.Count`	$-\log(c_{s,t})$
`Translation.Weights.Frequency.Counts.1`	$\begin{cases} 1 & \text{if } c_{s,t} = 1 \\ 0 & \text{otherwise} \end{cases}$
`Translation.Weights.Frequency.Counts.2`	$\begin{cases} 1 & \text{if } c_{s,t} = 2 \\ 0 & \text{otherwise} \end{cases}$
`Translation.Weights.Frequency.Counts.3`	$\begin{cases} 1 & \text{if } c_{s,t} = 3 \\ 0 & \text{otherwise} \end{cases}$

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

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

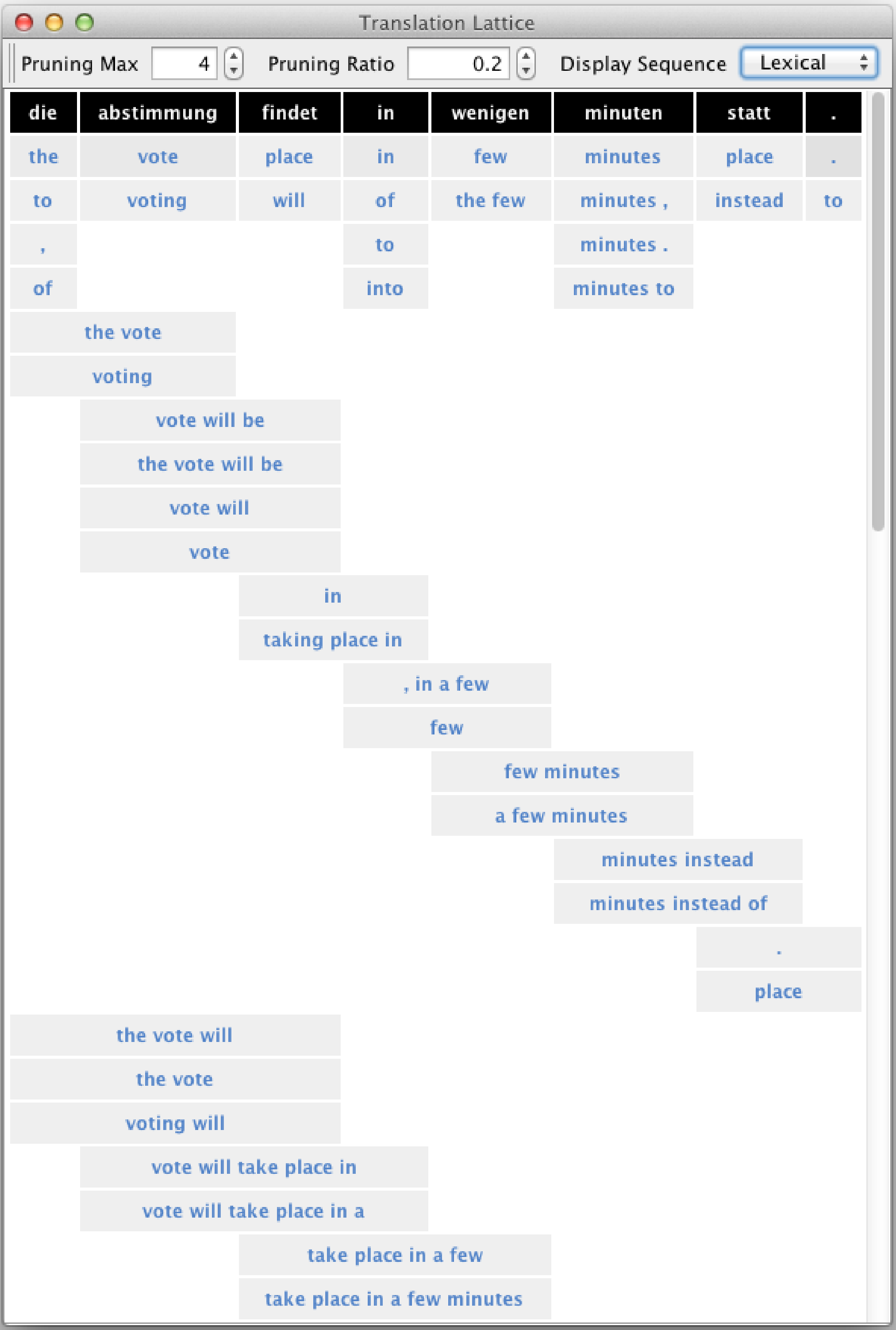
Length Ratios
The mean, μ, and variance, σ^2, of the lengths are calculated over the entire corpus.

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

Coverage
Let $|t|$ be the source length of the phrase pair and $|S|$ be the length of the input sentence.

`Translation.Weights.Spans`	1
`Translation.Weights.Coverage`	$\ln \frac{\vert t\vert}{\vert S\vert}$

Table 3.2: Description of Static SMT-like Features



Pruning Max 4 Pruning Ratio 0.2 Display Sequence Lexical

Figure 3.3: Translation Lattice Visualization

during decoding when Cunei evaluates a specific ordering of phrase pairs from the lattice. The features calculated during decoding are generally similar to the static, SMT-like features presented in Section 3.2.4 in that they are independent of a particular instance of translation. As shown previously in Equation 3.5, the summation over translation instances can be computed separately from such independent features. The construction of the phrase pairs allows Cunei to compute the summation with per-instance features once. After the phrase pairs have been constructed, the instances of translation can be discarded. During decoding, Cunei then applies additional, independent features with a simple linear function.

This separation between the lattice of phrase pairs with per-instance features and decoding with additional independent features was a pragmatic decision, and it represents a limitation of our current implementation. It is not, fundamentally, a restriction of the model we present. In Chapter 7 we will explore the use of per-instance target context features in which this separation is no longer possible and the resulting decoding times are, unfortunately, *much longer*.

The additional features which the decoder calculates when combining phrase pairs are detailed in Table 3.3. In order to compensate for divergences between the source and target language, Cunei may need to reorder the translations. The reordering is modeled by counting the total number of reorderings 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.[5] Any number of language models can be used simultaneously, and each will generate its own set of features. In order to offset the tendency of the language model(s) to prefer short output, we balance the overall score by including a feature that simply counts the number of words present in the target. The complete sentence length is also scored based on the ratio of source words to target words.

Cunei searches over possible alignments that define an ordering of phrase pairs with a statistical decoder. The features described in the previous paragraph score each these permutations of phrase pairs. The decoder uses the standard chart decoding and beam decoding algorithms to construct target hypotheses. This framework is similar to that employed by most phrase-based SMT systems.

Chart decoding generates a lattice of target hypotheses with an algorithm similar to CYK parsing (Chappelier and Rajman, 1998). The bottom-up algorithm incrementally combines target hypotheses and stores the results in a CYK chart. This chart is initialized with entries from the lattice of translation units; each is indexed according to its source position and coverage. Before a translation unit is inserted into the chart, Cunei applies any new decoding features. Cunei then attempts to combine adjacent hypotheses by exploring spans in order from smallest to largest coverage. This method is more commonly used for syntax-based SMT, but there are no grammar rules in Cunei. Instead, two hypotheses are combined by simply swapping positions or occurring in-order. The resulting combination is scored with any new decoding features and placed back in the chart where it may be used anew by the algorithm.

Beam decoding constructs a target hypothesis from left to right (Koehn, 2004a). The algorithm extends an existing target hypothesis by selecting another target hypothesis which represents a region of the input sentence that is not yet translated. For efficiency, the algorithm is restricted to selecting another target hypothesis that occurs within a reordering window (typically 5 words). Target hypotheses are added to a beam based on the number of words covered in the input sentence, but independent of which words are covered. The algorithm is initialized by adding all target hypotheses that cover the beginning of sentence marker to the appropriate beam. The beams are processed in order from smallest to largest coverage. For each hypothesis in a beam, the decoder

[5] 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 Kneser-Ney smoothing using the SRILM toolkit (Stolcke, 2002).

Reordering

Let the first position of the source span for the current partial translation be i and the last position of the source span for the previous partial translation be j.

`Hypothesis.Weights.Reorder.Count`	$\begin{cases} 1 & \text{if } i - j \neq 1 \\ 0 & \text{otherwise} \end{cases}$
`Hypothesis.Weights.Reorder.Distance`	$\lvert i - j - 1 \rvert$

Language Model

Multiple language models can be used; these refer to the model identified as `Default`. Let the order of the language model be denoted by n and the target sequence be represented as $w_0 w_1 w_2 ... w_n$.

`LanguageModel.Default.Weights.Probability`	$\sum_{i=0}^{n} \log P(w_i \mid w_{i-i} w_{i-2} ... w_{i-n+1})$
`LanguageModel.Default.Weights.Unknown`	$\sum_{i=0}^{n} \begin{cases} 1 & \text{if } w_i \text{ is unknown} \\ 0 & \text{otherwise} \end{cases}$

Sentence Length

Let the phrase x contain $|x|_{word}$ words and $|x|_{char}$ characters. The mean, μ, and variance, σ^2, of both word and character lengths are calculated over the corpus.

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

Table 3.3: Description of Decoder Features

attempts to combine it with another hypothesis. The resulting combination is scored with any new decoding features and placed in a larger beam where it may be used anew by the algorithm.

Cunei employs both chart and beam decoding as they exhibit slightly different properties. Within both algorithms Cunei is using the same model to score the target hypotheses, but the two algorithms search different possible reordering patterns. Chart decoding limits reordering to swaps whereas beam decoding limits reordering to a fixed window. In addition, both algorithms require pruning and do not explore their entire hypothesis space. Cunei begins with chart decoding because we expect this algorithm to model local reordering well. After generating a target hypothesis of a typical phrase length, Cunei passes these long phrases to the faster beam decoding. Beam decoding allows for more complicated reordering patterns, albeit with greater pruning.

`Alignment.Epsilon`	`0.01`
`Alignment.Pruning.Maximum`	`5`
`Alignment.Pruning.Minimum`	`2`
`Alignment.Pruning.Ratio`	`5.0`
`Alignment.Pruning.Weight`	`0.35`
`Alignment.Pruning.Window`	`2.0`
`Alignment.Unknown`	`0.15`
`Decoder.Chart.Minimum`	`4`
`Decoder.Chart.Maximum`	`100`
`Decoder.Pruning.Maximum`	`200`
`Decoder.Pruning.Minimum`	`100`
`Decoder.Pruning.Ratio`	`2.0`
`Decoder.Reordering.Window`	`6`
`Decoder.Search.Maximum`	`2000`
`Decoder.Search.Minimum`	`500`
`Decoder.Search.Ratio`	`2.0`
`Matcher.Lookup.Entries`	`150`
`Matcher.Lookup.Sample`	`750`
`Matcher.Sampling`	`UNIFORM`

Table 3.4: Hyperparameters in Cunei and their Default Values

3.3 Summary

In recent years SMT has dominated the field of machine translation research. In Cunei, we extend the SMT model in a novel direction. We integrate instance-based learning in a way that still allows the use of traditional SMT features. The advantage of Cunei is that it also permits features that score the relevance of each translation instance. 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 translation instances present in the training corpus. The development of Cunei has laid the groundwork for this thesis and provides a comprehensive framework for further research and experimentation.

Chapter 4

Learning Model Weights

Given λ we can easily compute the score Cunei's model assigns a phrase pair (see Equation 3.4 on page 35) by iterating over the instances of translation and calculating the requisite features in the distance function. However, learning the optimal parameters for our distance function is not so straightforward. The summation over translation instances is non-parametric, but it employs a parametric distance function. We will refer to the parameters of the distance function as model weights, but these model weights are quite different than those present in the SMT log-linear model. Cunei follows the approach that Smith and Eisner (2006) applied to the traditional SMT model, which uses simulated annealing to minimize expected loss on a development set. However, the added complexity of Cunei's model involves special care which we detail in this chapter. A significant portion of this work was first published in (Phillips and Brown, 2011).

4.1 The Objective Function

In the default configuration, Cunei will learn weights that maximize BLEU (Papineni et al., 2002) on a held-out development set. We use the following objective function that in log-space sums the expected value of BLEU's brevity penalty and precision score:

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

The variables in the objective function are defined as follows:

$$p_i = \frac{e^{\gamma m_i}}{\sum_k e^{\gamma m_k}} \qquad \mu(x) = \sum_i p_i x_i \qquad \sigma(x) = \sum_i p_i (x_i - \mu(x))^2$$

m_i	Log-score of translation hypothesis i in the n-best list
γ	Gamma (used for annealing)
h	Length of the translation hypothesis
r	Length of the selected (shortest or closest) reference
c_n	"Modified count" of matching n-grams according to BLEU
t_n	Total number of n-grams present in the translation hypothesis

Note that the variable m_i, representing the log-score assigned by the model, is the score evaluated by the decision rule in Equation 3.1. It is an unnormalized score, but p_i computes a probability distribution by summing over all translation hypotheses in the n-best list. With the exception of γ, which will be discussed next, the remaining variables relate to the calculation of BLEU. Negating this objective function corresponds to expected loss; in the text below we will refer interchangeably to maximizing the objective function and minimizing the expected loss.

4.2 Minimum Risk Annealing

Cunei follows the approach of Smith and Eisner (2006) and slowly anneals the distribution of the n-best list in order to avoid local minima while minimizing risk (expected loss). The γ parameter is used to modify the probability distribution of the n-best list. The log-score of each translation hypothesis is multiplied by the γ parameter, which has the effect of exponentially increasing or decreasing its likelihood. The idea is to start with a mostly flat distribution and mildly sharpen 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 hypothesis. Within this framework, Cunei maximizes the objective function using conjugate gradient descent to find the optimal set of model weights. A key component of Smith and Eisner (2006) was maximizing entropy prior to minimizing expected loss. The value of γ was allowed to fluctuate, but in order to maximize entropy γ remained small. We did implement this as well but found that it resulted in much longer run times with no significant benefit; it is disabled by default. Instead, Cunei simply initializes $\gamma = 0.25$ and doubles γ upon convergence. The algorithm halts when $\gamma = 4.0$. 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 can be achieved by uniformly increasing the magnitude of all the other weights. To address this weight creep, Cunei augments the objective function with an L2 regularization term. One improvement we made, not discussed by Smith and Eisner (2006), was initializing the n-best list with translations most similar to the references. This is accomplished with a special mechanism in Cunei for oracle decoding (when the references are known to the system). The oracle translations initially have very low scores, but they guide the learning process toward high-scoring, obtainable translations.

The most common technique for learning the model weights in machine translation historically has been minimum error rate training (MERT) (Och, 2003). MERT relies on the fact that one only needs to explore the values of λ that cause the 1-best translation hypothesis to change rank. These specific weights can be easily computed because the standard SMT model is a linear function (see Equation 3.3 on page 34), so modifying λ has a linear effect on the score of a translation hypothesis. Unfortunately, changing λ in Cunei's model will not necessary have a linear effect on its score due to the summation of translation instances (see Equation 3.4 on page 35). The result is that MERT is not stable with our model. Conceptually, Smith and Eisner (2006) is similar to MERT except it uses an objective function that minimizes the expected loss over the distribution of translations present in the entire n-best list. We believe it represents a better learning algorithm than MERT, but more importantly, it does not assume the model takes on a specific form like MERT.

4.3 Taylor Series Approximation

The procedure to maximize the objective function requires us to compute the gradient and re-score the model frequently under a new λ. Storing the features ϕ for *every translation instance* consumes too much memory, and re-translating under *every new* λ consumes too much time.

To address this problem, we approximate the score for each phrase pair $\langle s_i, t_i \rangle$ during training with the second-order Taylor series:

$$m(s_i, t_i, \lambda') \approx m(s_i, t_i, \lambda) + \sum_q (\lambda'_q - \lambda_q) \frac{\partial}{\partial \lambda_q} m(s_i, t_i, \lambda) + \sum_q (\lambda'_q - \lambda_q) \sum_r (\lambda'_r - \lambda_r) \frac{\partial}{\partial \lambda_q \lambda_r} m(s_i, t_i, \lambda)$$

Here we know the score at λ and we are approximating the score at λ'; q and r are, respectively, indices for the weights in λ and λ'. Substituting in the definition of $m(s_i, t_i, \lambda)$ from Equation 3.4 on page 35 yields:

$$m(s_i, t_i, \lambda') \approx \ln \left(\frac{1}{|X|} \sum_{x \in X} e^{\sum_k \lambda_k \cdot \phi_k(s_i, t_i, x)} \right) + \sum_q (\lambda'_q - \lambda_q) E_X[\phi_q(s_i, t_i, x)] + \frac{1}{2} \sum_q (\lambda'_q - \lambda_q) \sum_r (\lambda'_r - \lambda_r) (E_X[\phi_q(s_i, t_i, x) \cdot \phi_r(s_i, t_i, x)] - E_X[\phi_q(s_i, t_i, x)] \cdot E_X[\phi_r(s_i, t_i, x)]) \tag{4.1}$$

Both expectations can be computed efficiently with an online update that analyzes each translation instance once. Formally, the expectation is:

$$E_X[Y] = \sum_{x \in X} Y \cdot P(x \mid s_i, t_i, \lambda)$$
$$P(x \mid s_i, t_i, \lambda) = \frac{e^{\sum_k \lambda_k \phi_k(s_i, t_i, x)}}{\sum_{x'} e^{\sum_k \lambda_k \phi_k(s_i, t_i, x')}}$$

If $\lambda = \lambda'$ then this approximation has no modeling error. We use an approximation during training in order to evaluate the summation over all translation instances at a new λ' without scoring all translation instances. This approximation allows Cunei to explore a local region without needing to re-translate the entire development set. Once the algorithm has converged on a new set of weights, then we do re-translate the development set. If the approximation had significant error, then the convergence criterion no longer holds and the algorithm proceeds to search again for a new set of weights that maximize the objective function.

In Phillips (2010) we used a first-order Taylor series as it was easier to implement and we assumed its approximation was 'close enough'. However, as shown in Table 4.1, the second-order Taylor approximation significantly decreases modeling error. We measured modeling error as the absolute difference in log-score between the approximation and the actual score of each translation hypothesis. The most compelling finding here was that we reduced the variance in error to slightly less than half of that present with the first-order Taylor approximation.

The statistics for Table 4.1 were collected while training a Czech-English system. Initially, when translating the development set, all of the translation hypotheses were scored with λ. During this scoring process, we compute the expectations needed for the Taylor series approximation in Equation 4.1. Each training iteration predicted a new λ'. We recorded for each translation hypothesis the predicted log-score at λ' according the first-order and second-order Taylor approximations that were constructed with expectations computed at λ. Then we compared these approximations to their actual scores by re-translating the development set at λ'. Re-translating the development set involves re-scoring each translation instance under the new λ' and combining these scores as

	Average	Variance
First Order Error	0.1751	0.2893
Second Order Error	0.1202	0.1391
Relative Improvement	*31.36%*	*51.94%*

Table 4.1: Modeling Error and Variance of Taylor Series Approximations

presented in the previous chapter in Equation 3.4. This process was repeated over several training iterations and we collected approximately 20,000 such comparisons between the actual score and the Taylor approximations.

4.4 Combining Multiple Approximations

This training procedure, like most SMT training procedures, involves re-translating a small number of development sentences many times in order to locate the optimal λ. Each time we translate a sentence, we generate an n-best list of possible translations according to the model for the current λ. However, the n-best list contains at most a few hundred entries and it is a very limited perspective of the search space. Thus, it is common practice to merge n-best lists over all iterations. This technique is necessary for stability, but it creates a new problem. Because we approximate the score for translation hypotheses with a second-order Taylor series, we risk learning a λ' that is optimal for the approximation from λ and not for translation hypotheses scored directly at λ'.

To address this issue, we score multiple approximations and interpolate their values. Our second-order Taylor series approximation predicts the score at λ' based on expectations previously calculated at λ (see Equation 4.1). When Cunei re-translates the development set, we can then calculate the expectations for the Taylor series approximation at the new λ'. At this point we can compute a Taylor series approximation with expectations calculated from either λ or λ'. When we need to predict the score for a translation hypothesis at a new λ'', we can then interpolate between these approximations from λ' and λ. The most recently constructed Taylor series approximation is not necessarily the closest approximation for the new λ''. In addition, interpolating between multiple approximations provides added stability to training. The default settings keep track of the four most recent Taylor series approximations.

Figure 4.1 shows how the average modeling error increases as λ', where we evaluate the Taylor series approximation, moves away from λ, where the Taylor series expectations are calculated. We calculate the modeling error as described previously in Section 4.3. The individual data points are numerous and noisy, so we opted to bin the data. The x-axis displays a distance calculation between λ' and λ; the y-axis represents the average modeling error. Each bin is labeled with the range of error it represents, and the bins further from the origin span larger increments due to fewer data points in those regions. Over 70% of the data has a distance less than 10; the last bin from 50-100 represents less than 1% of the data. We spread out the larger bins, but the x-axis could not be represented fully to scale.

The obvious conclusion from Figure 4.1 is that we would prefer to select a Taylor series approximation for which expectations were calculated nearby the λ' currently being scored. As a result, Cunei performs a weighted interpolation of the values from each Taylor series approximation. We weight the values in inverse proportion to the distance between the λ at which the expectations for the Taylor series were calculated and the λ' we want to evaluate. The closer λ is to λ', the more the Taylor series approximation from λ will contributes to the the interpolated score.

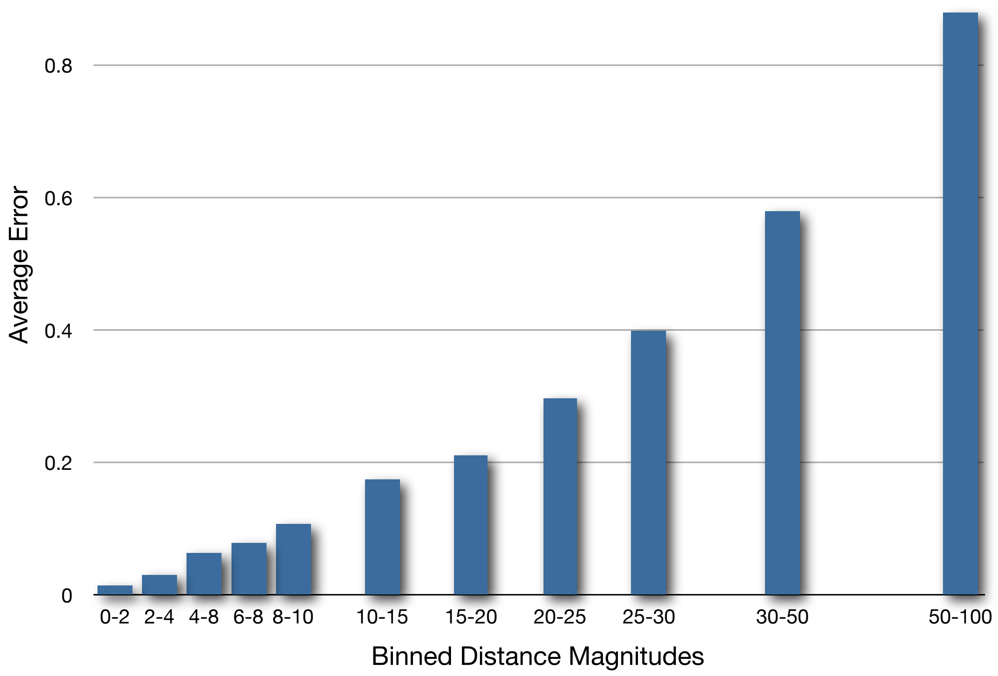


Figure 4.1: Average Modeling Error of Taylor Series Approximation Increases with Distance (x-Axis Not to Scale)

4.5 Summary

Training is a notoriously difficult task in machine translation and it is an even more complex challenge with Cunei's model. The use of BLEU as the objective function results in a very bumpy error surface with many local minima. In addition, Cunei particularly exacerbates the problem of finding the optimal λ as it is not guaranteed to respond linearly to changes in λ. The techniques described in this chapter–using a second-order Taylor series and storing multiple approximations–have made it feasible to learn the parameters of our distance function. While we have referred to these as model weights, they are not the same type of model weights employed within the SMT log-linear model. Nonetheless, we have described how to learn weights that maximize an objective function evaluating end-to-end translation performance. This will enable Cunei to effectively discriminate between translation instances and be competitive with state-of-the-art SMT systems.

Chapter 5

Baseline Evaluation

In this chapter we evaluate translation performance with both Cunei and the popular SMT system, Moses. We initially demonstrate that Cunei has comparable results with Moses when it is restricted to using the same model. We then create a baseline Cunei configuration with instance-based alignment features and show superior results. Recall from the discussion on phrase alignment in Section 3.2.3 of Chapter 3 that Cunei is capable of scoring phrase alignments whereas Moses uses a binary heuristic. Our instance-based approach allows us to add features to the model that score the alignment for each translation instance. These alignment features evaluate the word alignment context of the entire sentence in which a translation instance is situated. They cannot be added to the traditional SMT model because they cannot be computed independent of a specific location in the training data.

5.1 Experiments

We evaluated a total of four systems: two configurations of Moses and two configurations of Cunei. The *Moses Baseline* configuration corresponds to the default settings for Moses[1] which include fourteen features. Of these features, six score lexical reordering and require special handling during decoding that is not (currently) implemented in Cunei.[2] For comparison, we also build a configuration for *Moses without Lexical Reordering* that only includes eight features. *Cunei with Moses Phrase Table* is roughly equivalent to this latter Moses configuration; both of these configurations use the same phrase table as constructed by Moses, and the difference lies only in decoding. Finally, the *Cunei Baseline* configuration scores the phrase alignment of each translation instance on demand instead of loading an external phrase table. The *Cunei Baseline* configuration uses the features that were outlined in Tables 3.1, 3.2, and 3.3 from Chapter 3.

We carried out the experiments on the German-English and Czech-English parallel data released for the 2011 Workshop on Statistical Machine Translation (WMT'11).[3] The corpora for both language pairs included version 6 of the Europarl (Koehn, 2005) and the 2011 edition of parallel news commentary. In addition, the Czech-English parallel corpus included CzEng v0.9 (Bojar and Žabokrtský, 2009). For simplicity, we created a common set of monolingual English text that was shared by the two systems. This monolingual training data totaled 512 million words and consisted of all English text from the parallel corpora (including CzEng v0.9) and web-crawled news text

[1] SVN revision 3880 from February 14, 2011

[2] This is a limitation of our implementation and not Cunei's model. Until recently these features were not enabled by default in Moses and supporting them in Cunei has simply not been a priority.

[3] `http://www.statmt.org/wmt11/`

from 2010-2011 released by WMT'11.

Cunei applied light pre-processing, filtering, and tokenization suitable for Western languages to all corpora. From the parallel corpora, we trained IBM Model-4 word alignments in both directions using GIZA++ (Och and Ney, 2003). From the monolingual data, we trained a 5-gram language model with Kneser-Ney smoothing using SRILM (Stolcke, 2002). All systems were provided an identical tokenized, word-aligned parallel corpus and language model. The Moses training scripts built a standard phrase table and lexicalized reordering table that were used by some of the configurations, as previously described. The Moses systems were trained with MERT and the Cunei systems were trained with the procedure described in Chapter 4.

While WMT'11 and similar competitions have released "standardized" test sets, they usually consist of *only* newswire text and contain minimal contextual or structural information. We will be using the same training and evaluation sets across all of the experiments in this dissertation. In order to better evaluate the role of Cunei's model under varying conditions, we opted to withhold a subset of each parallel corpus and create multi-genre, annotation-rich development and test sets.

The evaluation sets were segmented into development and test sets. The development set is used by each system to learn model weights whereas the test set is held-out for a final evaluation. It is common practice in the machine translation community to report scores on the development set as there is usually a strong correlation between scores on the development set and the test set. However, the development set is not blind and the scores we report on it represent near-optimal system performance. The scores on the test set should be given more consideration as this data is in fact blind.

We evaluated each set of translations using BLEU (Papineni et al., 2002), NIST (Doddington, 2002), Meteor (Banerjee and Lavie, 2005; Denkowski and Lavie, 2010), and TER (Snover et al., 2006) metrics. All metrics compared the tokenized machine translation output to a single, tokenized reference translation. BLEU and NIST were both computed using the official `mteval-v13a.pl` script released by NIST,[4] except we modified it slightly to perform tokenization *only* on whitespace.[5] In addition, we used v1.2 of Meteor[6] and v7.25 of TER.[7] For Meteor we specified that the language was English, which enabled stem, synonym, and paraphrase matching. In all other respects, these metrics were run in their default configuration. Specifically, Meteor and TER require additional flags to normalize the translations which were not enabled. While we would like to see improvement across all metrics, it should be noted that the objective function used during training only computed BLEU.

Due to randomization, multiple training runs often result in different weights. The scores in the results table represent a peak to peak comparison in which we selected the configuration with maximum BLEU score on the development set. To improve robustness, we repeated the training procedure twice for each experiment. Scores identified in bold are statistically significant compared to the baseline over all runs at $p = 0.05$ while the scores in italics are only statistically significant at $p = 0.1$. Our test for statistical significance was an unpaired, two-tailed t-test with unequal variances. We applied this test to every metric score in every experiment, which means that even a value of $p = 0.05$ will likely identify some of the results as statistically significant when they are not.

[4] `http://www.itl.nist.gov/iad/mig/tools/`

[5] This change was necessary as the script (even with the `--international-tokenization` flag enabled) improperly split words containing non-latin characters, such as Czech words passed through untranslated, into multiple tokens. Furthermore, the period in abbreviations and the apostrophe in contractions were also split off as separate tokens. These tokenization issues caused certain categories of words to have greater importance because they were counted as bigrams or trigrams instead of unigrams.

[6] `http://www.cs.cmu.edu/~alavie/METEOR/`

[7] `http://www.cs.umd.edu/~snover/tercom/`

	Czech	English
Vocab	434,196	236,757
Tokens	18,629,039	21,163,940
Sentences	1,658,675	

Table 5.1: Czech-English Parallel Training Resources

	German	English
Vocab	130,141	119,757
Tokens	51,131,683	48,831,583
Sentences	1,782,749	

Table 5.2: German-English Parallel Training Resources

Furthermore, the metric scores from two training runs likely represent too small of a population for the *t*-test. We indicate statistical significance over these training runs in order to provide a sense of which configurations were more reliably better, with the above disclaimer.

We also assessed whether an improvement in metric scores was meaningful or merely reflectively of the particular dataset. For this task we used Koehn's bootstrap resampling method (Koehn, 2004b) which evaluates the statistical significance of BLEU. The idea here is that we can determine if the BLEU score consistently demonstrates improvement by computing it over multiple random samples of translation. BLEU scores identified with † are statistically significant compared to the baseline over 5,000 random samples at $p = 0.05$ while BLEU scores with ‡ are only statistically significant at $p = 0.1$. Note that we have adequate data for this statistical significance test and it is being applied far fewer times (only on the BLEU score).

5.1.1 Czech-English Dataset

As noted above, the Czech-English training data was augmented with the CzEng v0.9 corpus collected by Charles University.[8] This 141 million-word collection includes works of fiction, websites, subtitles, and technical documentation; it significantly diversified the training data. From a held-out portion of CzEng v0.9, we created a 763 sentence development set and 1,506 sentence test set by uniformly sampling across each *genre*. The combined training data for Czech-English totaled 169 million words, which was much larger than necessary for our experiments. There were no technical limitations that prevented us from using the entire training set, but generally a larger corpus correlates with higher memory requirements and more CPU cycles during training. Since we planned numerous experiments, it seemed prudent to reduce the size of the corpus by uniformly sampling one quarter of all training resources. Statistics describing the Czech-English parallel training data are shown in Table 5.1.

5.1.2 German-English Dataset

The German-English parallel training data is less diverse as it is dominated by Europarl proceedings (the newswire commentary is only 7% of the corpus). When including the Europarl corpus in the parallel training data, we followed the common practice of excluding the fourth quarter of 2000. From this held-out portion of the Europarl, we then sampled a 579 sentence development set and 910 sentence test set. The training resources for the German-English system are described in Table 5.2.

5.1.3 Analysis

The results of the evaluation are shown in Tables 5.3 and 5.4. *Cunei with Moses Phrase Table* performs better than *Moses without Lexical Reordering* on the Czech-English test set by 0.28 BLEU,

[8] `http://ufal.mff.cuni.cz/czeng/czeng09/`

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Moses Baseline	28.62	6.528	50.68	56.04	27.09	6.838	49.48	57.04
Moses without Lexical Reordering	28.32‡	6.525	50.62	56.05	27.26	6.823	49.45	57.05
Cunei with Moses Phrase Table	28.07‡	*6.504*	*51.35*	56.12	27.54†	6.875	**50.54**	*56.55*
Cunei Baseline	**31.61**†	**6.772**	**53.09**	**53.26**	**30.76**†	**7.212**	**52.49**	**53.85**

Table 5.3: Evaluation of Moses and Cunei on CzEng v0.9

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Moses Baseline	27.88	6.711	51.52	57.95	25.34	6.609	51.85	59.95
Moses without Lexical Reordering	*27.36*†	6.624	50.90	**58.89**	25.05‡	6.578	51.76	*60.21*
Cunei with Moses Phrase Table	**26.61**†	**6.588**	51.59	**59.14**	*24.83*†	6.582	52.04	*60.25*
Cunei Baseline	*27.50*	6.659	51.78	**58.85**	*25.76*‡	**6.675**	52.13	**59.45**

Table 5.4: Evaluation of Moses and Cunei on German Europarl v6

Scores in bold are statistically significant over multiple runs compared to the *Moses Baseline* at p = 0.05; italics at p = 0.1
BLEU scores with † are statistically significant over random sampling compared to the *Moses Baseline* at p = 0.05; ‡ at p = 0.1

	Czech-English		**German-English**	
	Moses Phrase Table	Cunei Alignment	Moses Phrase-Table	Cunei Alignment
1-gram	95.70%	97.82%	99.49%	99.88%
2-gram	64.28%	72.19%	89.14%	94.68%
3-gram	27.13%	36.75%	58.88%	72.79%
4-gram	10.81%	18.22%	29.24%	44.04%
5-gram	4.78%	11.03%	12.65%	22.19%

Table 5.5: Percent of n-grams in Test Set with at Least One Translation Unit when Using Moses Phrase Table and Cunei Alignment

	Czech-English		**German-English**	
	Moses Phrase Table	Cunei Alignment	Moses Phrase-Table	Cunei Alignment
1-gram	101.04	191.46	134.31	224.71
2-gram	33.52	106.98	71.10	203.19
3-gram	14.80	53.57	29.59	126.44
4-gram	6.00	25.12	13.16	70.71
5-gram	3.09	11.79	8.01	44.77

Table 5.6: Average Number of Translation Units per n-gram in Test Set when Using Moses Phrase Table and Cunei Alignment

but it is 0.22 BLEU worse on the German-English test set. Cunei is, however, obtaining higher Meteor scores in both language pairs (and by a decent margin). Some variation is expected from the different implementations so, overall, this confirms that when the model is held constant, Cunei is at least on par with Moses. Interestingly, including the additional lexical reordering features in Moses had very little effect on the test set scores.

By comparison, BLEU, NIST, Meteor, and TER all showed *Cunei Baseline* outperforming *Cunei with Moses Phrase Table* on the test data. The 0.93 BLEU gain in German-English is only small in comparison to the massive 3.22 BLEU gain in Czech-English. The relative gains according to the four metrics are also fairly consistent within each language pair. We believe the word alignments for the CzEng v0.9 corpus are less accurate due to the variety of genres. Unlike Moses's heuristic that was used to construct the phrase table, Cunei's on-demand phrase alignment does not enforce hard restrictions. The large gain on this corpus with an instance-based model is likely due to the more flexible phrase alignment. Recall, the on-demand phrase alignment will identify phrase pairs with conflicting alignments, albeit with a lower score.

With the exception of *Cunei Baseline* in Czech-English, all of the Cunei configurations have lower BLEU scores on the development set than their respective Moses configuration. This is likely a result of Moses overfitting the development set, which is a common problem with MERT, the training procedure used by Moses. In fact, as discussed in Chapter 4, Cunei's objective function includes a regularization term which could have prevented it from exceeding this score.

In the next section we will highlight a few examples, but in Table 5.5 and 5.6 we provide a broader sense of the difference between using the Moses phrase table and Cunei's on-demand phrase alignment. As shown in Table 5.5, Cunei and Moses both have very good coverage of individual words. However, Moses's coverage of longer phrases drops off more quickly than in Cunei. Cunei is finding translations for nearly twice as many five-word phrases. Even for two-word phrases, Cunei aligns 7.91% more in Czech-English and 5.54% in German-English. In addition to broader coverage, Table 5.6 shows that Cunei's on-demand phrase alignment also provides greater diversity. Cunei provides on average two hundred translations for each individual word, which is twice the number identified in the Moses phrase table. For longer phrases Cunei is providing four to five times as many translations as the Moses phrase table. While many of these have very low scores, additional translations can nonetheless provide more flexibility during decoding. We expect some of this broader coverage and diversity to be noise, but the superior results with Cunei indicate that we are also finding some high-quality translations.

5.1.4 Examples

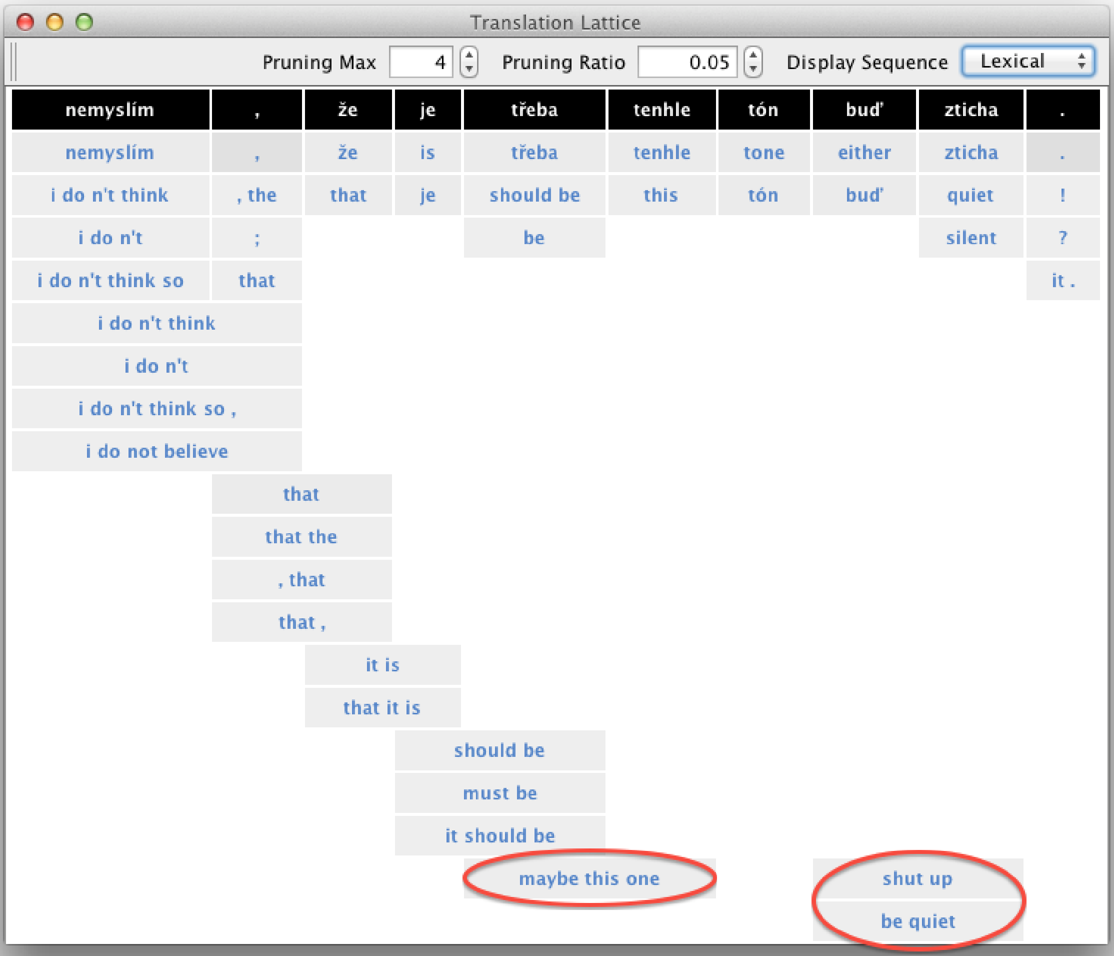
Examples of actual translations from all of the experiments in this dissertation are collected in Appendix A. To simplify the presentation, most of the examples we selected have undergone a single modification. Important changes are marked in bold and any additional aberrations between the sentences are italicized. When ellipses are used, they indicate that section of the output is identical to the baseline system. We will use this notation for examples consistently throughout this dissertation.

Overall, all of the translations are very similar, which is to be expected as they are trained on the same data. The three configurations that use the Moses phrase table have even less variance in their output. In general, the output with the Moses phrase table appears slightly more literal and stilted in comparison to Cunei with on-demand phrase alignment. This behavior can be found in Examples 2, 3, 4, 6, 7, and 10. This difference corresponds to the fact that the instance-based model scores the phrase alignment of each translation instance, which in turn allows for more alternative phrase alignments. The traditional SMT model from the Moses phrase table, on the other hand, would quickly deteriorate if more permissive alignments were allowed because all phrase alignments are counted equally.

A related phenomenon is seen in Examples 1, 5, 9, and 10 where the configurations that use the Moses phrase table either include spurious words or remove a key word. The word alignments are learned statistically and known to have errors. Moses's phrase alignment heuristic will sometimes only be able to identify a phrase pair that includes an incorrectly aligned word or a phrase pair that does not extend far enough to include a valid but unaligned word. This can also occur in Cunei, but it is less likely that word alignment errors will cause problems because Cunei scores multiple possible phrase alignments. Cunei's on-demand alignment will identify and score phrase pairs that include additional words as well as those with too few words. If there exists a spurious or missing word, then the phrase pair's score will reflect this in Cunei but not in Moses.

The translation lattices generated by *Cunei with Moses Phrase Table* and *Cunei Baseline* for Example 1 in Appendix A are illustrated, respectively, in Figures 5.1 and 5.2. In the lattice diagrams, translations within a span are ranked according to their score with translations closer to the top of the lattice being more likely. In Table 5.7 we further highlight some of the phrase pairs with their corresponding per-instance alignment features. Recall, the calculations for the alignment features were previously presented in Table 3.1 on page 43.

The translation in Example 1 from *Cunei Baseline* more closely matches the reference by translating the Czech phrase "buď ztícha" as "be quiet" instead of "shut up". By comparing the feature values for these phrase pairs in Table 5.7, we find that there is a large difference between scoring the phrase alignment with the word alignment score in the direction of the source, α_S, and in the direction of the target, α_T (see Section 3.2.3 in Chapter 3 for the description of α_S and α_T). This difference is most prominent for the features that score unknown word alignment links, where a greater value indicates a worse alignment. The translation "be quiet" has a high penalty for



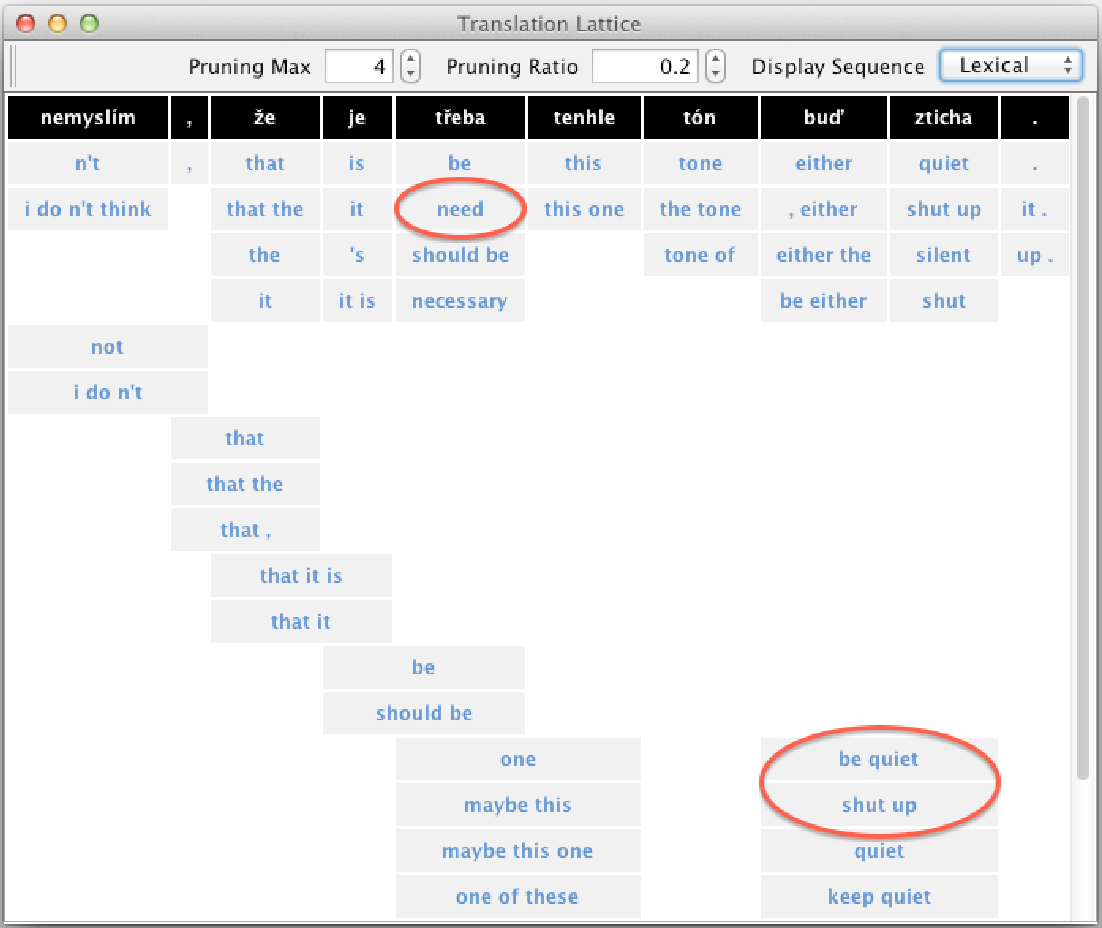
Pruning Max 4 Pruning Ratio 0.05 Display Sequence Lexical

nemyslím	,	že	je	třeba	tenhle	tón	buď	zticha	.
nemyslím	,	že	is	třeba	tenhle	tone	either	zticha	.
i do n't think	, the	that	je	should be	this	tón	buď	quiet	!
i do n't	;			be				silent	?
i do n't think so	that								it .
i do n't think									
i do n't									
i do n't think so ,									
i do not believe									
	that								
	that the								
	, that								
	that ,								
		it is							
		that it is							
			should be						
			must be						
			it should be						
				maybe this one					
							shut up		
							be quiet		

Figure 5.1: Translation Lattice for Example 1 with Moses Phrase Table

unknown word alignments in the target direction, but has very good scores for all other features. The translation “shut up” has has a penalty for unknown word alignments in both directions; in comparison to the feature values for “be quiet”, the source direction is better and the target direction is worse. By employing multiple, per-instance alignment features, Cunei can learn model weights during training that prefer one direction of the word alignments over the other. In this case, that resulted in changing the rank between “shut up” and “be quiet”.

We also find in Example 1 that *Cunei with Moses Phrase Table* identifies and selects a translation for “třeba tenhle”. While phrase pairs do not have to represent linguistic constituents, this is still an odd phrase to use, and all of the translations for this phrase pair lack the word “need” (corresponding to “třeba”). *Cunei Baseline* does include translations for this phrase as well. In fact, Figure 5.2 has more possible phrase pairs for “třeba tenhle” because the on-demand phrase alignment is scoring phrase pairs that exclude some words in an attempt to find one with a higher score. However, none of these phrase pairs are good candidates for translation because the feature values for unknown word alignments are consistently high. Instead, *Cunei Baseline* selects the



Translation Lattice

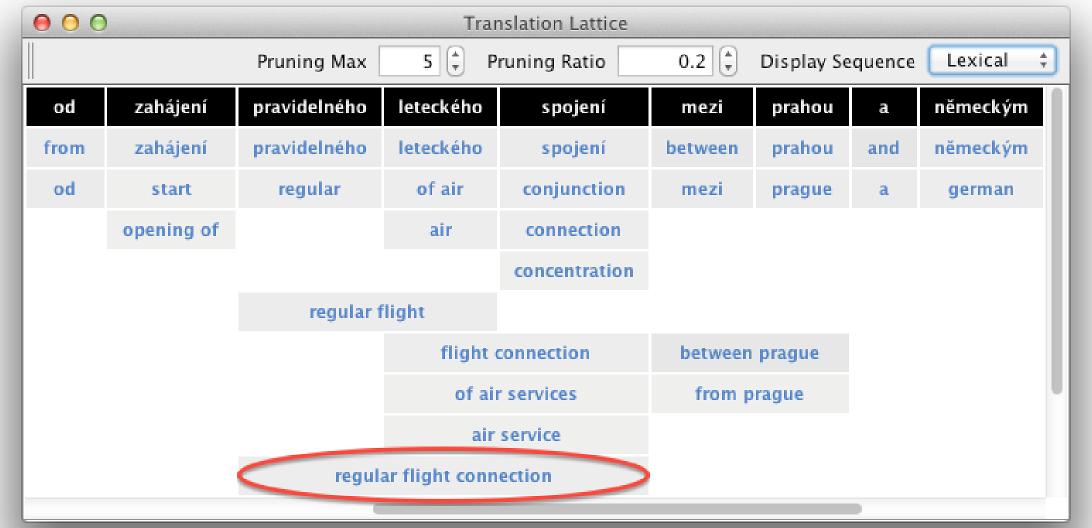
Pruning Max 4 Pruning Ratio 0.2 Display Sequence Lexical

nemyslím	,	že	je	třeba	tenhle	tón	buď	zticha	.
n't	,	that	is	be	this	tone	either	quiet	.
i do n't think		that the	it	need	this one	the tone	, either	shut up	it .
		the	's	should be		tone of	either the	silent	up .
		it	it is	necessary			be either	shut	
not									
i do n't									
	that								
	that the								
	that ,								
		that it is							
		that it							
			be						
			should be						
				one			be quiet		
				maybe this			shut up		
				maybe this one			quiet		
				one of these			keep quiet		

Figure 5.2: Translation Lattice for Example 1 with Cunei Baseline

correct translation “need” from the single word “třeba”, which the alignment features ranked three positions higher, as shown in Table 5.7.

Let us now turn to Figures 5.3 and 5.4, which present limited selections of the translation lattice for Example 4. The first noticeable difference is that *Cunei Baseline* finds phrase pairs for “zahájení pravidelného leteckého” which are not present in the Moses phrase table. These are high-quality translations that illustrate improved coverage with per-instance phrase alignment. In the translation of Example 4, *Cunei with Moses Phrase Table* incorrectly selects “flight connection” instead of “air service”. Note that the longest phrase in the Moses phrase table is “pravidelného leteckého spojení” and it is only translated as “regular flight connection”. As shown in Figure 5.4 and Table 5.8, Cunei finds alternative translations for this phrase. However, the phrase pairs that include the text “air service” still do not rank as high as “regular flight connection” due to conflicting outside word alignment links. Nonetheless, *Cunei Baseline* is able to select the correct translation, “air service”, but it uses the translation for the two-word phrase “leteckého spojení”, which now ranks higher than in the Moses phrase table. This phrase pair is not the highest ranked translation; the alignment features are better in one direction than the other. However, the

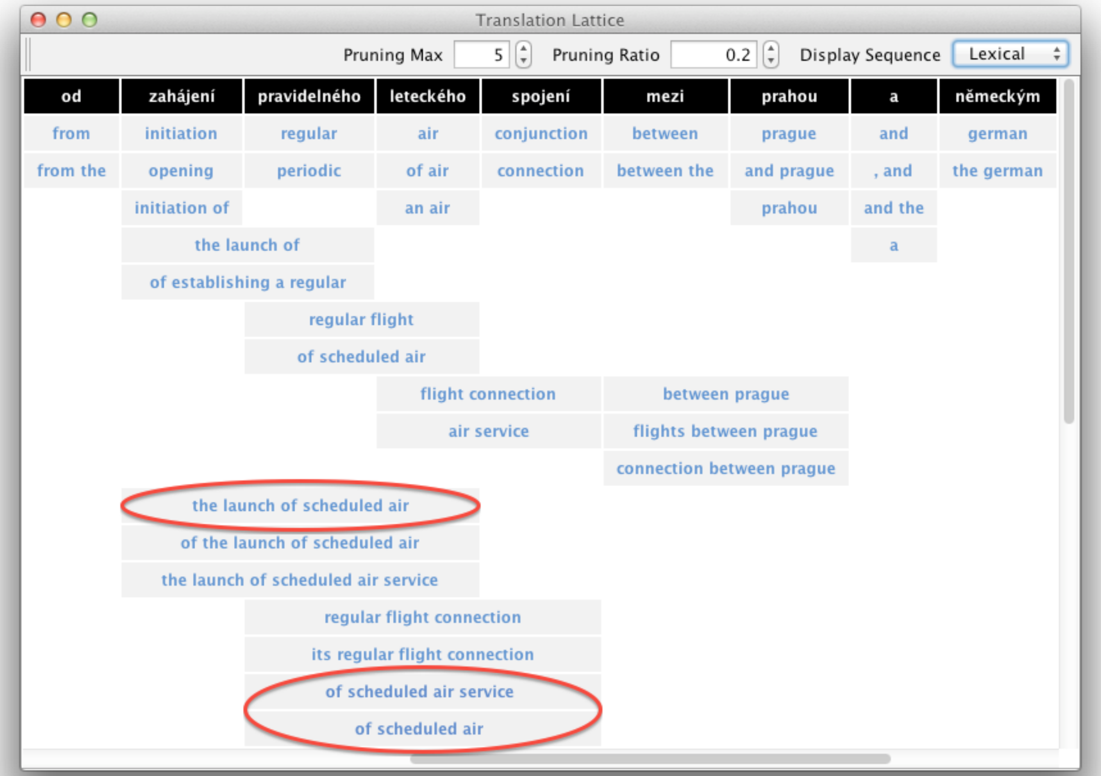


Translation Lattice

Pruning Max 5 Pruning Ratio 0.2 Display Sequence Lexical

od	zahájení	pravidelného	leteckého	spojení	mezi	prahou	a	německým
from	zahájení	pravidelného	leteckého	spojení	between	prahou	and	německým
od	start	regular	of air	conjunction	mezi	prague	a	german
	opening of		air	connection				
				concentration				
		regular flight						
			flight connection		between prague			
			of air services		from prague			
			air service					
		regular flight connection						

Figure 5.3: Translation Lattice for Example 4 with Moses Phrase Table



Translation Lattice

Pruning Max 5 Pruning Ratio 0.2 Display Sequence Lexical

od	zahájení	pravidelného	leteckého	spojení	mezi	prahou	a	německým
from	initiation	regular	air	conjunction	between	prague	and	german
from the	opening	periodic	of air	connection	between the	and prague	, and	the german
	initiation of		an air			prahou	and the	
	the launch of						a	
	of establishing a regular							
		regular flight						
		of scheduled air						
			flight connection		between prague			
			air service		flights between prague			
					connection between prague			
	the launch of scheduled air							
	of the launch of scheduled air							
	the launch of scheduled air service							
		regular flight connection						
		its regular flight connection						
		of scheduled air service						
		of scheduled air						

Figure 5.4: Translation Lattice for Example 4 with Cunei Baseline

	Original	**Inside**		**Outside**		**Unknown**	
Target Phrase	**Rank**	**Source**	**Target**	**Source**	**Target**	**Source**	**Target**
Source Phrase: třeba							
be	3	-1.0764	-1.2609	0.0000	-0.1011	0.0143	0.2872
need	5	-0.1230	0.0000	0.0000	0.0000	0.1179	0.1545
should be	2	0.0000	-2.3561	0.0000	-0.6049	0.0000	0.6552
necessary	11	-0.1862	0.0000	0.0000	0.0000	0.2499	0.2101
need to	6	0.0000	-0.0184	-0.0389	0.0000	0.0001	0.5600
must	12	-0.2890	0.0000	0.0000	0.0000	0.1383	0.3654
Source Phrase: třeba tenhle							
one	N/A	-0.8777	0.0000	0.0000	-0.2074	1.7776	1.3217
maybe this	N/A	-1.4421	0.0000	0.0000	0.0000	3.2280	4.4716
maybe this one	1	0.0000	0.0000	0.0000	0.0000	3.2280	9.0114
one of these	N/A	0.0000	0.0000	0.0000	-0.2074	1.7776	4.9508
Source Phrase: buď zticha							
be quiet	2	-0.0430	0.0000	0.0000	0.0000	0.0000	3.0317
shut up	1	-0.0198	-0.0180	0.0000	-0.0281	1.2602	1.9111
quiet	N/A	-2.0760	-1.0640	0.0000	-0.0101	1.6203	0.2119
keep quiet	3	0.0000	-1.2155	0.0000	0.0000	1.4225	1.7835
shut	N/A	-4.3599	-0.0185	0.0000	-1.3335	1.3214	0.6604

Table 5.7: Selected Alignment Features for Phrase Pairs in Example 1

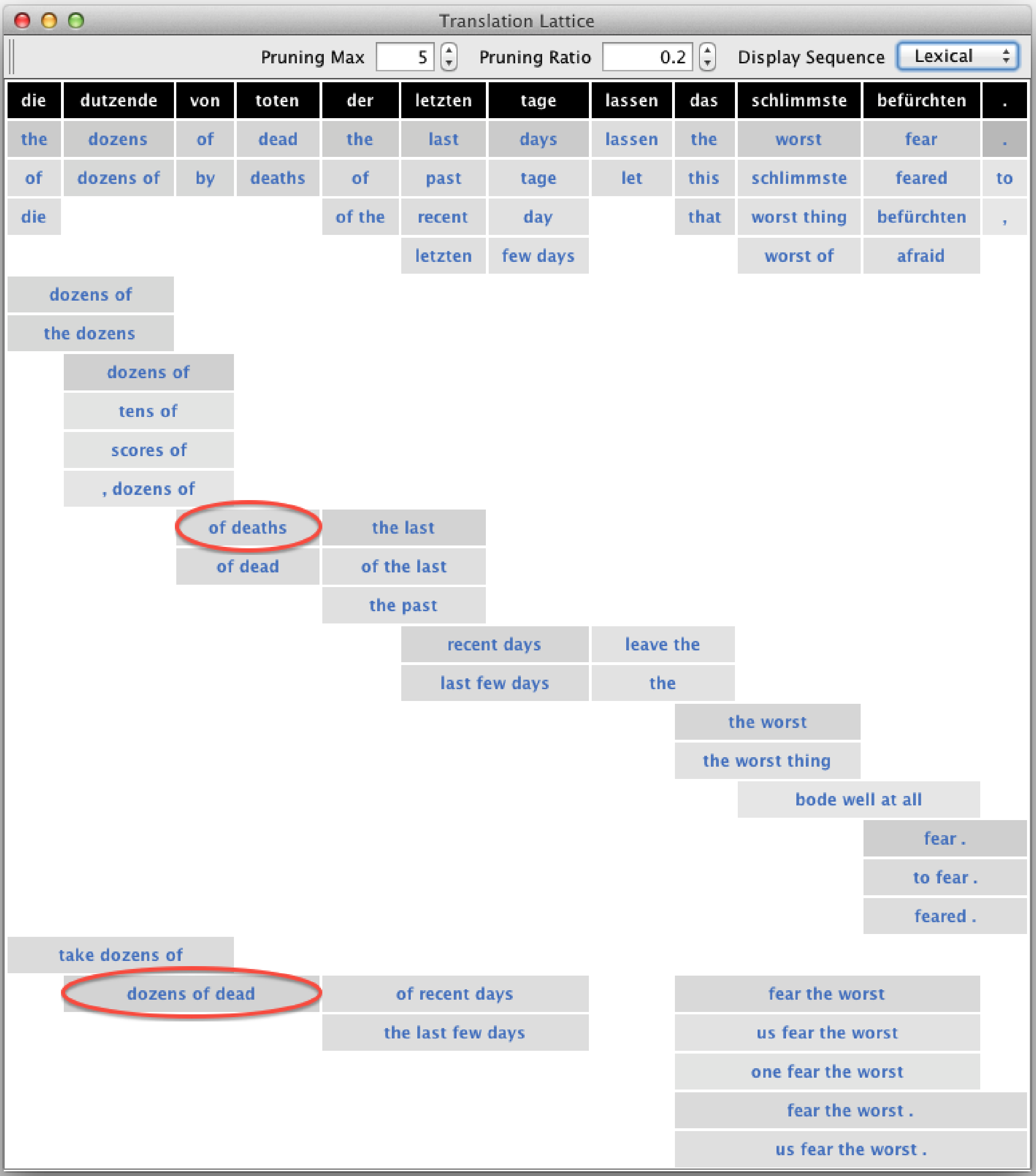
phrase pairs in Table 5.8 for "pravidelného leteckého spojení" and "leteckého spojení" all have high unknown word alignment feature values, which diminishes the difference between their scores. The rank of the translation "air service" is now high enough that when combined with other features during decoding, such as the language model score, it is selected.

Last, we analyze the translations in Example 7. The corresponding translation lattices in Figures 5.5 and 5.6 provide a sharp contrast with one another. The lattice from the Moses phrase table is much more sparse; it does not have nearly the same coverage of two and three-word phrases. The difference in translation stems from the phrase pairs for "von toten" and "dutzende von toten". Both systems (correctly) translate "von toten" as "of deaths", but the Moses phrase table only provides "dozens of dead" as the (incorrect) translation for "dutzende von toten". The problem is that decoding generally prefers longer phrases and "dozens of dead" is being selected instead of the independent translations for "dutzende" as "dozens" and "von toten" as "of deaths". *Cunei Baseline* offers a much larger selection of phrase pairs for "dutzende von toten" as shown in Table 5.9. Many of these are highly unlikely and some even have inside word alignment scores less than -50. However, the more flexible per-instance alignment captures the correct translation "dozens of deaths". The feature values indicate it is from a slightly worse alignment than "dozens of dead", but it is the second-highest ranked candidate overall.

Adding many alternate translations comes at the cost of increased risk. Example 8 illustrates more fluent output from *Cunei Baseline* due to alternative phrasing. However, in this example some of the information is lost as well. Cunei's output with on-demand phrase alignment is not always better, but according to the metric scores it is better on average.

Target Phrase	**Original Rank**	**Outside Source**	**Outside Target**	**Unknown Source**	**Unknown Target**
Source Phrase: pravidelného					
regular	2	0.0000	0.0000	0.0000	0.3308
periodic	3	0.0000	0.0000	0.0000	0.2262
the regular	7	0.0000	0.0000	0.0000	1.0146
a regular	4	-1.3278	0.0000	0.0000	0.3952
Source Phrase: pravidelného leteckého spojení					
regular flight connection	1	0.0000	0.0000	0.4215	0.7478
its regular flight connection	N/A	-0.7891	0.0000	0.4215	1.1584
of scheduled air service	N/A	-1.8034	-3.0855	0.5877	1.5501
of scheduled air	N/A	-1.8034	-3.0855	0.5877	1.1394
launch of scheduled air service	N/A	-4.0053	-3.0855	0.5877	1.6573
Source Phrase: leteckého					
air	3	0.0000	-0.1088	0.0000	0.3783
Source Phrase: leteckého spojení					
flight connection	1	0.0000	0.0000	0.4215	0.4498
air service	3	0.0000	-1.9479	0.4430	0.7450
the	N/A	0.0000	-0.3090	1.7442	0.7450
scheduled air service	N/A	-1.8034	-1.9479	0.4430	1.1394
connection	N/A	0.0000	-6.9315	0.4215	0.0936
the čsa	N/A	-24.3390	0.0000	1.7442	0.7450
regular flight connection	N/A	-38.9483	0.0000	0.4215	0.7478
flight connection between	N/A	-44.0595	0.0000	0.4215	0.4498
of air services	2	0.0000	-25.3211	0.2661	0.7152
the čsa service	N/A	-31.2705	0.0000	1.7442	1.5501
air services	N/A	0.0000	-25.3211	0.2661	0.4478
Source Phrase: spojení					
conjunction	2	0.0000	0.0000	0.0000	0.0000
connection	3	0.0000	0.0000	0.0000	0.2977
concentration	4	0.0000	0.0000	0.0000	0.0000

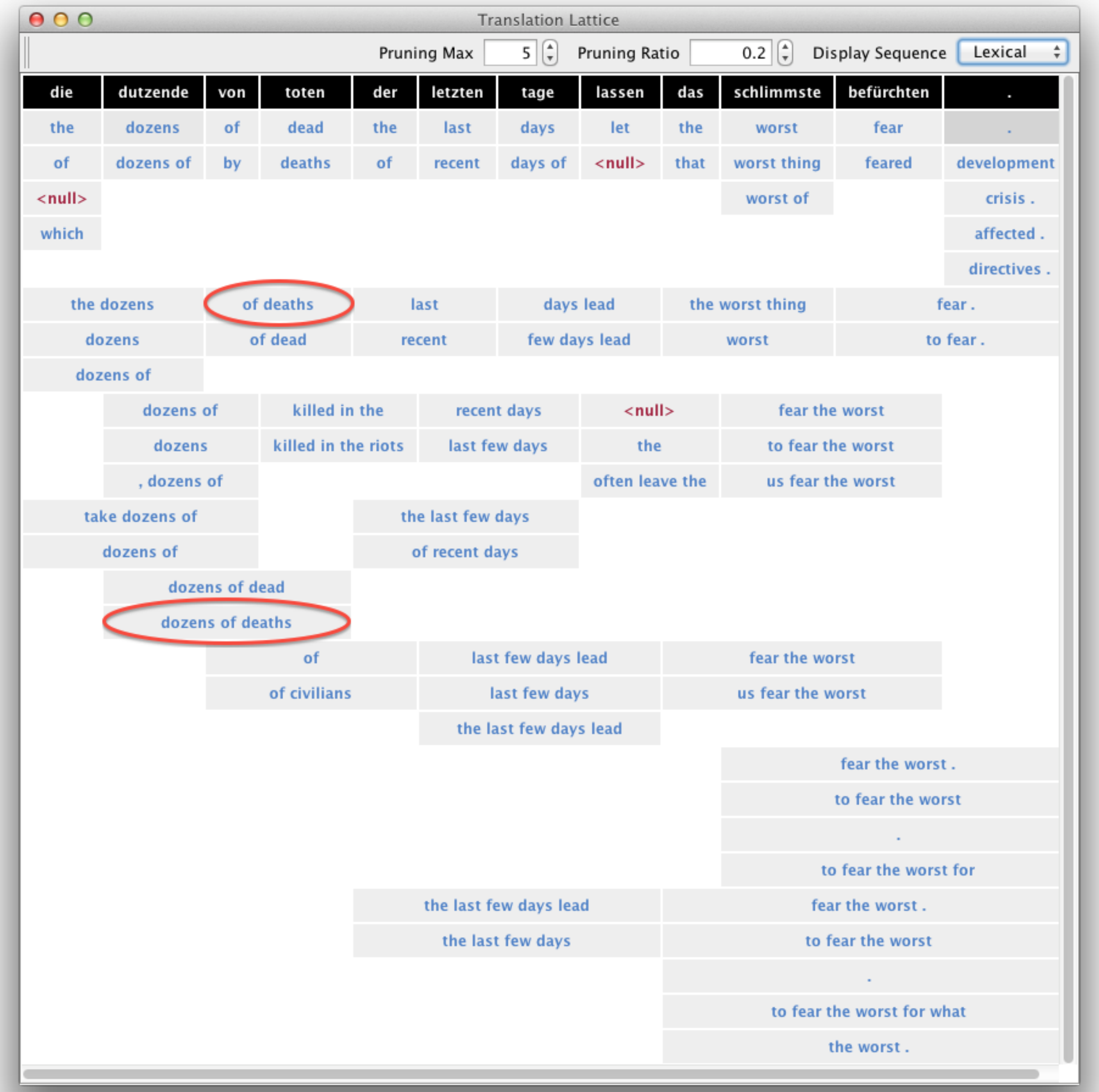
Table 5.8: Selected Alignment Features for Phrase Pairs in Example 4



Pruning Max 5 Pruning Ratio 0.2 Display Sequence Lexical

die	dutzende	von	toten	der	letzten	tage	lassen	das	schlimmste	befürchten	.
the	dozens	of	dead	the	last	days	lassen	the	worst	fear	.
of	dozens of	by	deaths	of	past	tage	let	this	schlimmste	feared	to
die				of the	recent	day		that	worst thing	befürchten	,
					letzten	few days			worst of	afraid	
dozens of											
the dozens											
	dozens of										
	tens of										
	scores of										
	, dozens of										
		of deaths		the last							
		of dead		of the last							
				the past							
					recent days		leave the				
					last few days		the				
								the worst			
								the worst thing			
									bode well at all		
										fear .	
										to fear .	
										feared .	
take dozens of											
	dozens of dead			of recent days				fear the worst			
				the last few days				us fear the worst			
								one fear the worst			
								fear the worst .			
								us fear the worst .			

Figure 5.5: Translation Lattice for Example 7 with Moses Phrase Table


Translation Lattice

Pruning Max 5 Pruning Ratio 0.2 Display Sequence Lexical

die	dutzende	von	toten	der	letzten	tage	lassen	das	schlimmste	befürchten	.
the	dozens	of	dead	the	last	days	let	the	worst	fear	.
of	dozens of	by	deaths	of	recent	days of	<null>	that	worst thing	feared	development
<null>									worst of		crisis .
which											affected .
											directives .
the dozens		of deaths		last		days lead		the worst thing		fear .	
dozens		of dead		recent		few days lead		worst		to fear .	
dozens of											
	dozens of		killed in the		recent days		<null>		fear the worst		
	dozens		killed in the riots		last few days		the		to fear the worst		
	, dozens of						often leave the		us fear the worst		
take dozens of				the last few days							
dozens of				of recent days							
	dozens of dead										
	dozens of deaths										
		of			last few days lead			fear the worst			
		of civilians			last few days			us fear the worst			
					the last few days lead						
									fear the worst .		
									to fear the worst		
									.		
									to fear the worst for		
				the last few days lead				fear the worst .			
				the last few days				to fear the worst			
								.			
								to fear the worst for what			
								the worst .			

Figure 5.6: Translation Lattice for Example 7 with Cunei Baseline

	Original	Inside		Unknown	
Target Phrase	**Rank**	**Source**	**Target**	**Source**	**Target**
Source Phrase: dutzende					
dozens	0	-0.0302	-0.0230	0.0000	0.0000
Source Phrase: dutzende von toten					
dozens of dead	0	0.0000	0.0000	0.0000	0.0643
dozens of deaths	N/A	-0.3457	-0.9151	0.1744	0.0959
dozens of fatalities	N/A	0.0000	-0.9066	0.3416	0.0864
to dozens of fatalities	N/A	0.0000	-0.9066	0.3416	0.5686
causing dozens of deaths	N/A	0.0000	-7.8466	0.1744	0.3117
also dozens of dead	N/A	0.0000	-42.5837	0.0000	0.0643
are also dozens of dead	N/A	0.0000	-75.8012	0.0000	0.0643
dozens have died	N/A	0.0000	0.0000	1.1711	0.9953
dozens	N/A	-9.0664	0.0000	1.1711	0.0000
and dozens have died	N/A	0.0000	-45.2961	1.1711	0.9953
dozens have	N/A	-9.0664	0.0000	1.1711	0.5037
Source Phrase: von					
of	0	-0.0081	0.0000	0.1287	0.0481
Source Phrase: von toten					
of deaths	0	-0.0166	-0.1298	0.0761	0.1224
of dead	1	-0.0485	-0.0494	0.0174	0.0479
Source Phrase: toten					
dead	0	-0.0843	-0.3507	0.0092	0.0000
deaths	1	-0.0934	-0.2884	0.0000	0.0946
the dead	22	0.0000	-16.1434	0.0291	0.1668
of deaths	6	-0.2061	-13.2154	0.0000	0.2815
fatalities	5	-0.2542	-4.2575	0.1792	0.0076
of dead	8	-0.5150	-11.3851	0.0000	0.2201

Table 5.9: Selected Alignment Features for Phrase Pairs in Example 7

5.1.5 Runtime

A runtime analysis of all four systems is presented in Table 5.10. The timings we report are averages over multiple runs. They were executed on multiple machines in a cluster, but all of these machines have dual quad-core 2.33GHz CPUs with 32GB of RAM. We translated the test set six times (on one machine) and averaged the time from the last five runs. We tossed the first run as it usually required additional time waiting on the operating system to swap in pages from disk (due to memory-mapping). For this experiment both systems were explicitly instructed to run with one thread, although both Moses and Cunei do support multiple threads.

Cunei is written in Java, which performs automatic memory management. Prior to execution, the user specifies how much memory the Java Virtual Machine (JVM) is permitted to allocate. If the user specifies a very large amount of memory, then the JVM will rarely bother freeing unused

	Czech-English		German-English	
	Runtime in Minutes	Words per Minute	Runtime in Minutes	Words per Minute
Moses Baseline	16.18	1,141	35.40	754
Moses without Lexical Reordering	11.00	1,678	26.34	1014
Cunei with Moses Phrase Table	20.36	906	55.45	482
Cunei Baseline	36.95	500	106.13	252

Table 5.10: Runtime on Test Sets of Moses and Cunei

objects and allows the memory usage to grow to the limit. If the user specifies too little memory, then the JVM will spend all of its time trying to identify objects that can be freed and runtime suffers. As a result, it is difficult to determine how much memory Java programs require for 'normal' execution. We observed that Moses typically used a little less than 2.5GB to translate these test sets. For the timings with Cunei, we specified that the JVM could allocate 5GB of memory.

Overall, the runtime speed of Cunei is lacking in comparison to Moses. When both Cunei and Moses use the same model from an external phrase table, Cunei takes approximately twice as long as Moses to translate a test set. Either Cunei is exploring a larger search space than necessary or it is not implemented as efficiently as Moses. The on-demand phrase alignment slows performance down even further, but this is to be expected from instance-based learning. Overall, Cunei's speed is still acceptable, but faster would be preferable.

5.2 Summary

In this chapter we showed that when the model is held constant and both Cunei and Moses use the same phrase table, they perform similarly. We then found that when Cunei leveraged a simple instance-based model, it outperformed the traditional SMT model. In particular, the only instance-specific features we added to the model were those related to phrase alignment. This difference in model accounted for a 3.22 BLEU gain on the CzEng v0.9 corpus which contains multiple genres. The less varied German-English dataset also showed a gain of 0.93 BLEU. In addition, the examples we presented highlight how the on-demand phrase alignment improves translation quality by permitting more alternative phrase alignments with per-instance features. Overall, the translations produced by *Cunei Baseline*, while not dramatically different, are consistently an improvement over those from either *Moses Baseline* or *Cunei with Moses Phrase Table*.

We will extend Cunei in the next chapters by adding more instance-specific features to the model. The simple Cunei configuration with on-demand phrase alignment presented in this chapter will form the baseline against which we compare all future work. In addition, all following experiments will be trained and evaluated with the same Czech-English and German-English corpora described in this chapter.

Chapter 6

Incorporating Source Similarity

As illustrated originally in Figures 1.1 and 1.2 from the introduction, the distance function that scores each translation instance can be conceptually broken down into three stages: $f_S(s, x)$, $f_X(x)$, and $f_T(x, t)$. The baseline system with per-instance phrase alignment described in Chapter 5 was limited to features representing $f_X(x)$, which signify whether the phrase pair is a good correspondence. In this chapter, we will expand on the sources of information available to the model by incorporating features that represent $f_S(s, x)$, the source similarity. Specifically, we will focus on the role of context in comparing the phrase s in the input sentence to the translation instance x.

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. In this chapter, we will exploit non-local information that is embedded within the surrounding context of each translation instance but not represented by a standard phrase pair. Cunei's approach extends well to this situation as its model enables the score of a phrase pair to be influenced more heavily by select instances of translation.

Adding features that score contextual similarity is similar to the work of (Chan et al., 2007), (Carpuat and Wu, 2007), and (Gimpel and Smith, 2008) as discussed in Section 2.3.1 of Chapter 2. While the incorporation of sentence-level annotations is unique to our work, many of the features that score the surrounding tokens are, in fact, inspired by this earlier work. The crucial difference is that that Cunei employs instance-based learning and compares the contextual similarity of each translation instance to the input sentence, whereas the previous work uses a traditional SMT model and creates more specialized phrase pairs from the additional source context.

To clarify this distinction, consider a simple SMT model with two feature functions: $P(s|t)$ and $P(t|s)$. Borrowing the notation from Section 3.1.1 of Chapter 3, we specify the function for scoring a phrase pair as:

$$m(s, t) = \lambda_1 P(s|t) + \lambda_2 P(t|s)$$

Typically, we estimate these probabilities with relative frequencies. The calculation is performed by iterating over all translation instances and counting which ones share the same source phrase and target phrase as the candidate phrase pair. The same function for scoring a phrase pair written with instance-based notation that includes the relative frequency calculations is:

$$m(s, t) = \lambda_1 \frac{\sum_{x \in X} \delta(s = S(x) \wedge t = T(x))}{\sum_{x \in X} \delta(t = T(x))} + \lambda_2 \frac{\sum_{x \in X} \delta(s = S(x) \wedge t = T(x))}{\sum_{x \in X} \delta(s = T(s))}$$

The delta function returns one when the condition within it is true and zero otherwise. Here the functions $S(x)$ and $T(x)$ indicate, respectively, the source phrase and target phrase of the translation instance x. Now consider that the input sentence is in genre g and has sentential context c. Let us also define the functions $G(x)$ and $C(x)$ to, respectively, identify the genre and sentence context of the translation instance x. The traditional SMT model can be extended to score more specialized phrase pairs that condition the score on g and c:

$$\begin{aligned} m(s,t|g,c) &= \lambda_1 P(s|t,g,c) + \lambda_2 P(t|s,g,c) \\ &= \lambda_1 \frac{\sum_{x\in X} \delta(s = S(x) \wedge t = T(x) \wedge g = G(x) \wedge c = C(x))}{\sum_{x\in X} \delta(s = S(x) \wedge g = G(x) \wedge c = C(x))} \\ &\quad + \lambda_2 \frac{\sum_{x\in X} \delta(s = S(x) \wedge t = T(x) \wedge g = G(x) \wedge c = C(x))}{\sum_{x\in X} \delta(t = T(x) \wedge g = G(x) \wedge c = C(x))} \end{aligned}$$

In the above equation, if the information that the phrase pair is conditioned on is very specific, then there will be very few occurrences of translation available for computing the relative frequencies. Our instance-based approach, instead, uses similarity features to compare each instance of translation with the input sentence. Given a similarity function for the genre, $S_G(g,x)$ and sentential context $S_C(c,x)$, the score for the phrase pair with our instance-based model could be represented as:

$$\begin{aligned} m(s,t) &= \ln\left(\frac{1}{|X|}\sum_{x\in X} e^{\lambda_1 P(s|t) + \lambda_2 P(t|s) + \lambda_3 S_G(g,x) + \lambda_4 S_C(c,x)}\right) \\ &= \lambda_1 P(s|t) + \lambda_2 P(t|s) + \ln\left(\frac{1}{|X|}\sum_{x\in X} e^{\lambda_3 S_G(g,x) + \lambda_4 S_C(c,x)}\right) \end{aligned}$$

Note that *all translation instances* are used in this approach to compute the score for phrase pair, not just those that have exactly the same genre and context. Instead of employing specialized, discriminative features, our approach uses simple similarity features. We also include $P(s|t)$ and $P(t|s)$ as features to highlight that any per-instance similarity features can be scored *in addition* to traditional SMT features.

If the corpus is relatively homogenous and composed of high-quality, in-domain translations, then this distinction may not be important. However, consider the situation in which we are provided out-of-domain corpora or low-quality comparable corpora in addition to high-quality, in-domain text. The best way to incorporate these additional corpora is not obvious. When dealing with data of varying quality, estimating the SMT model with relative frequencies over all of the data often degrades translation quality.

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. We discussed approaches like this in Section 2.3.5 of Chapter 2 and mentioned specifically the work of (Hildebrand et al., 2005), (Lu et al., 2007) and (Matsoukas et al., 2009). As described previously, these approaches filter or weight sentences or documents in the training data. The effect of modifying the training data is that it adjusts the counts for the relative frequencies, such as $P(s|t)$ and $P(t|s)$ in the equations above. This prior work did not assess context internal to a sentence as weights were applied to the whole sentence.

For the experiments presented in this chapter, we do not have clearly delineated in-domain and out-of-domain corpora. Instead–as is often the case–we have a large amalgam of texts. Within this

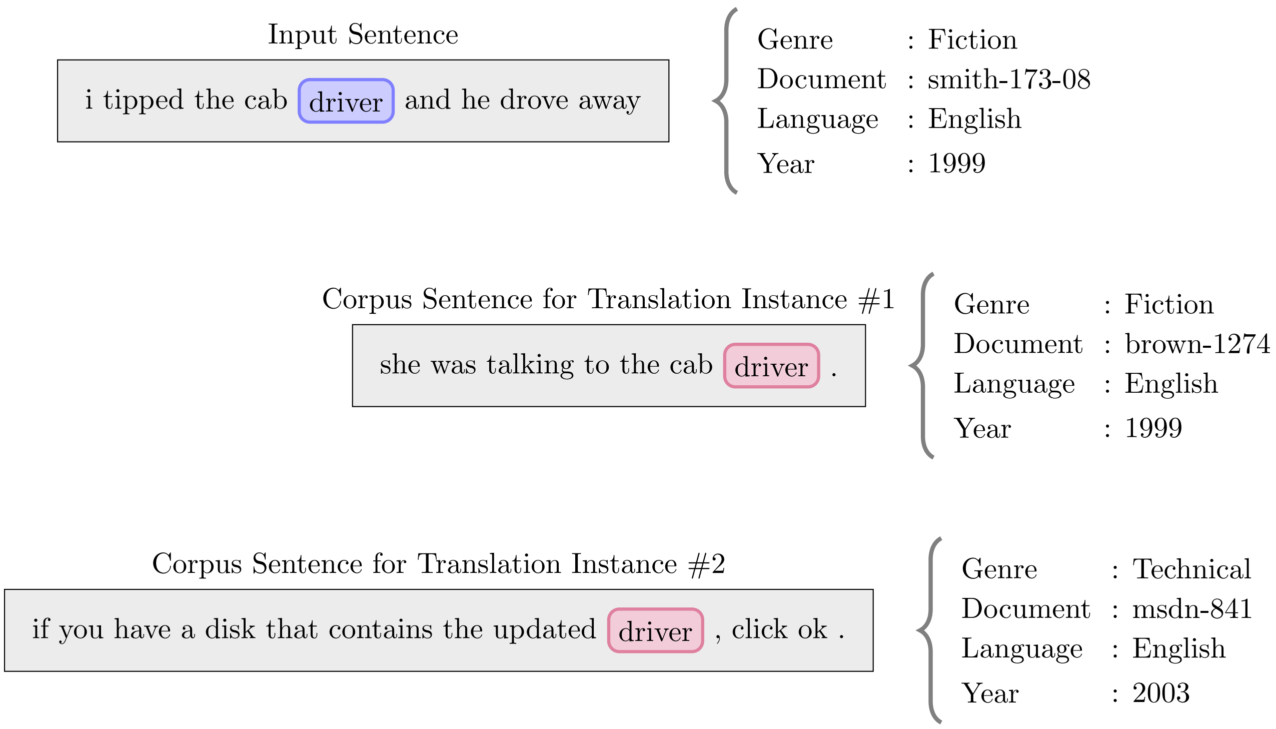

Input Sentence
i tipped the cab driver and he drove away
Genre : Fiction
Document : smith-173-08
Language : English
Year : 1999
Corpus Sentence for Translation Instance #1
she was talking to the cab driver .
Genre : Fiction
Document : brown-1274
Language : English
Year : 1999
Corpus Sentence for Translation Instance #2
if you have a disk that contains the updated driver , click ok .
Genre : Technical
Document : msdn-841
Language : English
Year : 2003

Figure 6.1: The Role of Sentence Annotations when Scoring Translation Instances

larger corpus, it is not obvious which, if any, texts should be preferred during translation. Cunei's instance-based approach allows it to use all instances of translation, but score each based on its similarity to the input sentence. Specifically, we compare the context of the input sentence to each instance of translation to identify which instances are the most relevant. Note that our model scores each instance of translation, not simply the sentence in which it is situated. Our approach also has the advantage of constructing a single model in which the model weights can be learned on a development set, as previously presented in Chapter 4.

This chapter is an expansion of the work published in (Phillips and Brown, 2009). We begin this chapter describing several new context-based features. We introduce the concept of a sentence annotation in Section 6.1 and describe how this form of context can be used to identify similar sentences. In Section 6.2 we score the tokens that surround each translation instance. In these sections, we discuss the motivation behind modeling the particular categories of context and then detail how each of the new features is calculated. Experimental results are then reported in Section 6.3. Before concluding, we explore an alternative method for sampling in Section 6.4 that exploits the new context features described in this chapter.

6.1 Context from Sentence Annotations

In preparing a large parallel corpus for machine translation, one usually combines text from multiple sources. For example, the WMT translation tasks provide parallel text from newswire commentary and European parliament proceedings. Similarly, the curated CzEng v0.9 corpus is a broad collection of texts from diverse genres such as fiction, movie subtitles, technical documentation, and web pages. Each of these sources may present different translations of the same phrase. While genre shifts may be the most obvious delineator, even a change in vocabulary or style of writing can affect translation choice. For example, news articles from The New York Times and The Associated Press exhibit very different writing styles regardless of the subject being discussed. Thus, a corpus may contain significant variability, but often these variances are correlated with the genre, style, or some other information about the text.

To represent this knowledge and make it available to the model, Cunei stores a set of annotations with each sentence in the corpus. Any type of annotation can be provided to Cunei, but we expect those that identify the genre and style of writing to be the most useful. Each sentence can be associated with multiple annotations, allowing the user to provide all available information at varying levels of detail. As an example, the Europarl distribution includes XML markup containing additional information about the text. One of these sentences was recorded in the *Europarl* proceedings in *November* of the year *2003* and spoken originally in *Spanish* by *De Palacio* who is the *Vice-President of the Commission*. Each italicized piece of information represents an annotation that we associate with that sentence in the corpus. Figure 6.1 further highlights how sentence annotations on the input sentence and the corpus sentence may be beneficial in identifying the most relevant translation instances.

We use the term annotation quite flexibly. In this chapter the focus is on annotations that are assigned to a sentence pair in the training data. Later, in Chapter 8 we will describe corpus annotations that are assigned to a sequence of tokens on either the source or target of a sentence in the training data. In both cases the additional information may represent metadata, human annotations, or statistically generated labels. The term annotation denotes how the information is applied to the corpus and not where it came from.

To limit the scope of this work, in this chapter we only explore the use of sentence annotations made by humans. We believe human annotations will be more accurate and provide the best opportunity for Cunei's model. Using an external classifier to create additional annotations is certainly worth exploring in the future, but we already present an alternative method for automated similarity detection in Section 6.2. However, this decision means we are at the mercy of those who collect the parallel text with regard to how much detail from the original source is preserved. Almost all corpora identify their genre(s), but the presence of additional information varies.

For these experiments, we intentionally selected corpora that provide extra information that we could embed as sentence-level annotations. In German-English translation, the Europarl corpus provides `Genre`, `Year`, `Month`, `Date`, `Language`, `Affiliation`, and `Speaker`. In Czech-English translation, the CzEng corpus provides only `Genre` and `Chapter`. The `Genre` is available for every sentence, but many of the other annotations are sprinkled throughout the training corpus. The annotations `Year`, `Month`, `Date`, and `Chapter` have numeric values. The `Language` annotation is associated with a two-letter ISO 639-1 language code; there are 25 possible values in the Europarl corpus. The values of `Speaker` and `Affiliation` are names of people, organizations, or roles. They are represented as textual strings and, unfortunately, not necessarily consistent throughout the corpus. In German-English, the `Genre` annotation is either `Europarl` or `News-Commentary`. In Czech-English, the `Genre` annotation is `Europarl`, `Fiction`, `Wikipedia`, `News`, `News-Commentary`, `Web`, `Subtitles`, or `Technical`.

6.1.1 Static Mixture Model

We can differentiate among sentences in the corpus based on the value(s) of their sentence annotations. Multiple sentences that are marked with the same sentence annotation compose a collection of related sentences. When a sentence contains multiple annotations, it simply belongs to more than one of these collections. Thus, within the broader corpus, we can identify collections of related sentences. We model each of these collections with a feature; the weight for this feature will correspond to whether or not instances of translation originating from that collection should be favored during translation.

When an instance of translation is retrieved from the corpus, we generate a set of binary features, formally defined in Table 6.1, identifying which sentence annotations are associated with it. The weights for these features are essentially mixture weights among all collections (as defined by the sentence annotations) that exist in the corpus. These weights, along with all the other model weights, are learned during training. Since training will select an appropriate mixture of collections to translate the development set, it is possible that these features will not generalize to the test set. We should be aware of this limitation, but it is often the case that the development and test sets are very similar.

6.1.2 Dynamic Comparison to Input

The sentence annotations in the corpus can also be compared directly to the annotations for the input sentence. This comparison, similar to Section 6.1.1, identifies collections of related sentences, but the collections are not defined statically and independent of the input sentence. Instead, we dynamically identify collections of sentences that are related to the input sentence based on the existence of similar annotations.

We score the similarity between the input sentence and an instance of translation using one feature for each type of sentence annotation (genre, author, year, etc.). The score for the feature depends on whether the values of that sentence annotation (news, Shakespeare, 1991, etc.) for the translation instance and the input sentence are equivalent. A sentence may contain multiple sentence annotations of the same type. We compute the score for each type of annotation as the accuracy between the two sets of values. The calculation is shown in Table 6.1. In addition, the accuracy is smoothed by adding one to the numerator and denominator, which enables a distinction between $\frac{0}{1}$ and $\frac{0}{5}$.[1] For example, a transcription we are translating from a TV newscaster may be annotated with the genres *spoken language* and *newswire*, while a relevant sentence from the corpus may only match the genre *newswire*. In this case, the accuracy for the genre annotation is $\frac{1}{2}$ and the smoothed feature value is $\frac{2}{3}$. Note that the actual value(s) of the sentence annotation are irrelevant–it only matters that the input sentence and translation instance have the same value(s).

The weights we learn during training suggest how important it is that we generate translations from sentences in the corpus with the same sentence annotations. Because these sentence annotation similarity features are conditioned on the input sentence, the system should be better at adapting to changes in the input. However, these features will only be computed during training when a particular sentence annotation is present in the development set. This does create an increased risk that the training procedure will not have sufficient evidence to determine their optimal weights.

6.2 Context from Surrounding Tokens

Instead of relying on human annotations, we can also use the vocabulary and structure of the text to automatically determine similarity between the input and sentences in the corpus. Figure 6.2 is a visual representation of how the surrounding tokens should be taken into consideration when scoring translation instances.

[1] This method bears similarity to plus-one smoothing, but it is not equivalent. Adding one to *both* the numerator and denominator does not yield a probability distribution. However, it is a good feature value in that we always obtain a non-zero score that is less than or equal to one. In addition, this score can be calculated when observing one instance without knowing the full distribution.

Let A be the set of annotations from the corpus that correspond to the translation instance and A' be the set of annotations for the input sentence. We will use A_X to represent the subset of annotations in A of type X. The features below are limited to the annotation types `Genre` and `Year`, but these features will be created for all annotations known to the system.

Static Mixture Model

`Corpus.Sentence.Group.Web.Weights.Match`	$\begin{cases} 1 & \exists a \in A_{\texttt{Genre}} : a = \texttt{Web} \\ 0 & \text{otherwise} \end{cases}$
`Corpus.Sentence.Group.News.Weights.Match`	$\begin{cases} 1 & \exists a \in A_{\texttt{Genre}} : a = \texttt{News} \\ 0 & \text{otherwise} \end{cases}$
`Corpus.Sentence.Group.1999.Weights.Match`	$\begin{cases} 1 & \exists a \in A_{\texttt{Year}} : a = \texttt{1999} \\ 0 & \text{otherwise} \end{cases}$

Dynamic Comparison to Input

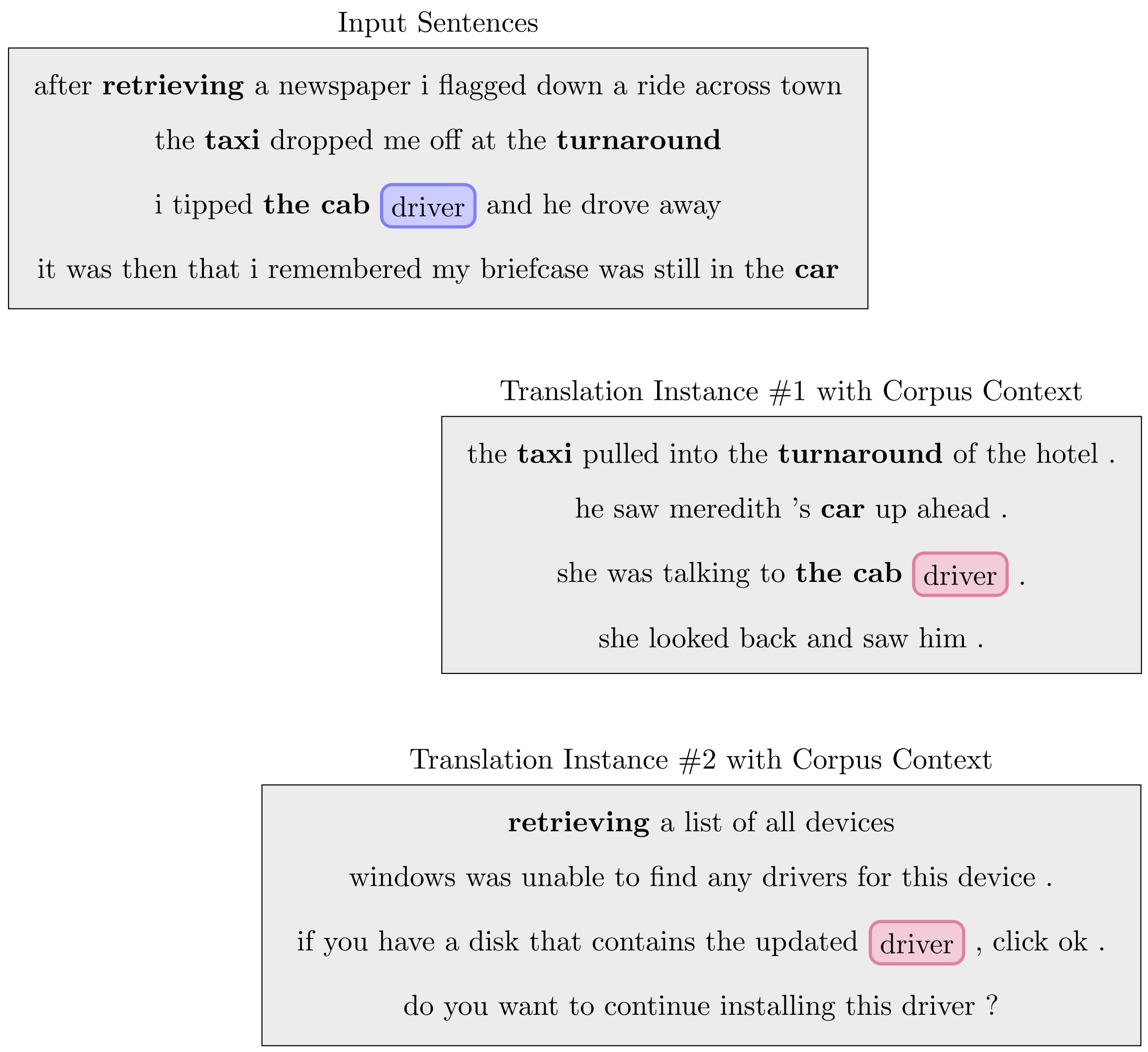
`Match.Weights.Divergence.Genre`	$\ln \frac{1 + \lvert A_{\texttt{Genre}} \cap A'_{\texttt{Genre}} \rvert}{1 + \lvert A_{\texttt{Genre}} \cup A'_{\texttt{Genre}} \rvert}$
`Match.Weights.Divergence.Year`	$\ln \frac{1 + \lvert A_{\texttt{Year}} \cap A'_{\texttt{Year}} \rvert}{1 + \lvert A_{\texttt{Year}} \cup A'_{\texttt{Year}} \rvert}$

Table 6.1: Description of Source Context Features Based on Sentence Annotations

6.2.1 Intra-Sentential Context

In order to capture local, intra-sentential context, we incorporate features that favor instances of translation in which the words immediately to the left and/or right also match the input sentence. For example, if we wish to translate the French phrase "Je bois", we would likely be able to locate an instance of translation for the complete phrase that identifies the English translation as "I drink". The decoder will usually prefer to select the longest match, but if this is part of a longer sentence, we could have a complete translation for everything except the word "bois". Within the original context, the word "bois" should be translated as "drink", but when translated on its own, we will find many translation instances in the corpus that provide the translation of "wood".

Our solution is to exploit the fact that one or more translation instances for "drink" also matched one additional word of the input ("Je") outside the phrase boundary while none of the translation instances for "wood" shared any context with the input. We achieve this by identifying longer matches before shorter matches and storing the longest possible match associated with each location in the corpus. Then, to affect the scoring, we add features that prefer translation instances that have long contextual matches with the input sentence. Specifically, we calculate the six new intra-sentential context features described in Table 6.2 using our knowledge of the longest match at each location in the corpus.


Input Sentences

after **retrieving** a newspaper i flagged down a ride across town

the **taxi** dropped me off at the **turnaround**

i tipped **the cab** driver and he drove away

it was then that i remembered my briefcase was still in the **car**

Translation Instance #1 with Corpus Context

the **taxi** pulled into the **turnaround** of the hotel .

he saw meredith 's **car** up ahead .

she was talking to **the cab** driver .

she looked back and saw him .

Translation Instance #2 with Corpus Context

retrieving a list of all devices

windows was unable to find any drivers for this device .

if you have a disk that contains the updated driver , click ok .

do you want to continue installing this driver ?

Figure 6.2: The Role of Surrounding Tokens when Scoring Translation Instances

These feature calculations were actually changed during the course of our research. Originally, the computation was simpler and only looked at the *external* context. For example, the feature `Match.Weights.Context.Left.Length` was computed as $\ln(m_s - p_s + 1)$. We noticed by looking at the translation output that this strongly biased the system against selecting longer phrase pairs when context was available with shorter phrase pairs. As the length of a match increases, by definition, the amount of external context decreases. This created an imbalance because the long match had little or no context while the composition of many shorter matches had significant context. The issue was that we neglected to model the fact that long matches intrinsically contain context embedded within the phrase. We note this issue to remind the reader that machine translation systems are still sensitive to how features are formulated. In general, the features for each instance of translation must be comparable to all instances of translation and not just comparable to those that match the same source span.

6.2.2 Document Context

On a broader scale, we can dynamically weight documents within the corpus based on their similarity to the input document. We capture this effect by modeling each document as a bag of words represented with a frequency vector of types. Extremely frequent words, such as ‘the’ in English, provide little information and may be discarded. Using vector-based distance metrics, we then compare the vector for the input document to each document vector in the corpus. A wide range of document-level distance metrics have been invented and explored within information retrieval. Currently, we include as separate features cosine distance, Jensen-Shannon distance (Lin, 1991), precision, and recall. Taking another cue from information retrieval, we also use TF-IDF scores instead of raw frequency counts in the vector of types. Table 6.2 shows the calculation for each of these features.

We use sentence annotations to identify which sentences within the corpus belong to a document. Generally, this is accomplished by adding a specific annotation (e.g. the annotation `Document`). It is usually most sensible to select an annotation which groups together small collections of sentences that were written as a unit by one author, but the user is free to select any annotation for defining the set of sentences which belong to a “document”. The difference from Section 6.1.1 and Section 6.1.2 is that instead of simply using the annotation labels, these document-based features look internally at the contents of each collection.

Occasionally, sentence annotations may not be available to identify document boundaries. For this situation, Cunei also provides the ability to represent documents with an arbitrarily-sized window of sentences. The user specifies the number of sentences to include in each window and may simultaneously enable multiple windows. Each window size creates a new set of pseudo-documents that all contain the same number of sentences. Using multiple windows can help reduce problems with the pseudo-documents straddling an actual document boundary. Due to parallelization and other performance considerations, these windows can only be generated for the corpus and not the input.

The system engineer can modify the scope of the document boundaries by selecting which annotations or sentence windows to use. For example, if broader contextual clues are desired, then the system engineer can select a more general sentence annotation or increase the size of the window. Furthermore, multiple document boundaries can be used simultaneously and combined with the sentence windows. This flexibility enables the user to create context-based features for documents at varying levels of granularity.

6.3 Experiments

We perform experiments using the Czech-English and German-English datasets described previously. We also use the same *Cunei Baseline* configuration with per-instance alignment that was described with the datasets in the previous chapter. For each language pair, we evaluate five experimental conditions corresponding to Sections 6.1.1, 6.1.2, 6.2.1, 6.2.2, and a combined system that enables all new context features. Recall that of the two corpora, the German-English parallel training data is less diverse as it is dominated by Europarl proceedings (the newswire commentary is only 7% of the corpus). However, within this particular genre, there are many contextual clues embedded as annotations that our features might leverage, such as the date, speaker, language, and affiliation of the original text.

Intra-Sentential Context

Let the longest match be from position p_s to position p_e and the current translation instance being scored cover the span starting at m_s and ending at m_e.

`Match.Weights.Context.Left.1-gram`	$\begin{cases} m_e - m_s & \text{if } m_s - p_s \geq 1 \\ m_e - m_s - 1 & \text{otherwise} \end{cases}$
`Match.Weights.Context.Left.2-gram`	$\begin{cases} m_e - m_s & \text{if } m_s - p_s \geq 2 \\ m_e - m_s - 1 & \text{if } m_s - p_s = 1 \\ m_e - m_s - 2 & \text{otherwise} \end{cases}$
`Match.Weights.Context.Left.Length`	$\sum_{i=1}^{m_e - m_s} \ln(i + m_s - p_s)$
`Match.Weights.Context.Right.1-gram`	$\begin{cases} m_e - m_s & \text{if } p_e - m_e \geq 1 \\ m_e - m_s - 1 & \text{otherwise} \end{cases}$
`Match.Weights.Context.Right.2-gram`	$\begin{cases} m_e - m_s & \text{if } p_e - m_e \geq 2 \\ m_e - m_s - 1 & \text{if } p_e - m_e = 1 \\ m_e - m_s - 2 & \text{otherwise} \end{cases}$
`Match.Weights.Context.Right.Length`	$\sum_{i=1}^{m_e - m_s} \ln(i + p_e - m_e)$

Document Context

Let $TF(t, d)$ be the count of type t in either the corpus document d or the input document d'. Let DF be the total number of documents and $DF(t)$ be the count of documents (over both the corpus and input) that contain the type t. Multiple context groups can be used–these refer to the group identified as `Docs`.

$$\alpha_i = TF(t_i, d) \ln(\frac{DF + 1}{DF(t_i)}) \qquad \beta_i = TF(t_i, d') \ln(\frac{DF + 1}{DF(t_i)})$$

`Context.Group.Docs.Weights.Cosine`	$-\ln\left(1 - \frac{\sum_i \alpha_i \beta_i}{\sqrt{\sum_i {\alpha_i}^2}\sqrt{\sum_i {\beta_i}^2}}\right)$
`Context.Group.Docs.Weights.JensenShannon`	$-\ln \sum_i \frac{\alpha_i \log_2 \frac{2\alpha_i}{\alpha_i + \beta_i}}{2 \sum_j \alpha_j} + \frac{\beta_i \log_2 \frac{2\beta_i}{\alpha_i + \beta_i}}{2 \sum_j \beta_j}$
`Context.Group.Docs.Weights.Precision`	$-\ln\left(1 - \frac{1 + \sum_i \min(\alpha_i, \beta_i)}{1 + \sum_i \beta_i}\right)$
`Context.Group.Docs.Weights.Recall`	$-\ln\left(1 - \frac{1 + \sum_i \min(\alpha_i, \beta_i)}{1 + \sum_i \alpha_i}\right)$

Table 6.2: Description of Source Context Features Based on Surrounding Tokens

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	31.61	6.772	53.09	53.26	30.76	7.212	52.49	53.85
+ Static Annotations	32.56†	6.897	53.51	52.31	30.77	7.211	52.44	53.80
+ Dynamic Annotations	**32.46**†	*6.859*	53.45	*52.58*	31.01‡	7.241	52.54	53.51
+ Sentence Context	**32.47**†	*6.856*	53.65	**52.41**	30.91	7.199	52.60	53.81
+ Document Context	32.38†	6.890	53.56	*52.36*	31.05‡	7.246	52.91	53.45
All Context Features	**32.53**†	**6.907**	53.63	**52.28**	31.20†	7.271	*52.90*	53.21

Table 6.3: Evaluation of Incorporating Source Context on CzEng v0.9

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	27.50	6.659	51.78	58.85	25.76	6.675	52.13	59.45
+ Static Annotations	*27.91*†	*6.703*	51.80	**58.43**	26.50†	6.735	52.22	59.13
+ Dynamic Annotations	*27.90*†	*6.715*	51.78	58.37	**26.17**†	6.699	52.17	59.50
+ Sentence Context	*27.85*†	6.691	51.87	58.63	**26.63**†	**6.764**	52.36	*58.82*
+ Document Context	*27.83*†	6.701	51.93	*58.54*	*26.22*†	6.738	52.30	59.14
All Context Features	28.46†	6.747	51.87	58.15	26.86†	**6.767**	52.14	*58.62*

Table 6.4: Evaluation of Incorporating Source Context on German Europarl v6

Scores in bold are statistically significant over multiple runs compared to the *Cunei Baseline* at p = 0.05; italics at p = 0.1
BLEU scores with † are statistically significant over random sampling compared to the *Cunei Baseline* at p = 0.05; ‡ at p = 0.1

6.3.1 Analysis

All experiments with Czech-English and German-English show improvement over the baseline system according to BLEU (and *almost* always according to NIST, Meteor, and TER despite the objective function not including those metrics). Incorporating context within the Czech-English system resulted in gains between 0.77-0.95 BLEU on the development set and 0.01-0.44 BLEU on the test set. Similarly, the range of improvement from context in German-English was 0.33-0.96 BLEU on the development set and 0.41-1.10 BLEU on the test set. The full battery of metrics are reported in Tables 6.3 and 6.4. As a reminder, the training procedure involves randomization, so each configuration was run twice. The reported scores are from the configuration with the highest BLEU score on the development set. The standard deviation over all BLEU scores was 0.21 in Czech-English and 0.12 in German-English, but the variance is not evenly distributed. Due to this variability over multiple runs, some of the larger gains reported in the tables are still not statistically significant.

The most apparent result from these evaluations is that adding new context features improves the evaluation scores on the development set. This is not entirely surprising as new features *should* only benefit the system; training can always assign a weight of zero voiding the impact of a feature on the system. That being said, the results are still encouraging because *in practice* adding additional features sometimes hurts performance (the training procedure gets stuck in a local minimum). Furthermore, in many cases the improvement on the development set was statistically significant over the baseline, suggesting that these new features provided novel information that the system previously lacked.

Unfortunately, the rank correlation between the scores on the development and test sets is quite low. All metrics generally show improvement with the use of context on both the development and test set, but not with the same magnitude. For example, the best development score for the Czech-English system was *Static Annotations*, but this configuration was nearly equivalent to *Cunei Baseline* on the test set. However, the BLEU scores on the development set for each of the context-based systems (within the same language pair) are quite close to one another, and we should not attempt to extrapolate too much from their relative ranking.

The one configuration that consistently performed well across language pairs on both the development and test sets is *All Context Features,*. This configuration produced the highest BLEU score (and among the highest NIST, Meteor, and TER scores) on the test sets. For comparison, the second-best configuration on the test set in both Czech-English and German-English was *Sentence Context*. However, *All Context Features* still improved over *Sentence Context* on the test set by 0.3 BLEU in Czech-English and 0.23 BLEU in German-English. The gains from incorporating all the context features are not as dramatic as the initial gain over the baseline from sentence context, but we are encouraged that mixing in additional categories of context still benefits the system. As identified in Tables 6.3 and 6.4, several of the gains with the *All Context Features* configuration are also statistically significant.

The relative improvement over the baseline from the static annotation features was much larger in German than in Czech. The Czech-English corpus included many genres, but no other types of annotation. The dynamic annotations performed well in Czech-English as they enabled Cunei to select translation instances from the same genre as the input sentence, which varied sentence to sentence. There may have been less room for improvement by setting a static weight for genres in the corpus because the test set included all genres in equal proportion. On the other hand, the characteristics of the German-English development and test sets were quite different in that they consisted of a single genre (Europarl), but contained numerous sentence-level annotations. With only one genre in the test set, the static annotation features were likely adequate for identifying relevant translation instances. Using dynamic annotations was better than the baseline in German-English, but worse than using static annotations. We suspect the cause of this to be that the non-genre annotations were sparse and led to overfitting on the development set.

A peculiar result is that all configurations in German-English have little effect on the Meteor score. Unlike the other metrics, Meteor is more focused on recall and permits synonym matching, but neither of these differences is a likely explanation of the constant scores. We remind the reader that the objective function used during training only incorporated BLEU. Often the training procedure is able to improve the BLEU score at the expense of NIST, Meteor, or TER. The training procedure does not know anything about these other metrics, but some characteristics that BLEU desires in a sentence are opposite those in other metrics. For example, BLEU is dominated by precision whereas Meteor is dominated by recall. While we prefer output that ranks highly among multiple metrics, lack of gain in a metric that is not part of the objective function is not indicative of a problem.

Each category of context we introduced to the Czech-English and German-English systems demonstrated improvement over the baseline systems. Furthermore, *All Context Features* was consistently better than any individual system according to BLEU, NIST, and TER. The gains from the combined systems were not a linear summation of the gains from each context category, but they were larger and more frequently statistically significant over the baseline systems. This suggests that we may safely include all context features without knowing beforehand which, if any, are most applicable for a particular corpus and language pair.

6.3.2 Examples

We will be more capable of understanding the effects from each category of context when analyzing the output of individual sentences if we compare systems that are *minimally* different from the baseline. The context and non-context weights for all systems in Tables 6.3 and 6.4 are jointly learned with the training procedure previously presented in Chapter 4. As a result, the common, non-context features can and do receive different weights among the various configurations. In addition, the training procedure will select optimal model weights for the entire development set, but the quality of one sentence may be degraded in order to improve the quality of one or more other sentences. There are, thus, many possible reasons unrelated to the presence of context that the output of a single sentence could be different between two systems when all model weights differ.

Therefore, to generate comparable output, we start with the *Cunei Baseline* configuration (which has learned feature weights) and add each set of context features with default values. These default values are generally in the same range as those from a configuration with learned weights, but they are not optimal. Adding the context features in this manner may also introduce a systematic bias (such as generating output that is too short) that had previously been balanced by modifying the weight of some common features. Thus, the resulting systems shown in Tables 6.5 and 6.6 do not perform nearly as well as their jointly trained counterparts. However, these systems show similar (albeit not as convincing) improvement from the context features, and we are guaranteed that all deviations from the baseline are due to the presence or absence of context.

An important point to remember is that our approach is data-driven. While the context features are designed to bias the output, Cunei is still reliant on the words present in the corpus. The lack of actual semantic knowledge is evident in Example 11. All of the context categories correctly select an adjective instead of a noun, but the hypothesis space is unfortunately split between "southeastern" and "south-eastern". To the human reader this is at most an insignificant difference in style, but to a data-driven machine translation system these are two distinct tokens and each is scored separately.

	Development Set				Test Set			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	31.61	6.772	53.09	53.26	30.76	7.212	52.49	53.85
+ Static Annotations	31.78	6.843	53.13	52.68	30.63	7.219	52.39	53.68
+ Dynamic Annotations	31.97	6.861	53.25	52.55	31.02	7.274	52.67	53.32
+ Sentence Context	31.72	6.822	53.20	52.83	30.66	7.224	52.34	53.73
+ Document Context	31.98	6.842	53.17	52.72	31.07	7.258	52.80	53.33
All Context Features	31.82	6.895	53.02	52.17	30.53	7.244	52.33	53.33

Table 6.5: Evaluation of Incorporating Source Context with Default Weights on CzEng v0.9

	Development Set				Test Set			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	27.50	6.659	51.78	58.85	25.76	6.675	52.13	59.45
+ Static Annotations	27.50	6.661	51.83	58.88	25.81	6.682	52.13	59.41
+ Dynamic Annotations	27.46	6.653	51.82	58.90	25.89	6.695	52.10	59.23
+ Sentence Context	27.08	6.613	51.43	58.99	26.01	6.724	52.00	58.98
+ Document Context	27.28	6.641	51.67	59.08	25.82	6.679	52.12	59.42
All Context Features	26.96	6.617	51.45	58.99	26.09	6.720	52.00	58.95

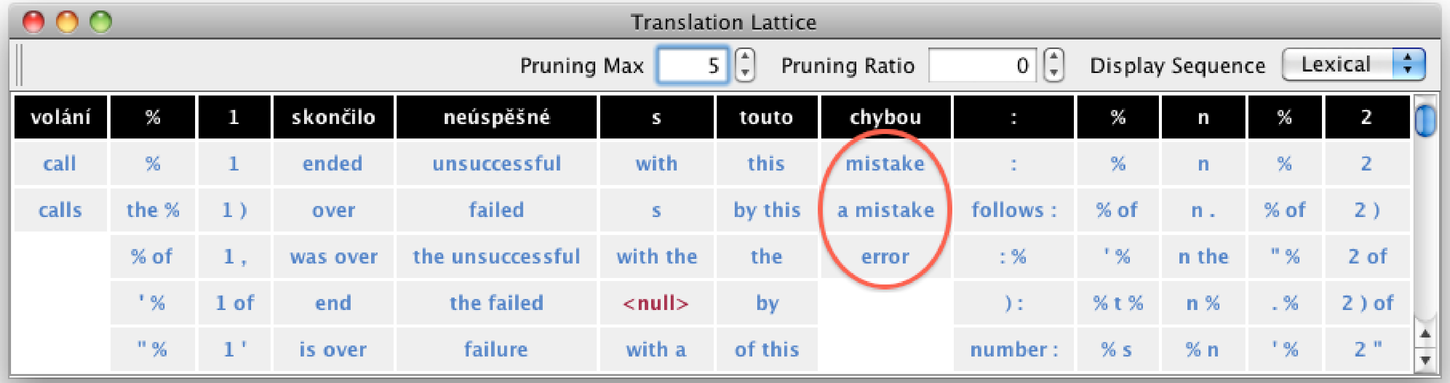
Table 6.6: Evaluation of Incorporating Source Context with Default Weights on German Europarl v6

Target Phrase	**Original Rank**	**Technical**	**Europarl**	**Genre**	**Left 1-gram**	**Right 1-gram**
		Static Annotations		*Dyn Ann*	*Sentence Context*	
Source Phrase: touto chybou						
by this error	1	2.0000	0.0000	0.0000	1.0000	1.0000
this error	2	2.0000	0.0000	0.0000	1.0000	1.0000
ended by this error	7	2.0000	0.0000	0.0000	1.0000	1.0000
this	3	0.0000	0.0000	-2.1972	1.0000	1.0000
by this	4	0.0000	0.0000	-2.1972	1.0000	1.0000
by this breech	5	0.0000	0.0000	-2.1972	1.0000	1.0000
Source Phrase: chybou :						
error :	1	2.0000	0.0000	0.0000	1.0000	1.9766
following error :	2	2.0000	0.0000	0.0000	1.0000	1.9054
with error :	3	2.0000	0.0000	0.0000	1.0000	2.0000
error : %	4	2.0000	0.0000	0.0000	1.0000	2.0000
error =	5	2.0000	0.0000	0.0000	1.0000	2.0000
Source Phrase: chybou						
error	3	0.9808	0.0034	-0.0211	0.0297	0.5559
mistake	1	0.0000	0.5574	-1.0986	0.0000	0.0000
a mistake	2	0.0000	0.4330	-1.0986	0.0000	0.0000
following error	19	1.0000	0.0000	0.0000	0.0000	1.0000
an error	9	0.9627	0.0065	-0.0410	0.0000	0.0000
reassign	18	1.0000	0.0000	0.0000	0.0000	0.0000

Table 6.7: Selected Source Context Features for Phrase Pairs in Example 13

The baseline system performs well, and there are many sentences for which including the context features yields no or trivial modifications. When changes do occur, they are often subtle and result in the use of different words to refer to the same underlying concept, such as in Examples 12, 13 and 14. For illustration, the phrases “the status quo”, “the current state”, and “state of play” used in Example 12 are all semantically equivalent *given the right context*. However, when the output is a diagnostic message for a computer driver, the only appropriate selection is “the current state”.

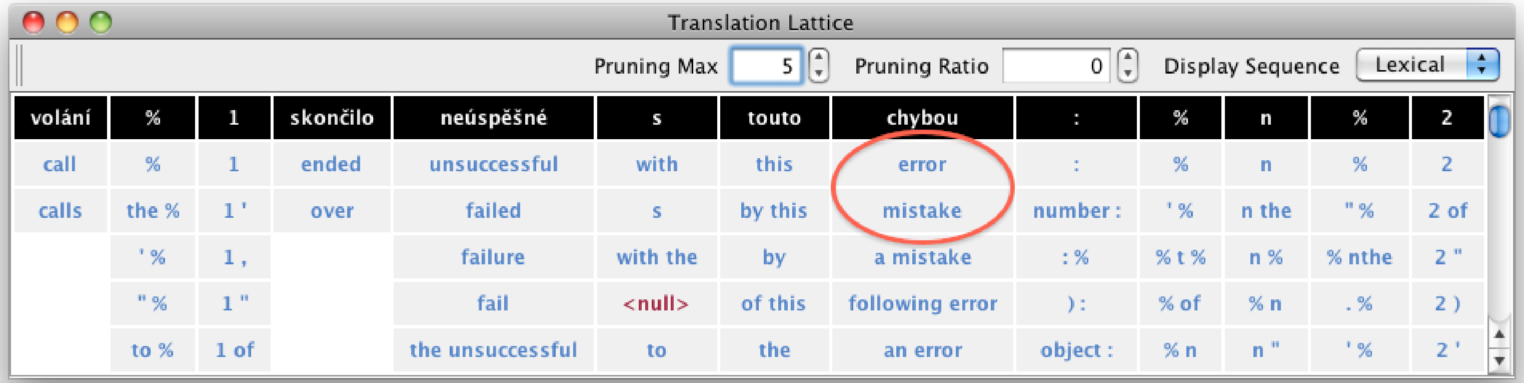
The (heavily pruned) translation lattice used by *Cunei Baseline* to translate Example 13 is visualized in Figure 6.3. In this example, the baseline system actually has the translations “this error” for “touto chybou” and “error :” for “chybou :” as shown in Table 6.7. The problem is that these longer phrases are not being used, and instead *Cunei Baseline* selects its highest-ranking translation for “chybou” which is “mistake”. Notice that the feature scores in Table 6.7 for phrase pairs with the word “error” indicate that they originate disproportionally from technical documents. The near-zero value for the dynamic genre feature also indicates that the translation “error” was constructed mostly from instances of translation with the same genre as the input sentence. Likewise, the feature values show that many of the instances of translation for “error” matched at least one additional word to the right in the input sentence. When these additional contextual features are employed by Cunei, then the translation “error” outranks “mistake” as shown in Figure 6.4.



Pruning Max 5 · Pruning Ratio 0 · Display Sequence Lexical

volání	%	1	skončilo	neúspěšné	s	touto	chybou	:	%	n	%	2
call	%	1	ended	unsuccessful	with	this	mistake	:	%	n	%	2
calls	the %	1)	over	failed	s	by this	a mistake	follows :	% of	n .	% of	2)
	% of	1 ,	was over	the unsuccessful	with the	the	error	: %	' %	n the	" %	2 of
	' %	1 of	end	the failed	<null>	by		) :	% t %	n %	. %	2) of
	" %	1 '	is over	failure	with a	of this		number :	% s	% n	' %	2 "

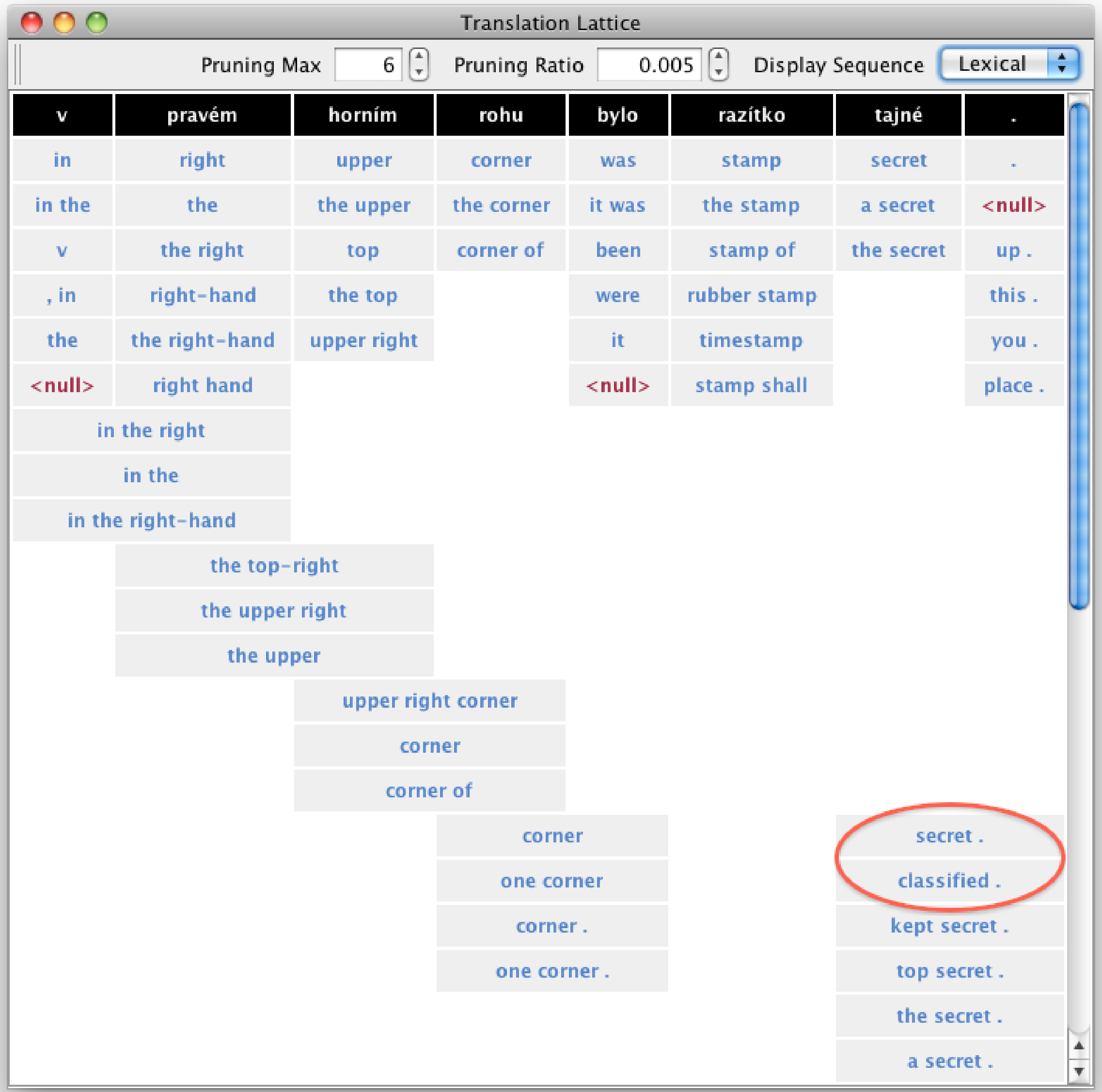
Figure 6.3: Translation Lattice for Example 13 with Cunei Baseline



Pruning Max 5 · Pruning Ratio 0 · Display Sequence Lexical

volání	%	1	skončilo	neúspěšné	s	touto	chybou	:	%	n	%	2
call	%	1	ended	unsuccessful	with	this	error	:	%	n	%	2
calls	the %	1 '	over	failed	s	by this	mistake	number :	' %	n the	" %	2 of
	' %	1 ,		failure	with the	by	a mistake	: %	% t %	n %	% nthe	2 "
	" %	1 "		fail	<null>	of this	following error	) :	% of	% n	. %	2)
	to %	1 of		the unsuccessful	to	the	an error	object :	% n	n "	' %	2 '

Figure 6.4: Translation Lattice for Example 13 with All Source Context Features

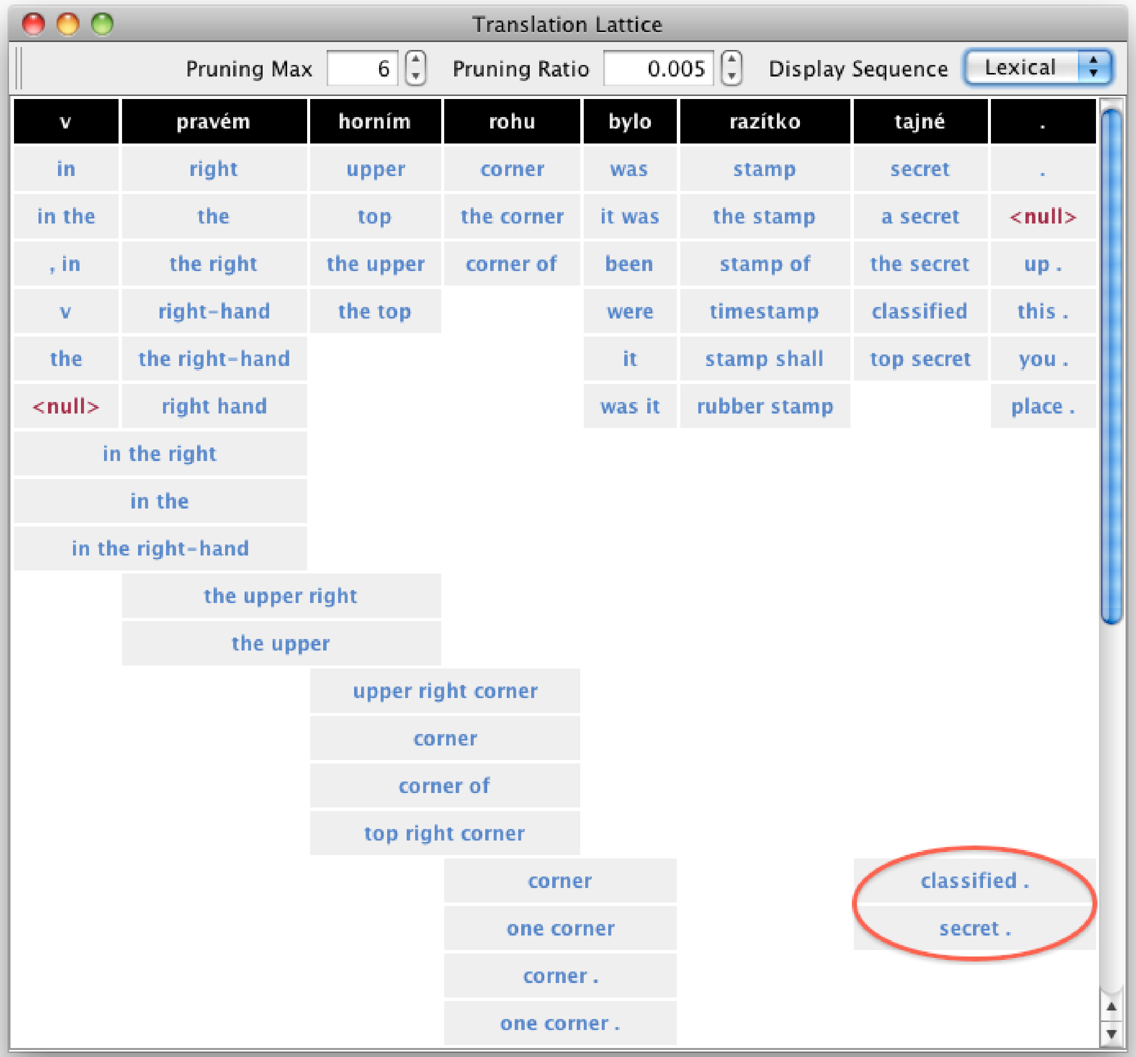


Translation Lattice

Pruning Max 6 Pruning Ratio 0.005 Display Sequence Lexical

v	pravém	horním	rohu	bylo	razítko	tajné	.
in	right	upper	corner	was	stamp	secret	.
in the	the	the upper	the corner	it was	the stamp	a secret	<null>
v	the right	top	corner of	been	stamp of	the secret	up .
, in	right-hand	the top		were	rubber stamp		this .
the	the right-hand	upper right		it	timestamp		you .
<null>	right hand			<null>	stamp shall		place .
in the right							
in the							
in the right-hand							
	the top-right						
	the upper right						
	the upper						
		upper right corner					
		corner					
		corner of					
			corner			secret .	
			one corner			classified .	
			corner .			kept secret .	
			one corner .			top secret .	
						the secret .	
						a secret .	

Figure 6.5: Translation Lattice for Example 14 with Cunei Baseline


Translation Lattice

Pruning Max 6 Pruning Ratio 0.005 Display Sequence Lexical

v	pravém	horním	rohu	bylo	razítko	tajné	.
in	right	upper	corner	was	stamp	secret	.
in the	the	top	the corner	it was	the stamp	a secret	<null>
, in	the right	the upper	corner of	been	stamp of	the secret	up .
v	right-hand	the top		were	timestamp	classified	this .
the	the right-hand			it	stamp shall	top secret	you .
<null>	right hand			was it	rubber stamp		place .
in the right							
in the							
in the right-hand							
	the upper right						
	the upper						
		upper right corner					
		corner					
		corner of					
		top right corner					
			corner			classified .	
			one corner			secret .	
			corner .				
			one corner .				

Figure 6.6: Translation Lattice for Example 14 with Document Context

A similar reranking occurs in Example 14 illustrated with Figures 6.5 and 6.6. The two-word phrase “tajné .” ranks the translation “secret .” above the translation “classified .” in *Cunei Baseline*. The addition of context features as shown in Table 6.8 reverses this ranking. Interestingly, the feature value for the genre gives preference to the translation “secret .”. However, the recall feature function that scores document context *strongly* prefers the translation “classified .” Translating the single word “tajné” as “classified” also shows a mismatch according to genre, but a strong preference based on document context. In addition, “classified” has stronger right sentential context than “secret”. Although “secret” is still ranked higher than “classified”, these context features increased the rank of “classified” from 14 in the baseline system to 6.

Example 20 has multiple changes that occur based on which context features are enabled. We do not show the translation lattices for this sentence as the changes occur over too large of a span to fit on a page, but the features are presented in Table 6.9. The most prominent change is that

Target Phrase	**Original Rank**	**Subtitles**	**Fiction**	**Genre**	**Right 1-gram**	**Recall**
		Static Annotations		*Dyn Ann*	*Sent Cont*	*Doc Cont*
Source Phrase: tajné						
secret	1	0.3281	0.4974	-0.5522	0.1581	0.0318
a secret	2	0.4165	0.5814	-0.4599	0.0513	0.0178
the secret	3	0.1856	0.6611	-0.3723	0.0808	0.0230
his secret	11	0.0000	1.0000	0.0000	0.0000	0.0631
kept secret	13	0.1918	0.8082	-0.2107	1.0000	0.1210
classified	14	1.0000	0.0000	-1.0986	0.6967	7.4356
top secret	4	0.9001	0.0000	-1.0986	0.3571	0.0540
top-secret	6	1.0000	0.0000	-1.0986	0.0000	0.0180
be secret	7	0.0000	0.0000	-1.0986	0.0000	0.0172
confidential	8	0.3397	0.0000	-1.0986	0.0106	0.0269
collusion	17	0.0000	0.9979	-0.0024	0.0000	0.0234
Source Phrase: tajné .						
classified .	2	2.0000	0.0000	-2.1972	1.0000	19.8575
secret .	1	0.6749	1.1448	-0.9395	1.0000	0.1310
kept secret .	3	0.1344	1.8656	-0.1477	1.0000	0.1147
top secret .	4	2.0000	0.0000	-2.1972	1.0000	0.2011
the secret .	5	2.0000	0.0000	-2.1972	1.0000	0.0798
classification .	9	0.0000	2.0000	0.0000	1.0000	0.0092

Table 6.8: Selected Source Context Features for Phrase Pairs in Example 14

the baseline system incorrectly produces the translation “clinical review” whereas the reference has “clinical trials”. This is a bit surprising as the highest-ranked translation of “klinische prüfung” in all configurations is in fact “clinical trials” and “clinical review” is not even a candidate translation for this two-word phrase. *Cunei Baseline* is selecting “review” based on it being the highest-ranked translation for the word “prüfung”. The translation of “prüfung” as “trials” is ranked 93 in *Cunei Baseline*. The addition of all context features moves this candidate up to rank 9, which allows the decoder to more readily select the correct translation. *Sentence Context* context corrects the translation “clinical review” to “clinical trials” as both the right and left context feature values prefer “trials”. The translation “trials” is also supported by the feature that assesses similarity of the language annotation, but the *Dynamic Annotations* configuration still does not correct the translation. In fact, *Dynamic Annotations* degrades another section of the translation by replacing “this objective” with “this aim”. While none of the phrase pairs that include the word “aim” change rank, the language and affiliation annotation feature give a slight preference to the use of “aim” instead of “objective” when translating either “dieses ziel” or “ziel”.

We presented four different types of context in this chapter because we believed they would leverage unique (and complementary) characteristics during the translation process. We do find evidence in the examples for each type of context yielding improvement over the baseline system. Examples 12, 13, and 18 show the effects of static annotations; Examples 12, 18, 19, and 21 show dynamic annotations; Examples 15, 17, and 19 show sentence context; and Examples 13, 14, and 21 illustrate modifications resulting from the use of document context. Interestingly, a large number of

Target Phrase	**Original Rank**	**Language**	**Affiliation**	**Left 1-gram**	**Right 1-gram**
		Dynamic	*Annotations*	*Sentence*	*Context*
Source Phrase: klinische prüfung					
clinical trials	1	-0.4363	0.0000	1.0000	1.8015
Source Phrase: prüfung					
review	1	-0.6655	-0.1937	0.0000	0.2821
verification	7	-0.6931	0.0000	0.0000	1.0000
test	8	-0.7064	-0.0472	0.0000	0.9164
examination	2	-0.6931	0.0000	0.0000	0.0000
audit	3	-0.8348	-0.1895	0.0000	0.2276
scrutiny	4	-0.7695	-0.1782	0.0000	0.6226
examining	5	-0.9891	0.0000	0.0000	0.2765
assessment	6	-0.7674	-0.2639	0.0000	0.1786
trials	93	0.0000	0.0000	1.0000	1.0000
examination of	9	-0.6931	0.0000	0.0000	0.0000
Source Phrase: dieses ziel					
this goal	1	-0.9060	-0.2090	1.0000	1.0000
this aim	2	-1.5470	-0.2026	1.0000	1.0000
this	3	-1.5355	-0.4813	1.0000	1.0000
that objective	6	-0.5940	-0.1682	1.5367	1.0000
this target	4	-1.6843	-0.6842	1.0000	1.3675
this objective	7	-1.7718	-0.3315	1.0000	1.0686
Source Phrase: ziel					
objective	1	-0.7770	-0.1801	0.7285	0.0210
aim	2	-0.7718	-0.0653	0.5599	0.1727
goal	3	-0.8158	-0.1527	0.7040	0.0000
target	4	-0.6931	0.0000	1.0000	0.0000
objectives	5	-0.7727	-0.0826	0.0000	0.0000

Table 6.9: Selected Source Context Features for Phrase Pairs in Example 20

these examples demonstrate multiple forms of context agreeing on the best output. This suggests that in many situations, evidence for a contextual change is prevalent throughout the corpus.

When all the context features are combined together, more often than not, the appropriate modification is selected (see Examples 12, 13, 15, 17, and 21). Sometimes the best output is not even selected unless all the features are combined, as is the case in Examples 16 and 20. The latter is an especially nice example where each context category alters part of the sentence, but only when combined together are all three terms “trials”, “believe”, and “objective” present in the output. Examples 14 and 18 are counter-examples to this trend where combining all the features produces instead additional, unwanted modifications to the output. We believe this behavior is partially explained by the fact that we are using default weights, but there may also be conflicting information provided by the various sources of context. Overall, these results are consistent with

	Czech-English		**German-English**	
	Runtime in Minutes	Words per Minute	Runtime in Minutes	Words per Minute
Cunei Baseline	36.95	500	106.13	252
+ Static Annotations	36.36	508	112.80	237
+ Dynamic Annotations	35.58	519	106.38	251
+ Sentence Context	36.42	507	103.69	258
+ Document Context	82.28	224	111.10	240
All Context Features	86.16	214	109.91	243

Table 6.10: Runtime on Test Sets of Incorporating Source Context

our findings in Tables 6.3 and 6.4, in which "All Context Features" outperformed any one category of context features.

6.3.3 Runtime

We perform the same timing experiments as in Section 5.1.5 of Chapter 3 with these new configurations to determine if the additional instance-based context features significantly alter performance. In fact, as shown in Table 6.10 translation time is approximately the same with the exception of computing document context in Czech-English. The document context features do have to load all the sentences that occur in the same document as the translation instance and then score each one for similarity to the input document (as discussed in Section 6.2.2). The document context feature functions are more costly, but the difference between this added cost in Czech-English and German-English is peculiar. Some of the information pertaining to each document is cached and it is possible that in translation of the German-English test set, which is all from the Europarl, the same documents in the training corpus were frequently used over and over.

6.4 Sampling Matches with Source Context

As described in Section 3.2.2 of Chapter 3, Cunei samples matches prior to phrase alignment and does not actually score all possible instances of translation. The default configuration with per-instance alignment performs a uniform sample, as none of the features could be scored with only the source phrase and its location in the corpus. However, similarity between the input sentence and the location in the corpus can be scored without the aligned target phrase. We experimented using the context features described in this chapter (Tables 6.2 and 6.1) to calculate a partial score for each match. The score is partial in that it lacks many of the features we normally apply to an instance of translation after phrase alignment. For example, features presented in Tables 3.1, 3.2, and 3.3 from Chapter 3 are not computed until *after* sampling. The partial score was then used to select the best matches according to Algorithm 3 previously presented on page 40.

The results of these experiments are presented in Tables 6.11 and 6.12. Sampling matches with the context features consistently showed improvement on the development set. On the test set, the only notable improvement was the Meteor score in German-English. Some of the other metrics registered statistically significant improvements, but the magnitude of these gains is incredibly small. The scores suggest that this approach is merely enabling more aggressive overfitting of the development set.

Overall, these differences are small. As was mentioned earlier, we observed that selecting a

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Uniform	32.53	6.907	53.63	52.28	31.20	7.271	52.90	53.21
Best Partial Score	*33.02*†	6.944	53.75	51.80	31.34	*7.289*	52.89	53.17

Table 6.11: Evaluation of Sampling on CzEng v0.9 with All Source Context Features

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Uniform	28.46	6.747	51.87	58.15	26.86	6.767	52.14	58.62
Best Partial Score	28.71	6.818	52.24	57.54	26.84	6.810	52.57	58.36

Table 6.12: Evaluation of Sampling on German Europarl v6 with All Source Context Features

larger sample had minimal impact on the score for a phrase pair. We, therefore, should not expect a change in the sampling algorithm to have a large impact either. This is due in part to the fact that sampling only affects phrase pairs whose source phrase occurs in the corpus more than the sampling threshold. In addition, all matches are already restricted to *exactly* matching the source phrase and, therefore, all matches should represent reasonable candidates.

One possible concern with this approach is that the context features are only a subset of the complete model. These features alone may not be an adequate heuristic for sampling. During training, the context features tend to receive less weight than other features like the translation frequency and the language model, which means that they will generally contribute less to the final model score. While these context features are an important source of information when distinguishing between between two competing translations, it is not clear that they will correlate strongly with the final model score.

6.5 Summary

In this chapter we presented four types of context: static annotations, dynamic annotations, intra-sentential context, and document context. For each type of context, we explained its role in the translation process and defined a set of features to include in the translation model. Combining all of these new context features resulted in a gain over the baseline system on the held-out test set of 0.44 BLEU in Czech-English and 1.1 BLEU in German-English. NIST, Meteor, and TER also generally showed gains, and several of these metric scores were statistically significant in comparison to the baseline. In addition, we looked at output of specific sentences which corroborated these findings and suggested that the context features enabled better lexical selection. 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 integrated into Cunei and leveraged to score the relevance of each translation instance.

Chapter 7

Incorporating Target Similarity

In this chapter we turn our attention to scoring $f_T(x,t)$, which represents the relevance of each translation instance with respect to the target hypothesis. Conceptually, this work is very similar to the last chapter where we scored $f_S(s,x)$. The key difference is that instead of comparing to the input sentence, we compare each translation instance with the target hypothesis. This relationship is visually illustrated in Figures 1.1 and 1.2 from the introduction with $f_T(x,t)$ as the counterpart to $f_S(s,x)$.

7.1 Complexity of the Task

The target sentence, unlike the input sentence, is not provided to us. In fact, constructing the correct target sentence is the goal of machine translation. In Section 3.2.5 of Chapter 3 we described the use of chart decoding and beam decoding to combine phrase pairs. Both of these methods construct a contiguous target hypothesis that is incrementally extended to cover new words in the input sentence. We use this n-best list of target hypotheses which have been scored by the model as a stand-in representation of the true target sentence.

Due to the additional complexity of comparing a translation instance to a target hypothesis we are in the process of constructing, we limit the scope of target context to the role of surrounding tokens within the sentence. Specifically, we only model preceding contextual matches. The preceding context is straightforward to match as the target hypotheses are constructed from left-to-right. Scoring the right context is possible by storing additional state information with each target hypothesis, but it would substantially complicate decoding. On the source we also used the surrounding tokens to model document similarity. Document context would be interesting to explore on the target as well, but it would require accumulating information across multiple sentences. This would break Cunei's current method of parallelization and also require substantial modification to the code. We also do not score sentence annotations on the target as they have already been scores on the source. The sentence annotations are defined per-sentence and apply equally to the source and target.

Section 7.2 describes intra-sentential context and defines new features to evaluate it on the target. We present how decoding is modified in Section 7.3 in order to perform the necessary calculations for these features and then provide experimental results in Section 7.4. Before concluding, we also describe in Section 7.5 a failed attempt at scoring target similarity without modifying the decoding process.

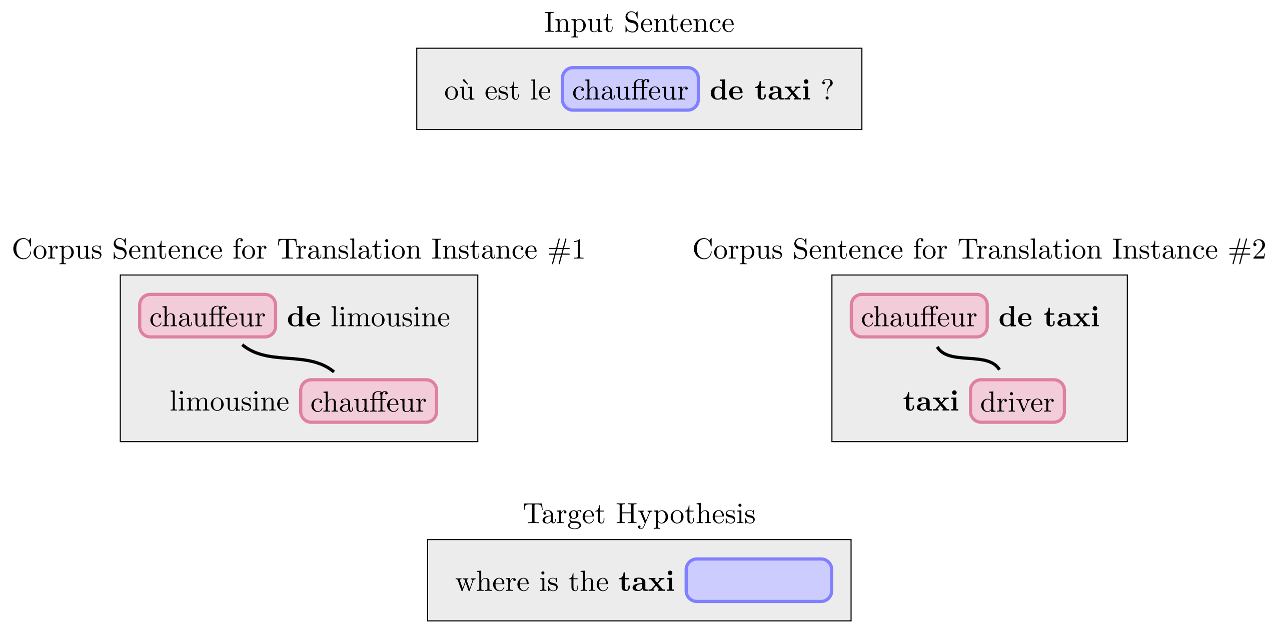

Input Sentence
où est le chauffeur **de taxi** ?
Corpus Sentence for Translation Instance #1
chauffeur **de** limousine
limousine chauffeur
Corpus Sentence for Translation Instance #2
chauffeur **de taxi**
taxi driver
Target Hypothesis
where is the **taxi**

Figure 7.1: The Role of Source and Target Context when Scoring Translation Instances

7.2 Intra-Sentential Context

Consider Figure 7.1 where we are attempting to translate the word “chauffeur” and we have two translation instances that offer guidance. As discussed in the previous chapter, we can identify that “driver” is a better translation than “chauffeur” by looking at the source context. In this case, the same bigram, “de taxi”, occurs immediately after both source phrases. By the same principle we can also consider what tokens are present in the target context. If we have constructed a target hypothesis that ends in “taxi”, we can use this information to prefer the translation “driver” as it too is immediately preceded by “taxi”. As is the case in this example, we recognize that the source context and target context may provide redundant information. On the other hand, the words which abut the source and target phrases may correspond to different sections of the input sentence due to reordering in the target hypothesis.

We score the surrounding intra-sentential tokens with n-gram calculations similar to those presented in the previous chapter. The target context features, shown in Table 7.1, identify which 1-grams, 2-grams, and 3-grams histories do not match. (Longer n-gram matches could be scored as well at the cost of more memory.) The comparison is performed between each instance of translation and the target hypothesis it will extend, as specified by the decoder. The calculation for each feature is quite simple: to score a context of length n, Cunei checks if the n right-most words in the target hypothesis are equivalent to the n words that precede the translation instance. Like the language model, the target context features are scored for each target word. Unlike the source context features, we cannot compute the target context features until the target hypothesis is being constructed. This requires a tight integration with decoding that is presented in the next section.

Intra-Sentential Context
Let n represent the 3-gram from the corpus that precedes the translation instance and h be the target hypothesis prior to being joined with the translation instance.

`Hypothesis.Weights.Context.1-gram`	$\begin{cases} -1 & \text{if } n_3 \neq h_{\lvert h\rvert} \\ 0 & \text{otherwise} \end{cases}$
`Hypothesis.Weights.Context.2-gram`	$\begin{cases} -1 & \text{if } n_2 n_3 \neq h_{\lvert h\rvert-1} h_{\lvert h\rvert} \\ 0 & \text{otherwise} \end{cases}$
`Hypothesis.Weights.Context.3-gram`	$\begin{cases} -1 & \text{if } n_1 ... n_3 \neq h_{\lvert h\rvert-2} ... h_{\lvert h\rvert} \\ 0 & \text{otherwise} \end{cases}$

Table 7.1: Description of Target Context Features Based on Surrounding Tokens

7.3 Combining Translation Units

Recall from Section 3.2.5 of Chapter 3 that Cunei scores each instance of translation and aggregates this information into phrase pairs that are stored in a lattice. This lattice of phrase pairs was designed to be an intermediate data structure that separated the processing of translation instances from decoding. It allowed Cunei to compute the summation over multiple translation instances once and store the score for each phrase pair in the lattice. This worked because the features added during decoding were independent of a particular instance of translation. In Equation 3.5 on page 45 we presented how independent features could be added to the score of a phrase pair after discarding the instances of translation. However, the scoring of target context described in this chapter is not independent of an instance of translation. Indeed, we are comparing the current target hypothesis with the target context of *each instance of translation*. Unfortunately, we cannot complete the calculation of these features until the target hypothesis is constructed during decoding.

Let us define a new feature ϕ_τ that scores an instance of translation based on the current target hypothesis. Extending Equation 3.4, the score for a phrase pair, with this feature yields:

$$m(s,t,\lambda) = \ln\left(\frac{1}{|X|}\sum_{x\in X} e^{\sum_j \lambda_j \cdot \phi_j(s,t,x) + \lambda_\tau \phi_\tau(x)}\right)$$

In this equation ϕ_τ is not known to be independent X and cannot be moved outside of the summation. Let us define a subset of translation instances Y for which the value of ϕ_τ is constant and therefore independent of all translation instances in Y. We can then divide the set of all translation instances X into sets $Y_1, Y_2, Y_3, \ldots, Y_n$, such that:

$$X = \bigcup_{i=1}^{n} Y_i \qquad \wedge \qquad Y_j \cap Y_k = \emptyset \quad \forall j \neq k$$

Now that the feature ϕ_τ is constant for all instances of translation in a particular Y_i, we can represent the score of a phrase pair as:

$$m(s,t,\lambda) = \ln\left(\frac{1}{|X|}\sum_{i=0}^{n} e^{\lambda_\tau \cdot \phi_\tau(Y_i)} \sum_{x\in Y_i} e^{\sum_j \lambda_j \cdot \phi_j(s,t,x)}\right) \tag{7.1}$$

Each Y_i represents the subset of translation instances with the same target context. Two contexts are considered equivalent in this formalism if they are assigned the same score by ϕ_τ. For simplicity in the explanation above, we only added a single new feature ϕ_τ, but the same method can be applied with multiple target context features. In this case, the value for all target context features must be constant over the translation instances in Y_i.

Given our target context features, presented in Table 7.1, each Y_i represents the set of translation instances that are proceeded by the same 3-gram. When building the translation lattice, Cunei stores the trigram target context and the inner summation for each Y_i. Each phrase pair, therefore, no longer contains a single score, but the score for each set Y_i. We can still discard the translation instances after generating the lattice, but the amount of information we store may not be appreciably less. If every instance of translation occurs in a unique target context, then each Y_i will represent a single instance of translation.

When a phrase pair is selected during decoding, ϕ_τ is calculated for each Y_i by comparing the associated target context to the current target hypothesis. The decoder calculates Equation 7.1 by summing over all target contexts and multiplying the new feature ϕ_τ for each context by the inner summation that was previously stored with the phrase pair in the lattice. The outer summation over all target contexts has to be computed every time the decoder attempts to extend the current target hypothesis with a new phrase pair. When there are many possible target contexts, which unfortunately is all too frequent, this is very slow.

7.4 Experiments

We evaluated the effect of modeling target context on the same German-English and Czech-English datasets that were used in the previous experiments. For consistency, we also continue to compare against the baseline configuration with per-instance alignment features that was described in Chapter 5.

Scoring the target contexts is computationally expensive. As a result, we did not jointly learn all model weights. Instead, we started with the weights learned for *Cunei Baseline*. We then experimented with two configurations that enabled target context. In the first configuration, training was only permitted to adjust the new target context features. In the second configuration, training was also permitted to adjust the language model feature. The rationale for this second configuration was that the language model was the only other source of knowledge that we expected to overlap with the target context features. Limiting which features could be updated ensured a quick convergence and made training reasonable given the increased duration of translation. It is likely that joint training of all features would result in slightly better scores, but we could not do so in a reasonable amount of time.

In Chapter 5, where we described the experimental setup, we discussed running two training runs and reporting scores from the system with the maximum BLEU score on the development set. However, the results for *Target Context (No Modified Weights)* and *Target Context (Modified LM Weight)* are from a single run. Each run took over a week and only showed marginal gain; we did not spend further time and effort on a second run. This prevents us from calculating statistical significance over the training runs, because there is only one run. We do, however, still apply bootstrap resampling to calculate the statistical significance of BLEU.

7.4.1 Analysis

BLEU, NIST, Meteor, and TER scores on the development and test sets are presented in Tables 7.2 and 7.3. The results show a small but consistent gain across both language pairs. When modifica-

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	31.61	6.772	53.09	53.26	30.76	7.212	52.49	53.85
+Target Context (No Modified Weights)	31.96‡	6.770	52.92	53.33	30.87	7.214	52.20	53.92
+Target Context (Modified LM Weight)	31.74	6.755	52.97	53.33	31.02†	7.228	52.44	53.75

Table 7.2: Evaluation of Incorporating Target Context on CzEng v0.9

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	27.50	6.659	51.78	58.85	25.76	6.675	52.13	59.45
+Target Context (No Modified Weights)	27.52	6.657	51.76	58.75	25.83	6.680	52.07	59.42
+Target Context (Modified LM Weight)	27.84†	6.652	51.61	58.66	25.95	6.678	52.15	59.43

Table 7.3: Evaluation of Incorporating Target Context on German Europarl v6

BLEU scores with † are statistically significant over random sampling compared to the *Baseline* at $p = 0.05$; ‡ at $p = 0.1$

tion of the language model weight was permitted, we saw a 0.13 BLEU gain on the Czech-English and 0.34 BLEU gain on the German-English development set. Furthermore, this gain held at 0.26 BLEU on the Czech-English and 0.19 BLEU on the German-English test set. Movement in NIST, Meteor, and TER is minimal, but the general pattern in both language pairs is that these three additional metrics degrade on the development set and improve on the test set.

These gains are clearly smaller than the cumulative gain we reported in Chapter 6 from integrating all forms of source context. However, we are only modeling one form of context on the target in contrast to the many approaches we explored on the source. In fact, these gains in the range of a quarter BLEU point are consistent with the gains we saw using each individual form of source context. The intra-sentential features, specifically, were actually more useful on the target in Czech-English as the source features only yielded a gain of 0.15 BLEU on the test set. However, in German-English the intra-sentential features on the source, with a gain of 0.87 BLEU, strongly outperformed the intra-sentential features on the target.

7.4.2 Examples

Several examples from the Czech-English and German-English datasets are presented in Appendix A that compare *Cunei Baseline* to a new configuration incorporating target context (and allowing modification of the language model weight during training). Changes to the output are minimal as all other features are fixed and the new target features are similar to the existing language model feature. In fact, 863 out of the 1,506 Czech-English sentences and 242 out of the 910 German-English sentences are identical. These examples are, therefore, a sample of the sentences with the most substantial changes.

As with source context, one of the primary changes we see when incorporating target context is improved lexical selection. This is evident in Examples 22, 24, 25, 27, 32, and 33. Example 24 is from a technical document discussing DNS domains. The baseline system states the DNS domain was “amended” whereas the system incorporating target context refers to the DNS domain as being “changed”. While these are semantically equivalent, one usually amends legislature or legal documents; it is awkward in the context of this sentence. A more obvious improvement can be found in Example 27 which discusses roman numerals. The baseline system uses the phrase “large roman numerals” whereas the *Target Context* configuration generates “capital roman numerals”. Once again, both are semantically equivalent, but “capital” is the proper term when referring to letters of the alphabet.

However, the modifications from the target context are not confined to lexical selection. Unsurprisingly, translations from *Target Context* tend to be more fluent. Examples 22, 23, 28, and 31 highlight the insertion of missing words and removal of disfluent words. The word “element” in the phrase “depends on the element of surprise” is missing in the baseline system in Example 22. Conversely, in Example 31 *Cunei Baseline* includes the extraneous word “companies” in the phrase “efforts to reduce emissions companies”. Both of these errors are corrected when target context features are enabled. In addition, improved fluency through reordering is shown in Examples 26, 29, 32, and 33. The use of target context reorders “average distance values data” to “average distance data values” in Example 26. This phrase could still use a preposition, but at least the word ordering now conveys the appropriate meaning.

Most of these examples do show improvement, but Example 30 illustrates a situation where the target context harmed the translation. Instead of the phrase “the public access to information”, the *Target Context* configuration generates “the access of the public to the information”. In this case, no information is actually lost, but the rendition is extremely verbose.

	Czech-English		**German-English**	
	Runtime in Minutes	Words per Minute	Runtime in Minutes	Words per Minute
Cunei Baseline	36.95	500	106.13	252
+Target Context	391.45	47	923.86	29

Table 7.4: Runtime on Test Sets of Incorporating Target Context

7.4.3 Runtime

Scoring the target context features during decoding degraded the runtime. We compare Cunei's runtime with and without target context in Table 7.4. These timing experiments follow the same setup as described in Section 5.1.5 of Chapter 3, except *Target Context* is provided a JVM with 10GB of memory (instead of 5GB). If the target contexts are diverse, much of the information from the translation instances must be retained during decoding and not discarded. We initially ran the timing experiments with 5GB of memory, but Cunei ran out of memory when translating the German-English test set. Cunei ran successfully with target context enabled after we doubled the memory allocation and we did not explore intermediate values. Overall, we observed approximately a tenfold reduction in the number of words per minute that Cunei was capable of translating.

7.5 Scoring Target Similarity Prior to Decoding

Before the work presented above, we explored whether it was useful to score target similarity independent of the target hypothesis constructed during decoding. As presented in Section 7.3, scoring the target context adds significant complexity to decoding as the score for a phrase pair cannot be fully calculated until the target hypothesis is available. Instead, to score target similarity, we sought to compare the target phrase of the translation instance to other target hypotheses *for the same span of the lattice*. Essentially, we scored phrase pairs based on their similarity to other phrase pairs in the lattice. This did increase the complexity of scoring each phrase pair, but it meant that decoding could proceed as normal.

When constructing phrase pairs, we have thus far made the assumption that that each lexical sequence of words in the target phrase is distinct. While the collection of translation instances extracted 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 similar in that they share significant semantics and may only differ by one or two words. For example, we would like the similar, but inexact, translation instances for "the car" and "in the auto" to both contribute to the score of a phrase pair with target hypothesis "in the car". The score for a phrase pair, Equation 3.4 on page 35, already allows for features of this form as the summation is defined over all instances of translation. However, as our baseline configuration did not include target similarity features, we previously limited scoring a phrase pair to the summation over instances of translation with the same target phrase. Incorporating additional target similarity features and summing over all instances of translation does increase the amount of computation, but not by as much as performing the summation during decoding, as necessitated by the target context features.

The similarity between two target phrases was initially assessed simply by taking the F_1 scores of 1-grams, 2-grams, 3-grams, and 4-grams. The F_1 scores of 1-grams were not very meaningful as most words occurred in multiple hypotheses (recall that Cunei's phrase alignment finds many alternatives that differ by one word). To compound matters, the F_1 scores for 2-grams, 3-grams, and 4-grams were then too sparse. We also explored weighing the F_1 scores by the inverse frequency of a word in the training corpus to minimize the effect of stop words, such as "the" and "of", on the similarity measure.

We also explored scoring the target similarity of phrases based on their target-side distributional profiles. The distributional profiles use the surrounding context in the training data (as opposed to the hypothesis constructed during decoding) to identify similarity between phrases. When Cunei retrieves each instance of a translation from the corpus, the entire target sentence in which it resides is available. With the complete target sentence of a translation instance, we formed a distributional profile. This distributional profile was represented as a series of count vectors for n-gram phrases that immediately preceded or followed the phrase and lone "trigger" words that appeared anywhere in the sentence. Before scoring a phrase pair, we compiled a distributional profile for the phrase pair from the count vectors of translation instances that had the same target phrase. We could then calculate the cosine distance and Jensen-Shannon distance between the distributional profile of the candidate phrase pair and the distributional profile of each instance of translation. These were integrated as new target similarity features, and each phrase pair was scored by summing over all instances of translation, including those with different target phrases.

Unfortunately, neither method showed improvement. In fact, both methods resulted in slightly lower evaluation scores, and they exacted a significant cost in computation. We were hoping to use simple, corpus-based metrics to compute the similarity scores, but it is possible that we could have obtained more traction using an external resource like WordNet (Miller, 1995), which we did not explore. Instead of continuing down this path, we decided to compute the target similarity scores using the context of the constructed target hypothesis instead.

7.6 Summary

We find that modeling target context, while computationally expensive, is both possible and yields improved translation performance. The gains we saw of 0.26 BLEU on the Czech-English and 0.19 BLEU on the German-English test set are small but comparable to the gains from each individual form of source context in the previous chapter. When discussing source context in Chapter 6, we presented four different types of context that were efficient and practical to use in a real-world system. In contrast, this chapter used much more computation to model only one type of context. We recognize that target context is likely not practical for a real-world system. However, it is helpful to understand and explore target context in this dissertation as the counter-part to source context.

Chapter 8

Incorporating Corpus Annotations

In this chapter we extend the work presented in previous chapters by providing additional information for Cunei's instance-based model to score. Specifically, we augment the corpus with multiple forms of annotations. These annotations differ from the sentence annotations in Chapter 6 in that they describe a subsequence within the source or target sentence. Specifically, we explore embedding word clusters and parse trees as annotations. These annotations allow us to move beyond modeling lexical tokens in isolation. We expect that enriching the training data in this manner will yield a more robust and a more accurate translation model.

We are able to conveniently integrate corpus annotations throughout the entire translation process as a result of Cunei's modeling approach. Recall, Cunei scores the relevance of each translation instance present in the training data. The corpus annotations simply enhance the existing per-instance scoring. In one respect, the annotations can be viewed as another form of context that aids in disambiguation and refinement. We used the annotations to better score each translation instance, but fundamentally they are not a requirement. Corpus annotations may be present, as available, on the source and/or the target of the parallel corpus.

We both present new features and extend existing ones to operate over these annotations. On the source we extend the existing context features with annotations and create new features to model divergences between the translation instance and the input sentence. In scoring the correspondence between the source and target phrases, the annotations contribute to the existing lexical and translation probability scores. On the target we extend the existing target context and language model features. Annotations also allow for generalization and enable the construction of translation templates with gaps. When a gap occurs, we add new features to identify which translation instances are most relevant for substitution based on the source and target annotations. The Figures 1.1 and 1.2 from the introduction decomposed the process of scoring a translation instance into three components. In sum, the incorporation of annotations influences all of them: $f_S(s, x)$, $f_X(x)$, and $f_T(x, t)$.

8.1 Early Work with Part-Of-Speech Labels

An early incarnation of Cunei that employed part-of-speech labels was presented in (Phillips, 2007). In addition to the standard lexical matching, we used the part-of-speech labels 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, the gaps were filled with other lexical translations to compose a new, synthetic translation. In addition to respecting the

	Dev	**Test**	**Partial MT04 by Genre**			
	MT03	**MT04**	**News A**	**News B**	**Editorial**	**Speech**
Lexical Baseline	44.4	39.7	48.3	45.5	32.1	33.9
Part-of-Speech	45.2	41.2	49.0	47.5	32.9	36.4
Lexical Baseline (Monotone)	41.9	38.5	46.1	43.4	32.0	33.3
Part-of-Speech (Monotone)	44.6	41.1	49.0	47.0	33.3	36.3

Table 8.1: Evaluation of Incorporating Part-of-Speech on Arabic-English in BLEU (Phillips, 2007)

word alignment, the substitution was also required to match the target part-of-speech sequence. In this manner, the part-of-speech matches identified templates that guided local reordering.

To evaluate this setup, the predecessor to Cunei[1] 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 learned using part of the 2003 NIST MT Evaluation dataset (MT03); the 2004 NIST MT Evaluation dataset (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 8.1 illustrates the gain in performance from annotating the corpus with part-of-speech labels. It is clear that the part-of-speech 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 part-of-speech matching. This suggests that the part-of-speech templates make the system more robust and allow it to degrade more gracefully. We also experimented with monotone decoding (no reordering) which, as expected, lowered performance of the system with only lexical matching. Interestingly, this was not true for the system with part-of-speech matching enabled, signifying that the templates captured most (if not all) of the reordering.

The work described above allowed for gaps but was limited to one form of annotations (part-of-speech labels) that had a 1:1 correspondence with the sequence of lexical tokens. This early work also placed hard constraints (equivalent part-of-speech labels) on the gap-filling process. The more recent work presented in the next two sections seeks to address both of these limitations.

8.2 Types of Corpus Annotations

We present two types of corpus annotations below: sequential annotations and hierarchical annotations. Conceptually, both types convey the same information, but they have different capabilities and limitations. For computational efficiency, they are indexed and scored separately.

[1] This system followed a similar approach to the work in this dissertation, but it was a substantially different implementation.

8.2.1 Sequential Annotations

We begin by exploring sequential annotations which extend the representation of a word from a simple lexical token to include one or more annotation labels. Sequential annotations represent an 'alternate' perspective of the data that has exactly the same segmentation and number of tokens as the lexical sequence. They are simple and can be processed in much the same manner the lexical tokens. We will allow multiple sequential annotations to simultaneously be specified on a phrase. The main purpose of these annotations is to be able to construct new similarity calculations. However, in Section 8.3 we will explain how they can also assist Cunei in retrieving similar but inexact instances of translation. The sequential annotations in our corpora are formed by word clusters and part-of-speech labels.

Factored models described by Koehn and Hoang (2007) and implemented in Moses use the same representation as sequential annotations, but the role of the information is different. In a factored model, the user specifies the dependencies for each factor and builds independent models for generating or translating the factors. The factored approach defines a specific structural model and further fragments the translation probabilities. In Cunei, we use the additional word labels for scoring the relevance of each translation instance. The annotations serve as an additional resource, but depending on the model weights they can also be completely ignored. The key idea of this approach is that a translation instance whose annotations exactly match the input sentence will be more relevant than a translation instance whose annotations only partially match the input sentence.

8.2.2 Hierarchical Annotations

Sequential annotations are restricted to information that can be associated with individual words in the corpus, but there is a wealth of information that cannot be easily described in this manner. For example, we may wish to denote named entities or linguistic constituents that span multiple words. Furthermore, we may require annotations that are hierarchical in nature to represent a parse tree or a dependency graph. Thus, we define a new type of corpus annotation that is associated with one or more words and optionally references a parent annotation. Just as we permitted multiple sequence annotations, we also permit multiple types of hierarchical annotations to be specified on a phrase.

As was the case with sequential annotations, we will use hierarchical annotations to model the relevance of each translation instance. It is important to point out that this is quite different from the role that parse trees usually serve in translation. Unlike a system such as Joshua (Li et al., 2009), we do not learn a synchronous context-free grammar that describes the translation process. Furthermore, we do not explicitly model the probability of generating the current parse tree nor do we use the parse tree to restrict which translation instances can be combined. Parse trees and all other forms of hierarchical annotations present alternative representations of the data and provide additional context. Cunei uses the annotations to score similarity features that compare each instance of translation to the input sentence or target hypothesis.

8.2.3 A Simple Example

A parsed German phrase is presented in Figure 8.1. The terminal nodes, colored in blue, are the lexical tokens and one level up, colored in green, are part-of-speech labels. These part-of-speech labels have a 1:1 correspondence with the lexical tokens, so they are indexed as a sequential annotation. Above each part-of-speech label is one or more constituent labels, colored in red. These

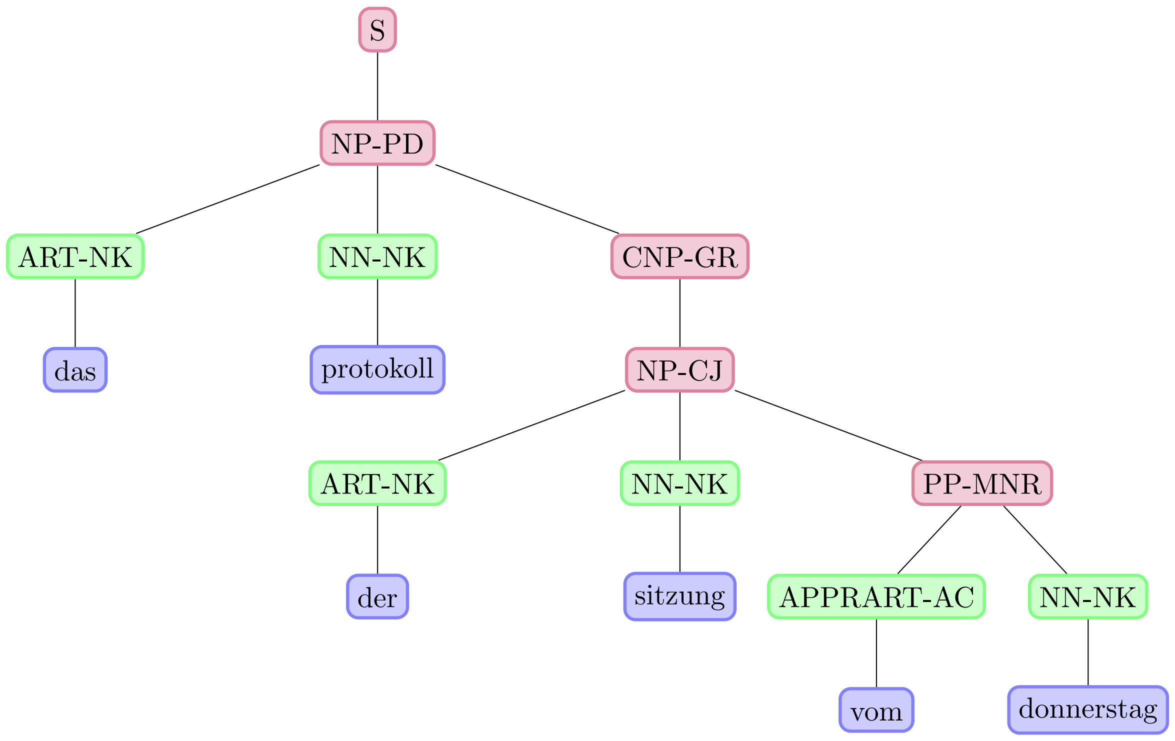


Figure 8.1: A German Parse Tree

constituent labels are permitted to span multiple tokens and are, therefore, indexed as hierarchical annotations.

8.3 Extending Cunei with Annotations

8.3.1 Locating Matches

The addition of annotations to the corpus provides a new mechanism for locating instances of translation. In particular, we describe below how Cunei is able to leverage sequential annotations to retrieve instances of translation that are similar but not equivalent to a subsequence of the input sentence. The hierarchical annotations, while scored on each instance of translation, are not currently used for locating instances of translation.

Cunei represents each sentence in the corpus and the input document as the union of multiple sequences. An example of this is presented in Figure 8.2. Minimally, a sentence contains a lexical sequence representing the unaltered surface strings (or less precisely, 'words'). In this abstraction, sequential annotations are merely another sequence. As stated in Section 8.2.1, sequential annotations specify one token for every position in the corpus. Each sequential annotation can be considered a sequence of tokens in parallel with the original lexical tokens. Lemmas, part-of-speech,

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 8.2: A Sentence Represented with Multiple Sequences

or statistical cluster labels can all be used as additional sequences that 'annotate' a sentence.

When the parallel corpus is described with multiple sequences, Cunei constructs a separate suffix array index for each sequence. We described the use of suffix arrays for indexing lexical tokens previously in Section 3.2.1 of Chapter 3. Cunei uses exactly the same methods to store each sequential annotation. This makes it possible for Cunei to query the training corpus for specific annotation sequences. For example, consider using a part-of-speech sequence to retrieve instances of translation that are structurally similar to the input, but semantically unrelated. They may not as-is provide valid translations, but they do still contain useful information about the translation process. Also consider Figure 8.4 on page 107. If we could only search by the lexical sequence "koukni se na toile", we would not be able to locate the translation instance for "podívej se na tady" that, while inexact is quite similar to the phrase we are translating (this is discussed in more detail later).

In Section 3.2.1 of Chapter 3 we also discussed locating matches in the corpus for any subsequence of the input sentence. We assumed in this description that the corpus was represented with one suffix array. When the corpus includes sequential annotations, Cunei simultaneously retrieves matches against any sequence. For each type of sequence present in the input sentence, the respective suffix array for that sequence is queried for matches. Algorithm 1, presented previously on page 38, is now executed for *each sequence*. The matches Cunei identifies in the corpus are the union of the matches found for each sequence. This gives rise to a problem we address in the next section: the matches we identify in the corpus may not exactly match the input across all sequences

8.3.2 Sampling Matches

We discussed in Chapter 3 that sampling is necessary to maintain reasonable performance and in the default configuration Cunei performs a uniform sample of corpus matches. We also investigated an alternate approach to sampling in Chapter 6 that scored each match, but it did not demonstrate much improvement. However, selecting the best scoring matches is necessary when we represent the corpus with multiple sequences. The role of sampling is suddenly far more important because many of the matches we find do not exactly match the input across all sequences.

Independently searching the training corpus across multiple sequences will locate lots of matches in the corpus. However, many of these matches are abstractions and not equivalent to the input across all sequences. The sheer number of matches typically saturates the sampling threshold, so Cunei is forced to select a subset of matches for continued processing. With the default uniform sampling, no preference is given to any type of match. A match across all sequences is ranked the same as one that matches an annotation sequence but not the lexical sequence. The whole point of finding abstractions is that they may be useful to the translation process, but they are clearly not as useful as an exact match. Unfortunately, a uniform sample throws out many exact matches in order to accommodate the inexact matches. In fact, we may be left with only abstract, inexact matches even if exact matches were found in the corpus.

The result is that Cunei *must* apply the approach that we initially investigated in Section 6.4 of Chapter 6 and sample the collection of matches based on their partial scores. The partial score of a match consists of the features that can be computed prior to alignment when only the source phrase and the location in the corpus is known. We will be presenting new similarity features in Section 8.4.1 that score sequential annotations, hierarchical annotations, and the original lexical sequence. These features identify divergences by comparing annotations or lexical tokens at a specific location in the corpus to the input sentence. We instruct the sampling algorithm to score matches with these features, thereby giving preference to the most similar (and hopefully exact) matches. As presented previously in Algorithm 3 on page 40, this biased sampling technique selects the n-best scoring matches, where n is the size of the sample. Thus, when using multiple sequences we will allow for matches that diverge from the input across one or more sequences, but only when there are not sufficient exact matches in the training data.

8.3.3 Identifying Divergences and Creating Gaps

When we locate a match according to one sequence type, we are able to use the positional information we stored in the corpus (see Section 3.2.1 in Chapter 3) to retrieve all other sequence types present in the corpus at that position. The instances of translation that Cunei constructs after phrase alignment include all sequential and hierarchical annotations that were defined for the position in the corpus each instance represents. However, if we originally found a match in the corpus based on the annotation sequence, then the corresponding instance of translation may lexically diverge from the input. We identify if divergences exist by comparing each sequence of the translation instance with the input sentence. When an instance of translation does diverge from the input, we can either use it as-is or mark the divergent region(s) as gaps requiring replacement.

The simplest option is to use all translation instances as-is, regardless of whether they exactly match the input. When an instance of translation does diverges from the input, the quality of the target phrase is in doubt. It may provide a perfect translation, but more likely it is at best a paraphrase and at worst semantically unrelated. Cunei signifies that these translations are less relevant by including features that score how divergent each instance of translation is from the input sentence. In particular, we will define similarity features in Section 8.4.1 that score sequential annotations, hierarchical annotations, and the original lexical sequence. In general, an instance of translation with divergences will be assigned a lower score than an instance of translation that exactly matches the input.

The second option is that the structure of the original translation can be preserved by marking divergent regions in the translation instance as gaps requiring replacement. A lexical divergence is a strong indicator that we should create a gap, but when a translation instance only diverges according to an annotation sequence, the decision is less clear. As a result, a divergent instance of translation can generate multiple template instances. Each template instance is an instance of translation in which specific regions are marked as gaps requiring replacement. These template instances are constructed with Algorithm 5 that identifies possible regions for gaps in a translation instance.

The resulting template instances are scored individually along with all other instances of translation. The distinction is that the region defined by a gap in a template instance is not scored; it represents a hole. The region outside the gap is scored by all the same features that score each instance of translation. It is possible for a template instance to specify a gap in one region and still be divergent from the input in another. If a divergence occurs outside a gap, it will be scored like any other divergence in an instance of translation. The score for the region defined inside the gap is assigned during decoding, as discussed next, once a replacement has been made.

The region defined by a gap in a template instance is projected to the target phrase during phrase alignment. Knowledge about the gap, such as part-of-speech labels, is preserved in order to aid selection of a valid replacement. In this manner, the template instances may identify discontiguous phrases, but they also may not follow linguistically-motivated paradigms. As such, template instances allow for the formation of novel phrases, but they also add risk. To balance this risk, the number of gaps in a template instance is identified with a feature. Generally, Cunei will assign a negative weight to this feature in order to prefer phrasal translations, if they are present.

Normally, our model sums over translation instances to score phrase pairs. Because they include gaps, template instances are not used to score phrase pairs. Instead, template instances score

Algorithm 5: Generating Templates Instances

```
Initialize all elements in array anchors to False
foreach type do
    Initialize all elements in array $t$ to False
    for $p = 0$ to $|\text{match}|$ do
        if match diverges from input in sequence type at position $p$ then
            $t[p]$ = True
        end
    end
    for $p = 0$ to $|\text{match}|$ do
        $\text{anchors}[p] = \text{anchors}[p] \wedge t[p]$
    end
    Add $t$ to set templates
end
foreach template $t$ in set templates do
    $q$ = False
    for $p = 0$ to $|\text{match}|$ do
        $q = q \vee \text{anchors}[p] \wedge t[p]$
    end
    if $q$ = True then
        instances = PhraseAlignment(match, $t$)
        Append instances to result
    end
end
return result
```

translation templates. Translation templates are phrase pairs that additionally store the location of gaps and any corpus annotations specified over each gap. A translation instance is to a phrase pair as a template instance is to a translation template.

8.3.4 Combining Translation Units

As described in Section 3.2.5 of Chapter 3, Cunei constructs a lattice of translation units as an intermediary step prior to decoding. When the corpus contains sequential annotations, this translation lattice may now also contain translation templates in addition to complete translation units. When processing translation templates, the decoder must fill the gaps to form a complete target hypothesis. Instead of simply concatenating two abutting target hypotheses, the translation template specifies where the decoder should insert new target hypotheses (to fill the gaps). The process of combining the translation template with a previously composed hypothesis is shown in Figure 8.3. The score for the translation template is combined with the scores for the hypotheses that were used to fill the gap(s). The result is a complete target hypothesis that is processed by the decoder in the normal fashion.

Recall from Section 3.2.5 of Chapter 3 that one method for combining phrase pairs is chart decoding. As previously described, the decoder stores possible target hypotheses in a chart. The decoder constructs new hypotheses either from phrase pairs in the translation lattice or by combining two adjoining hypotheses in the chart. Hypotheses corresponding to one source word are

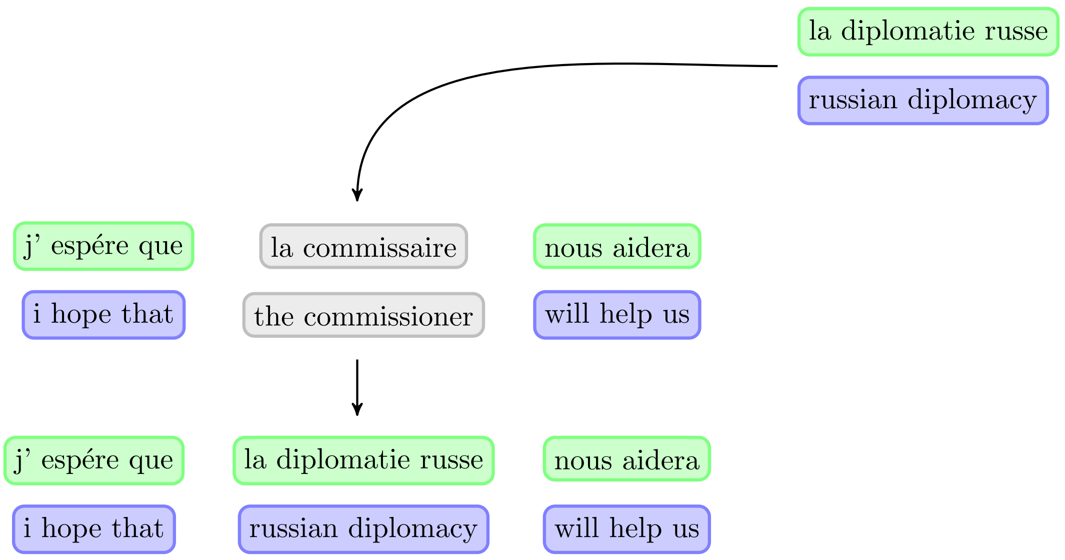


Figure 8.3: Filling a Gap in a Translation Template

constructed before those that correspond to two source words, which are in turn constructed before those for three source words, and so on. This guarantees that by the time we encounter a translation template with a gap, there will already exist hypotheses that can be used to fill the gap(s). As currently implemented, translation templates can only be filled by the chart decoder. When the translation lattice includes translation templates, Cunei will not switch to beam decoding until all translation templates have been processed.

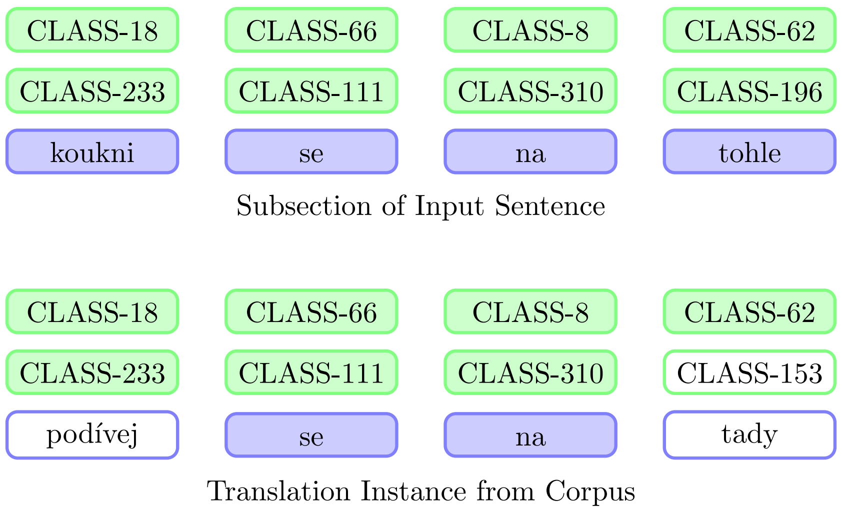


Figure 8.4: Comparing Sequential Annotations on a Translation Instance to the Input Sentence

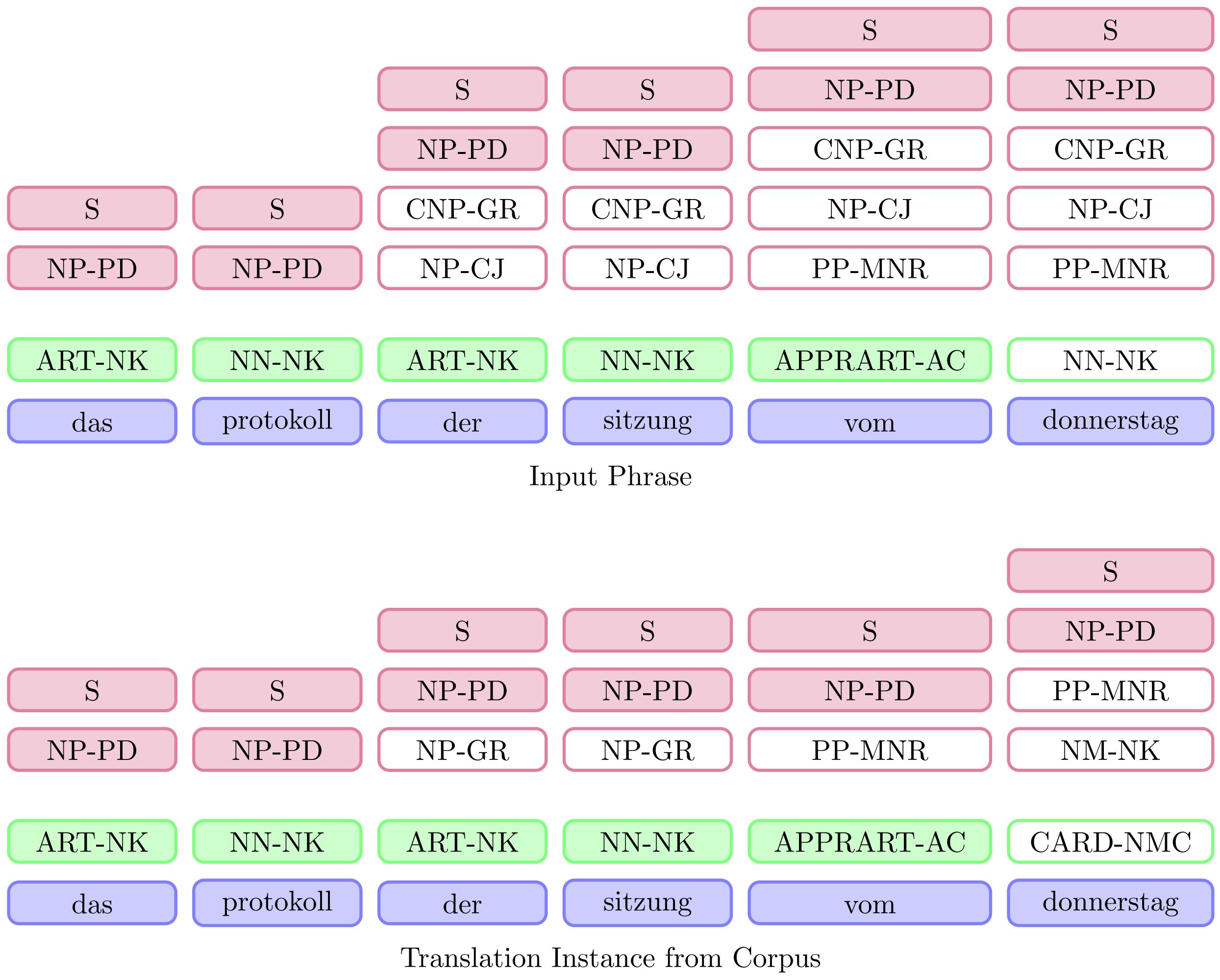


Figure 8.5: Comparing Hierarchical Annotations on a Translation Instance to the Input Sentence

8.4 Scoring Corpus Annotations

8.4.1 Input and Replacement Similarity

We use the annotations on the source to determine how similar a retrieved translation instance is to the input sentence. An example of two Czech phrases with sequential annotations is presented in Figure 8.4. The first lexical token on the translation instance differs from this section of the input sentence, but it is also labeled with the same set of annotations, suggesting that it may still convey similar information. In fact, the Czech words "koukni" and "podívej" can both be translated into English as "look". However, at the end of the phrase we find both one of the annotations and the lexical sequence diverging. Here "tohle" and "tady" serve a similar function in the phrase, but their translations do differ ("this" vs. "here").

Similarly, Figure 8.5 presents how the hierarchical annotations are represented when scoring a translation instance. Note that hierarchical annotations are stored over spans of words, but they are illustrated as labeling each word because we will be computing the similarity of hierarchical annotation labels at each word position. The subsequence of the input sentence being translated in Figure 8.5 corresponds to the parse tree previously presented Figure 8.1. The translation instance

Let the instance of translation and the section of the input sentence being evaluated each contain n tokens.

Sequential Annotations
The feature below is specified for the annotation type `POS`, but this feature will be created for all sequential annotations known to the system.

$$\delta(i) = \begin{cases} 1 & \text{if the } i^{\text{th}} \text{ tokens are equal} \\ 0 & \text{otherwise} \end{cases}$$

`Match.Weights.POS.Divergence`	$\frac{1+\sum_{i=0}^{n}\delta(i)}{1+n}$

Hierarchical Annotations
Let A be the set annotations from the corpus that correspond to the translation instance and A' be the set of annotations for the section of the input sentence being evaluated. We will use $A_X(i)$ to represent the subset of annotations in A of type X at position i. The feature below is specified for the annotation type `Parse`, but this feature will be created for all hierarchical annotations known to the system.

`Match.Weights.Parse.Divergence`	$\sqrt[n]{\prod_{i=0}^{n} \frac{1+\lvert A_{\texttt{Parse}}(i) \cap A'_{\texttt{Parse}}(i)\rvert}{1+\lvert A_{\texttt{Parse}}(i) \cup A'_{\texttt{Parse}}(i)\rvert}}$

Table 8.2: Description of Annotation Similarity Features

has a similar parse tree, but it contains a few divergences that are highlighted by annotations with a white background. For example, “der sitzung” is labeled with a `NP-GR` constituent in the translation instance instead of `NP-CJ`.

Restricting the annotations on the input sentence and the translation instance to match exactly will produce high precision translations but significantly reduce the number of possible translation instances we are able to retrieve from the corpus. Instead, we allow inexact matches and score the similarity of each type of annotation in comparison to the input sentence. We view annotations on the source as another type of context that is complementary to those presented in Chapter 6.

To score the similarity between two sequential annotations, we use the feature presented in the top half of Table 8.2. This feature is calculated by determining how many tokens in the two sequences are labeled with the same annotation. In addition, we avoid a score of zero by adding one to the numerator and denominator. Therefore, the middle annotation sequence in Figure 8.4 has a similarity score of $\frac{4}{5}$, whereas the top annotation sequence is a perfect match.

Like the similarity calculation for sequential annotations, we score hierarchical annotations based on the accuracy of annotations. The difference is that hierarchical annotations may span multiple tokens, so we assess how many annotation labels are equal at each position in the phrase. Two annotations are considered equal if they have the same value and their parents are also equal. Thus, when we compare two sets of hierarchical annotations we evaluate the trees from top to bottom, checking how far down the hierarchy there exist equivalent annotation labels. In Figure 8.5, the hierarchical annotation labels for the first two words match exactly, but starting at the third word position there exist only partial matches. Two out of five unique annotation labels match for the third word position; we add one to the numerator and denominator and assign this position in the phrase a score of $\frac{3}{6}$. Even though the fifth and sixth words are both labeled with the constituent `PP-MNR`, the labels occur at different levels in the hierarchy so they are not considered a match. As shown in the bottom half of Table 8.2, the cumulative score for a phrase is the geometric mean of the similarity scores at each position in the phrase.

In addition to scoring the similarity of the source annotations relative to the input sentence, we score the similarity of both the source and the target annotations when filling a gap. A gap is identified in a translation instance when some subsection that diverges from the input sentence is elided, as was described in Section 8.3.3. We score possible replacements for the gap by comparing the similarity of the annotations defined on the gap and on the replacement. Recall, any types of annotation that match the source sentence are carried over to the target during alignment, generating a template instance. For example, we may define a gap over the last token in Figure 8.4 because both the middle annotation sequence and the lexical token differ from the corresponding section of the input sentence. However, because the top annotation sequence matches the input, we still expect the corresponding annotation sequence to be valid on the target. We therefore have some information about what annotations should be specified on a replacement. While we score both source and the target annotations, the target annotations are more likely to provide novel information as the source annotations are already scored on all translation instances to be maximally similar to the input sentence. We model gap replacement with separate features but use the same similarity calculations presented in Table 8.2. These features guide the replacement process and are intended to favor substitutions that are most compatible.

8.4.2 Source and Target Context

On the target-side, we cannot compute the similarity between the translation instance and the target because the actual target is unknown. However, during decoding we generate a set of possible target hypotheses which provide some context of what target to expect. The standard method for modeling the lexical tokens in the target is to use an n-gram language model that predicts the next token based on the history. We can apply this same approach to sequential annotations as we have an annotation label for every word position in the corpus. In this manner we replicate the language model feature from Table 3.3 to score the likelihood of each sequence known to the system.

In addition, we can also extend the source and target context features described, respectively, in Chapter 6 and Chapter 7 to operate over the annotation sequences. Specifically, when we model context based on surrounding tokens, we are no longer limited to lexical tokens. We replicate the intra-sentential and document context features defined in Table 6.2 to score the context of each annotation sequence on the source, and we replicate the intra-sentential context features defined in Table 7.1 to score the context of each annotation sequence on the target. When we replicate these source and target context features, we create a new feature *for each sequential annotation* that performs its calculation over annotation labels instead of lexical tokens.

8.4.3 Translation Probability

When sequential annotations are present on both the source and target, we can also estimate the likelihood of translating the source annotation sequence into the target annotation sequence. To be clear, these translation features are not novel; they correspond to one topology of a factored model.

However, as we already compute these scores over the lexical sequence, it is trivial to compute similar scores over an annotation sequence. Specifically, we add as features the lexicon probabilities and phrase frequencies defined in the top half of Table 3.2 on page 46 for each annotation sequence. Following the same approach to extending existing features, we simply replicate the features and score each sequence individually.

8.4.4 Missing Annotations

As stated in the introduction to this chapter, annotations are optional and the model does not depend on their existence. That being said, we still have to ensure that translation instances without annotations are scored comparably to translation instances with annotations. If we do nothing, translation instances from sections of the corpus that we did not annotate will never match the input sentence. These instances of translation will be penalized for diverging from the input sentence even though no annotations were specified. Instead, when no annotations are available, we assign all annotation similarity features to be a perfect match and enable a new binary feature to identify that the annotations are in fact missing. The missing annotations feature will likely receive a negative weight during training, but we identify the missing annotations separately from the similarity features so that their relative weights can be learned.

8.5 Experiments

In both Czech-English and German-English, we started with the baseline system described in Chapter 5 and experimented with three different approaches for using annotations. We assume that when annotations are present in the corpus, we should include all available features for scoring them. The choice is less clear, however, how permissive the system should be in modeling inexact translation instances. We evaluate the following three configurations:

1. **Annotations without Lexical Divergence**
   We limit translation instances to those that lexically match the input sentence. The additional annotations are still allowed to diverge from the input sentence and will be scored. This configuration is similar to *Cunei Baseline* augmented with the new annotation features, except the sampling has also been modified as described in Section 8.3.2.

2. **Annotations with any Divergence**
   Translation instances are allowed to lexically diverge from the input sentence. When divergences occur, the divergence will be scored and the translations will be used as-is. This configuration does not permit gaps.

3. **Annotations with Divergence & Replacement**
   Translation instances may diverge from the input sentence and any sections that diverge may be identified as gaps requiring replacement. Cunei is not required to construct a gap and may still allow a divergent translation to be used as-is depending on the score assigned by the model.

8.5.1 Czech-English Dataset

On the Czech-English dataset, we automatically create sequential annotation labels using unsupervised learning. For many non-mainstream languages, additional resources such as parsers or morphological analyzers are not available. A data-driven statistical approach has the advantage that it can easily be applied to any language pair.

The toolkit that we used for alignment, GIZA++, includes a program MKCLS for generating word clusters (Och, 1999). We run this command using the default algorithm which defines a map C from each word in the corpus to a class. It uses a variant of simulated annealing to maximize the likelihood estimation of the corpus according to the following equation:

$$\arg\max_{C} \prod_{i=1}^{N} P(C(w_i)|C(w_{i-1})) \cdot P(w_i|C(w_i))$$

For both Czech and English we learn 100 and 1000 clusters, providing two levels of granularity. Words that were either not present or only occurred once in the corpus were assigned to a single unknown cluster. These word clusters are then used to annotate the corpus and input document. The only modification to the training corpus is that it has been augmented with sequential annotations– the lexical tokens and word alignments have not changed. We use the same lexical language model, but also train two new 9-gram language models with Witten-Bell smoothing on the target annotation sequences. The increased history length and simpler smoothing algorithm is due to the fact that the size of the vocabulary is drastically reduced when modeling these annotation labels generated from word clustering.

8.5.2 German-English Dataset

To the German-English corpus, we added annotations from parse trees. We used the Stanford parser (Klein and Manning, 2003a,b; Rafferty and Manning, 2008) and built-in factored models to independently parse each side of the parallel corpus. In English we were able to parse all sentences with less than 50 words, but in German we were only able to parse sentences less than 25 words. The reason for this is that there are approximately ten times the number of part-of-speech labels in German as there are in English. As a result, the German grammar was larger and required far more memory. In total we were able to successfully parse 92% of the English sentences and 50% of the German sentences. The part-of-speech labels were indexed as a sequential annotation. The remainder of the parse tree was indexed as hierarchical annotations. As discussed previously, indexing the part-of-speech labels separately is more efficient and assists us in locating inexact matches.

Once again, the only modification to the training corpus is that it has been augmented with sequential and hierarchical annotations. Using the part-of-speech labels, we also add a new 9-gram language model with Witten-Bell smoothing.

8.5.3 Analysis

The BLEU, NIST, Meteor, and TER scores for all three conditions are presented in Table 8.3 for Czech-Englsh and Table 8.4 for German-English. The BLEU scores on both the development and test sets are consistently higher when we include annotations. Furthermore, *Annotations with Divergence & Replacement* was the best performer on the test set in both language pairs. However, the 2.11 BLEU gain on the test set in Czech-English was much larger than the 0.39 BLEU gain in German-English. This is the opposite of what we anticipated. We expected the parse tree information in German-English to be more useful than the statistical clusters we built in Czech-English. The discrepancy in score is likely a result of the fact that we were only able to fully annotate half of the German corpus.

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	31.61	6.772	53.09	53.26	30.76	7.212	52.49	53.85
+Annotations without Lexical Divergence	**33.37**†	*6.910*	53.91	**51.97**	*32.85*†	**7.362**	*53.29*	**52.59**
+Annotations with any Divergence	33.23†	6.889	53.99	52.08	*32.50*†	7.319	53.07	52.74
+Annotations with Divergence & Replacement	**33.56**†	*6.921*	*54.08*	*51.85*	**32.87**†	**7.354**	**53.47**	**52.68**

Table 8.3: Evaluation of Incorporating Annotations on CzEng v0.9

	Development Set				**Test Set**			
	BLEU	**NIST**	**Meteor**	**TER**	**BLEU**	**NIST**	**Meteor**	**TER**
Cunei Baseline	27.50	6.659	51.78	58.85	25.76	6.675	52.13	59.45
+ Annotations without Lexical Divergence	**28.22**†	6.621	51.71	58.58	*26.06*‡	6.604	**51.91**	59.76
+ Annotations with any Divergence	28.40†	6.665	51.72	*58.48*	26.08‡	*6.644*	52.06	59.60
+ Annotations with Divergence & Replacement	*28.36*†	6.668	51.75	**58.31**	*26.15*‡	6.641	**51.96**	59.40

Table 8.4: Evaluation of Incorporating Annotations on German Europarl v6

Scores in bold are statistically significant over multiple runs compared to the *Baseline* at $p = 0.05$; italics at $p = 0.1$
BLEU scores with † are statistically significant over random sampling compared to the *Baseline* at $p = 0.05$; ‡ at $p = 0.1$

	Czech-English			**German-English**		
	No Lexical Divergence	With any Divergence	Divergence & Replacement	No Lexical Divergence	With any Divergence	Divergence & Replacement
1-gram	97.83	97.84	97.84	99.87	99.87	99.87
2-gram	72.19	95.72	95.72	94.65	94.64	95.90
3-gram	36.73	85.55	88.19	72.65	72.60	78.95
4-gram	18.14	50.24	55.59	43.81	43.75	55.18
5-gram	10.84	22.83	25.84	22.05	22.01	34.60

Table 8.5: Percent of n-grams in Test Set with at Least One Translation Unit when Incorporating Annotations

	Czech-English			**German-English**		
	No Lexical Divergence	With any Divergence	Divergence & Replacement	No Lexical Divergence	With any Divergence	Divergence & Replacement
1-gram	170.14	208.32	204.52	347.60	352.58	344.86
2-gram	100.41	220.31	304.64	289.15	302.11	298.03
3-gram	51.35	96.74	218.26	178.87	184.06	204.71
4-gram	24.78	44.48	89.09	101.99	103.13	134.86
5-gram	12.05	22.20	40.93	67.31	67.31	94.60

Table 8.6: Average Number of Translation Units per n-gram in Test Set when Incorporating Annotations

In Table 8.5 and 8.6 we provide statistics describing the coverage and density of the translation lattices. For configurations that disallow lexical divergences, divergent phrase pairs still exist in the lattice but with such low scores that they are never selected by the decoder. To calculate these statistics, we filtered all translation units with a score less than -100. However, this did cause small fluctuations in the counts; it is the reason we report slightly fewer translation units per 1-gram in German-English when replacements are allowed. In general, these translation lattices are quite large because the annotations enable exact phrase pairs, divergent phrase pairs, and translation templates. Allowing divergences increases both coverage and density in Czech-English, but it does not affect the translation lattice as significantly in German-English. Allowing replacements slightly increases coverage, but mostly results in the addition of many new translation templates.

8.5.4 Examples

In Appendix A we present examples of translations produced by *Cunei Baseline* and the three different methods for integrating annotations. Unlike the examples presented in Chapters 6 and 7, the changes between the baseline and the improved systems are not simply the result of adding a few features with default weights to the baseline. As discussed in Section 8.3.2, once we add annotations to the corpus and allow divergent translation instances, we must use a biased technique for sampling. We are unable to perform a minimal comparison when the sample of translation instances used to score each phrase pair is different. Instead, each configuration jointly learned a new set of weights over all features. This comparison, thus, leaves open the possibility that some of the changes are not due to the inclusion of annotations.

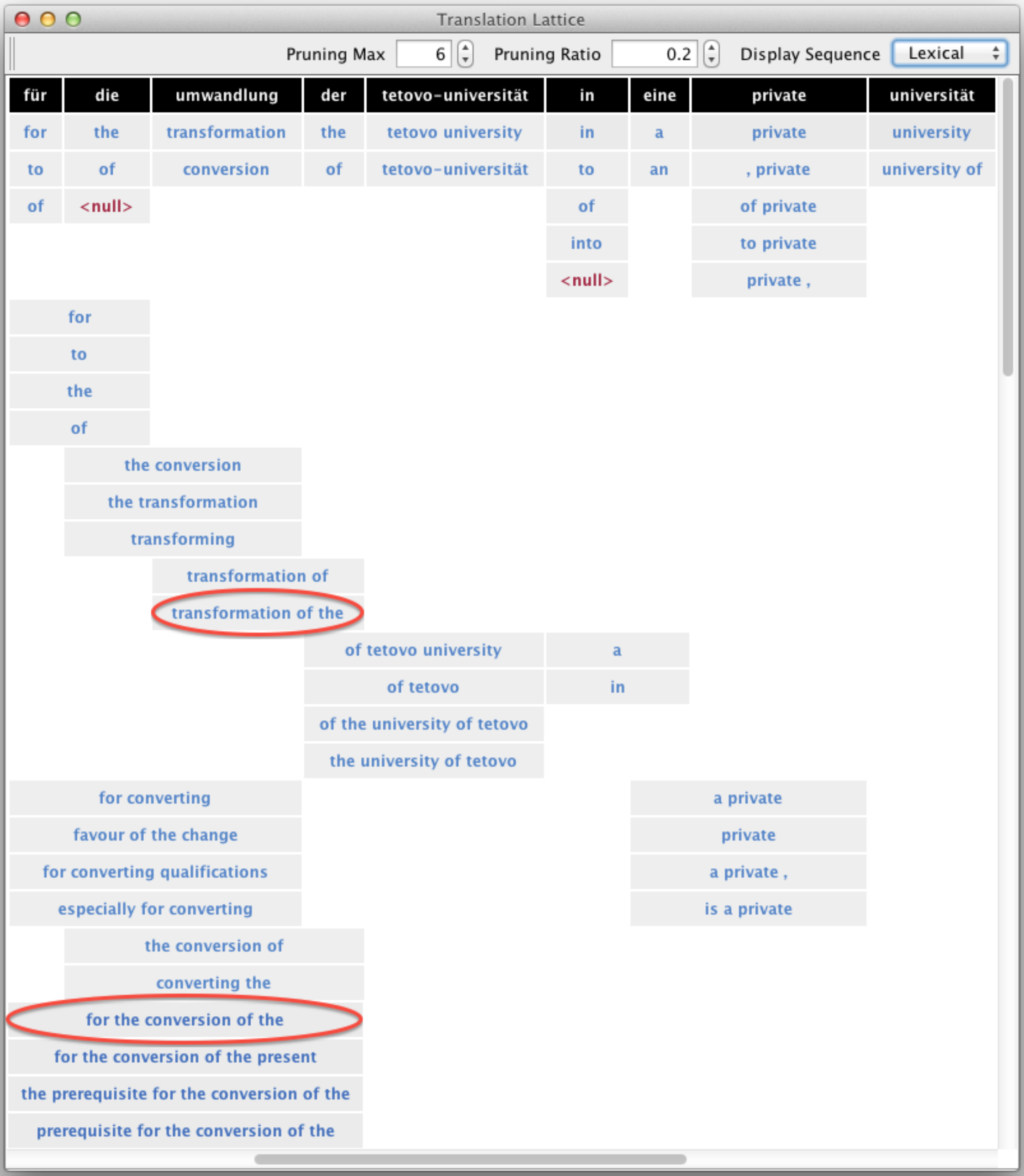
A common theme among the examples is that all three methods for exploiting annotations tend to produce output that is closely related and slightly different from *Cunei Baseline*. As the features predominately focus on the translation model, it comes as no surprise that the use of annotations frequently results in improved lexical selection. This is evident in Examples 34, 35, 36, 39, 40, 42, and 47. The switch from "film evaluation system" to "film rating system" in Example 40 is minor, but contextually more accurate. Preferring the phrase "in connection with" over "the context of" in Example 36 is more substantive, but it still does not change the underlying meaning of the translation. These types of improvements are similar to the changes we saw in Chapter 6 because one aspect of the annotations is that they provide additional contextual information.

A more detailed case of improved lexical selection is presented in Example 42, which discusses the institution "tetovo" becoming a private university. The reference refers to this process as "for the transformation of tetovo university" whereas *Cunei Baseline* provides "for the conversion of the tetovo university". Figures 8.6 and 8.7 highlight the relevant portions of the translation lattice for this example. The longest phrase pair in this section is for "für die umwandlung der" and no translations of it contain the word "transformation"; instead, the highest-ranked translation in the baseline configuration is, predictably, "for the conversion of the". Translations that include the word "transformation" can be found instead with the source phrases "die umwandlung der", "umwandlung der", and "umwandlung". These phrase pairs with select features that score the annotations are detailed in Table 8.7. Interestingly, the part-of-speech features are not that discriminative over these phrase pairs. Even worse, the feature that scores the parse tree is constant for these phrase pairs over each span of the input sentence. The reason for this is that both "the conversion of" and "the transformation of" represent the same syntactic role. Unfortunately, this means that the new annotation features do not cause many of the phrase pairs for this section of the input sentence to be re-ranked. Nonetheless, the configurations that score annotations correctly produce a translation that includes the word "transformation". The feature that scores the parse tree identifies that the longer phrases have a greater mismatch with the parse tree of the input sentence. Combining the top-ranked translations of "für", "die", "umwandlung", and "der" yields the correct output "for the transformation of the" and has the best combined score for the parse tree.

Sometimes the improved lexical selection also leads to a significant improvement in semantics. Figures 8.8 and 8.9 correspond to Example 35. This is a fairly simple sentence with the exception that, according to our model, "řádku" can be translated as either "row" or "line". In Table 8.8 we find the new features that score the annotation labels help disambiguate between these two possibilities. While *Cunei Baseline* prefers "row", the annotations change the rank of the correct translation "line" to be the top candidate. The feature values for both *Cluster 100* and *Cluster 1000* indicate that "line" is a better translation. In fact, all translations that include the word "line", such as "the line" and "line option", have very good scores for the source, but the additional, spurious words in the target phrase result in lower scores overall. Likewise, in Example 39 all annotation configurations identify the translation "product brands" instead of "product marks". One could perhaps guess what "product marks" are, but the meaning is not readily apparent.

Enabling divergent lexical sequences introduces more possible target translations. This can be advantageous when we have little evidence in the corpus for a word or phrase. In Example 44 *Annotations with any Divergence* is much more fluent even though it inserts the spurious, but innocent word "adequate" in the translation "persons , which need adequate protection , as far as possible". On the other hand, the use of divergent phrases can sometimes yield translations with very different meanings. For example, *Annotations with Divergences* in Example 38 finds many possible translations that differ from the input sentence by only one word. A selection of these are presented in Table 8.9. The system ultimately selects a translation that generates precisely the correct structure, but substitutes "morocco" for "bulgaria" (which is obviously a problem).

Another common theme is that the additional context provided by the annotations leads to



Translation Lattice

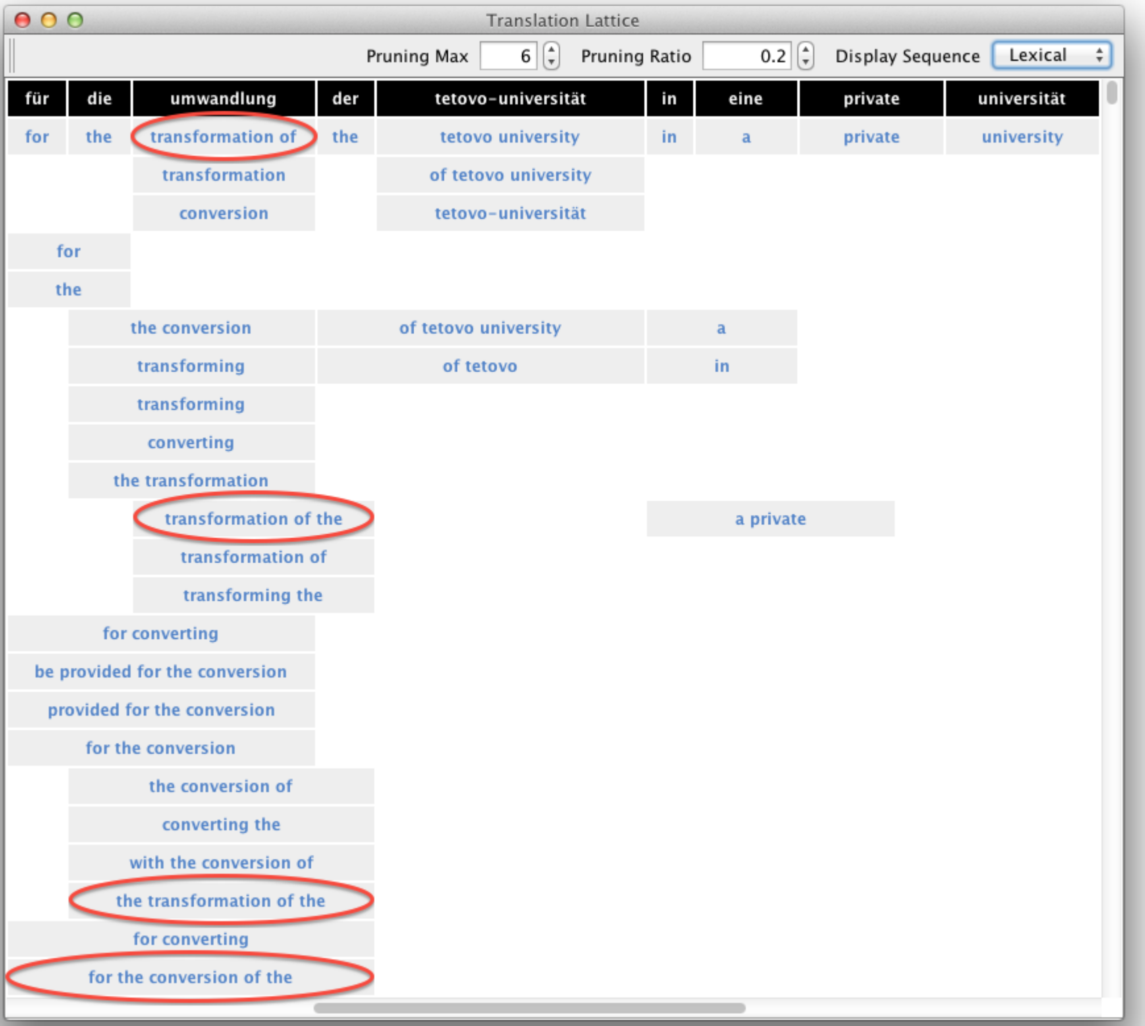
Pruning Max 6 Pruning Ratio 0.2 Display Sequence Lexical

für	die	umwandlung	der	tetovo-universität	in	eine	private	universität
for	the	transformation	the	tetovo university	in	a	private	university
to	of	conversion	of	tetovo-universität	to	an	, private	university of
of	<null>				of		of private	
					into		to private	
					<null>		private ,	
for								
to								
the								
of								
	the conversion							
	the transformation							
	transforming							
		transformation of						
		transformation of the						
			of tetovo university		a			
			of tetovo		in			
			of the university of tetovo					
			the university of tetovo					
for converting						a private		
favour of the change						private		
for converting qualifications						a private ,		
especially for converting						is a private		
	the conversion of							
	converting the							
for the conversion of the								
for the conversion of the present								
the prerequisite for the conversion of the								
prerequisite for the conversion of the								

Figure 8.6: Translation Lattice for Example 42 with Cunei Baseline

Target Phrase	**Original Rank**	**Source POS**	**Target POS**	**Source Parse**
Source Phrase: für				
for	1	-0.9935	-0.6694	0.0000
Source Phrase: für die umwandlung der				
for converting	5	-1.3679	-5.8797	-5.7683
for the conversion of the	1	0.0000	0.0000	-5.7683
the	6	0.0000	0.0000	-5.7683
for the conversion of the present	2	0.0000	0.0000	-5.7683
opportunities for converting	10	-1.3679	-10.4849	-5.7683
the prerequisite for the conversion of the	3	0.0000	0.0000	-5.7683
prerequisite for the conversion of the	4	0.0000	0.0000	-5.7683
Source Phrase: die				
the	1	-0.6716	-0.6065	0.0000
Source Phrase: die umwandlung der				
the conversion of	1	-1.3679	-7.6449	-4.3820
converting the	2	-1.9031	-6.4196	-4.3820
with the conversion of	3	-1.3679	-8.9195	-4.3820
the transformation of the	5	-1.3679	-9.4594	-4.3820
“ & the transformation of the ”	14	-1.2100	-11.7631	-4.3820
the easy transformation of	19	-1.3679	-12.2501	-4.3820
for the transformation of the	16	-1.3679	-10.7339	-4.3820
Source Phrase: umwandlung				
transformation of	5	-1.3679	-5.8305	-1.3863
transformation	1	-4.6052	-4.5559	-1.3863
conversion	2	-4.6052	-4.5559	-1.3863
transforming	3	-4.6052	-4.6052	-1.3863
converting	4	-2.0444	-3.7120	-1.3863
transmutation	15	-4.6052	-4.5559	-1.3863
convert	6	-2.7997	-3.5503	-1.3863
of transforming	34	-1.3679	-5.8797	-1.3863
Source Phrase: umwandlung der				
transformation of the	2	-1.3679	-7.6449	-2.9957
transformation of	1	-1.3679	-5.8305	-2.9957
transforming the	5	-1.9031	-6.4196	-2.9957
conversion of	3	-1.3679	-5.8305	-2.9957
Source Phrase: der				
the	1	-0.6716	-0.6065	0.0000

Table 8.7: Selected Annotation Features for Phrase Pairs in Example 42


Translation Lattice

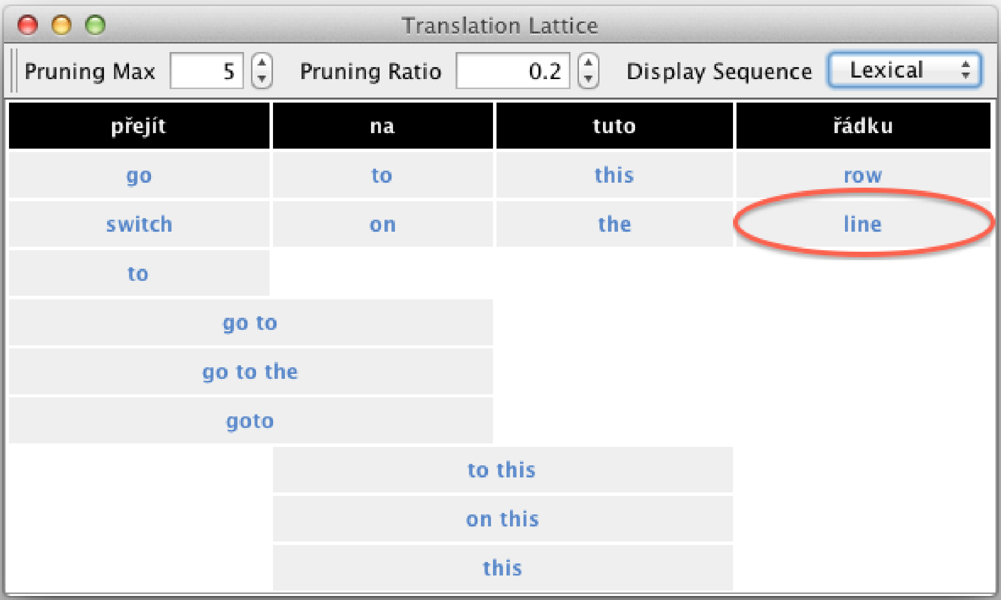
Pruning Max 6 Pruning Ratio 0.2 Display Sequence Lexical

für	die	umwandlung	der	tetovo-universität	in	eine	private	universität
for	the	transformation of	the	tetovo university	in	a	private	university
		transformation		of tetovo university				
		conversion		tetovo-universität				
for								
the								
	the conversion		of tetovo university		a			
	transforming		of tetovo		in			
	transforming							
	converting							
	the transformation							
		transformation of the			a private			
		transformation of						
		transforming the						
for converting								
be provided for the conversion								
provided for the conversion								
for the conversion								
	the conversion of							
	converting the							
	with the conversion of							
	the transformation of the							
for converting								
for the conversion of the								

Figure 8.7: Translation Lattice for Example 42 with Annotations

better word reordering. Examples 46 and 47 highlight improved word reordering across all three annotation configurations. In addition, allowing replacements specifically aids reordering in Examples 41 and 45. Gaps that are later replaced allow for longer structural matches in which individual words or phrases may be substituted. We also find that when Cunei produces the correct target words, the existing reordering models perform better. Example 47 illustrates this symbiotic relationship between improved lexical selection and reordering. Replacing "resolution applications" with "motions" is in and of itself an improvement. It is also more likely that the language model has now seen text that includes the words "withdrawn" and "motions". Indeed, the phrase "have withdrawn" is reordered to produce "have withdrawn their motions" in all configurations that score annotations.

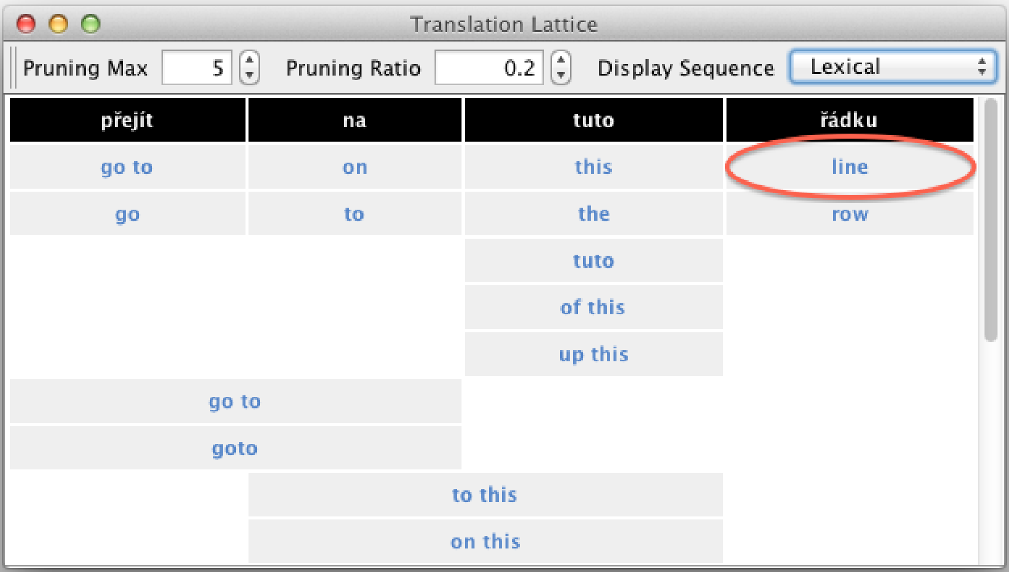
Example 37 is interesting in that the baseline system decided not to translate the Czech word "nacpěte", whereas the systems with annotations incorrectly believed they could adequately translate it. This word does exist once in the corpus (note it was translated by the system which does



Pruning Max 5 Pruning Ratio 0.2 Display Sequence Lexical

přejít	na	tuto	řádku
go	to	this	row
switch	on	the	[✓ line]
to			
go to			
go to the			
goto			
	to this		
	on this		
	this		

Figure 8.8: Translation Lattice for Example 35 with Cunei Baseline



Pruning Max 5 Pruning Ratio 0.2 Display Sequence Lexical

přejít	na	tuto	řádku
go to	on	this	[✓ line]
go	to	the	row
		tuto	
		of this	
		up this	
go to			
goto			
	to this		
	on this		

Figure 8.9: Translation Lattice for Example 35 with Annotations

Target Phrase	Original Rank	Source Cluster 100	Target Cluster 100	Source Cluster 1000	Target Cluster 1000
Source Phrase: řádku					
line	2	-0.8607	-0.5242	-2.8184	-3.0007
row	1	-1.4321	-1.2429	-2.9054	-3.9170
line of	4	-1.2960	-5.3033	-2.4651	-5.3033
the line	3	-0.8845	-5.3033	-2.8184	-5.3033
line no	14	-1.0307	-6.7413	-0.2513	-6.7707
of line	N/A	-0.9858	-5.3033	-2.5307	-4.6102
line option	9	-0.4281	-1.9409	-0.0606	-6.8279
line options	15	-0.4281	-1.9409	-0.5328	-5.8325
a row	23	-1.8693	-5.3033	-3.0007	-4.6102
each line	30	-0.9089	-5.3033	-0.5108	-6.9530
at line	26	-1.1137	-5.3033	-0.1866	-5.8110
’ line	86	-0.6981	-5.3033	-0.8473	-7.7152
command line	25	-0.4281	-1.9409	-0.4939	-4.3881
bar	8	-0.8607	-0.5242	-2.9054	-3.5115

Table 8.8: Selected Annotation Features for Phrase Pairs in Example 35

not permit lexical divergences), but the word alignments are poor (the target text in the corpus does not appear to be the correct translation). The additional annotations improved the score of the translation instance just enough that the translation “let me kill” was more likely than passing through the word untranslated. In this case, both the more permissive online phrase alignment and the use of annotations led the system down the wrong path.

8.5.5 Runtime

We perform the same timing experiments as in Section 5.1.5 of Chapter 3 to analyze the effect scoring new annotation sequences has on system runtime. The results are presented in Table 8.10. *Annotations without Lexical Divergence* and *Annotations with any Divergence* perform similarly and the decrease in speed compared to the baseline configuration is due to searching the training data over multiple sequences and retrieving all annotations from the training data when scoring a translation instance. The further reduction in *Annotations with Divergence & Replacement* is a result of identifying gaps in translation instances and allowing replacement of these gaps during decoding.

8.6 Summary

In this chapter we experimented with three different methods for integrating annotations. Enabling lexical divergences and gap replacement was consistently the best, but all uses of annotations showed improvement over the baseline. In fact, the 2.11 BLEU gain we achieved on the test set in Czech-English is better than the gain from source context. The ease with which we can annotate the corpus with new information and integrate it into the model is central to this dissertation and we are encouraged by these results.

SMT has attempted to move beyond purely lexical translation by incorporating factors or

Input Phrase: - článek 4 dohody bulharsko - španělsko

Source Phrase	**Target Phrase**
- článek 4 dohody maroko - nizozemsko	- article 4 of the morocco - netherlands agreement ;
- článek 4 dohody albánie - nizozemsko	“ - article 4 & of the albania - netherlands agreement ; ”
- článek 4 dohody chorvatsko - lucembursko	- article 4 of the croatia - luxembourg agreement ;
- článek 4 dohody rumunsko - lucembursko	- article 4 of the romania - luxembourg agreement
- článek 6 dohody uruguay - španělsko	- article 6 of the uruguay - spain agreement
- článek 3 dohody bulharsko - lucembursko	- article 3 of the bulgaria - luxembourg agreement ;
- článek 5 dohody bulharsko - lucembursko	- article 5 of the bulgaria - luxembourg agreement ;
- článek 2 dohody bulharsko - nizozemsko	- article 2 of the bulgaria - netherlands agreement ;
- článek 4 dohody chile - španělsko	- article 4 of the chile-spain
- článek 5 dohody ázerbájdžán - nizozemsko	- article 5 of the azerbaijan - netherlands agreement ;
- článek 9 dohody chorvatsko - nizozemsko	- article 9 of the croatia - netherlands agreement ;
- článek 5 dohody ukrajina - nizozemsko	- article 5 of the ukraine - netherlands agreement ;
- článek 6 dohody albánie - nizozemsko	“ - article 6 & of the albania - netherlands agreement ; ”
- článek 9 dohody ukrajina - nizozemsko	- article 9 of the ukraine - netherlands agreement ;
- článek 3 dohody maroko - lucembursko	- article 3 of the morocco - luxembourg agreement ;

Table 8.9: Structurally Similar Phrase Pairs in Example 38

	Czech-English		**German-English**	
	Runtime in Minutes	Words per Minute	Runtime in Minutes	Words per Minute
Cunei Baseline	36.95	500	106.13	252
+ Annotations without Lexical Divergence	266.76	69	395.33	68
+ Annotations with any Divergence	273.83	67	374.77	71
+ Annotations with Divergence & Replacement	257.23	72	575.54	46

Table 8.10: Runtime on Test Sets of Incorporating Annotations

synchronous context-free grammars with non-terminal labels. While some improvement has been reported, both of these approaches eventually suffer from fragmenting translation units. The structure of the model may vary, but the basic idea is that each translation unit is conditioned on additional information, such as factors or a parse tree. The conditional information lowers the bias of the model, but it also tends to increase the variance because many of the translation units now occur very rarely. Smoothing techniques are intended to address this issue, but they also make modeling more complicated.

When we add annotations to the model, we add new similarity features instead of creating specialized translation units. In this chapter we were able to re-use many of the existing per-instance features and re-compute them over the annotation labels. We also introduced new similarity features, such as those in Tables 8.2. In the same way that Chapter 6 preferred translation instances with the same source context as the input sentence and Chapter 7 preferred translation instances with the same target context as the target hypothesis, we are now able to prefer translation instances with annotations similar to both the source and target. We extend on these previous contextual models by allowing sequential and hierarchical annotations that can describe a variety of new information. Instead of fragmenting the phrase pairs based on each type of annotation, we score a phrase pair with all translation instances while identifying those with similar annotations as more relevant.

Chapter 9

Conclusion

9.1 Summary

As discussed in Chapter 2, researchers have over the years introduced several different ways to augment and extend the traditional SMT model. The instance-based approach described in this dissertation is novel in that it is able to build upon and integrate work in context, paraphrasing, adaptation, and the like into one unified model. As described formally in Chapter 3, our model scores the relevance of each translation instance with a distance function that allows for the simple integration of instance-specific features. These per-instance features enable new sources of information and unlock new potential in machine translation. We demonstrated how to effectively learn weights for the features of the distance function in Chapter 4 and then compared our model to a traditional SMT model in Chapter 5. We then extended the instance-based modeling in three key areas: source similarity (Chapter 6), target similarity (Chapter 7), and annotation similarity (Chapter 8). Each of these techniques explicitly takes advantage of the fact that Cunei scores each instance of translation; similar extensions would be impossible or quite difficult with a traditional SMT model.

The advantage of this approach is not merely conceptual; in our experiments it consistently outperformed the traditional SMT model. Throughout this dissertation we maintained common training and test sets in Czech-English and German-English on which to evaluate our approach. Starting with a baseline Cunei configuration that only added per-instance phrase alignment features, we outperformed a comparable SMT model. Each successive set of per-instance features we added to our model showed even further gains. On held-out test sets Moses scored 27.09 BLEU in Czech-English and 25.34 BLEU in German-English. In comparison, Cunei scored 32.87 BLEU in Czech-English with per-instance annotation similarity and 26.86 BLEU in German-English with per-instance source-side context. The larger relative improvement in Czech-English is due to our approach exploiting similarity to better model rare events and exploiting context to better adapt to the many different genres present in the CzEng 0.9 corpus. Even on the German-English corpus, which is larger and mostly consistent in style, our approach still yielded a gain of 1.52 BLEU.

9.2 Future Directions

We are excited by the improvements to translation we have already obtained as a result of the work in this dissertation, but we recognize there is much more within the realm of translation with instance-based modeling yet to be explored.

In evaluating the role of source context, target context, and annotation similarity, we built upon the same baseline configuration that included per-instance alignment features. This permitted us to isolate differences between the systems and better understand the effect of each new category of features. The first step in continuing this work would be to combine *all* the features into one grandiose system. When we combined all the source context features together we found small, but incremental gains. We anticipate a similar effect from combining the features presented in Chapters 6, 7, and 8.

Another obvious extension of this work is that the type and quantity of corpus annotations could be significantly expanded. In addition to the current use of part-of-speech labels and parse trees, we expect it would be beneficial to model coreference resolution and dependency trees. When applying statistical parsers, we could also move beyond 1-best analyses and include parse forests. Currently the annotations are not associated with a probability, but this would be a relatively easy extension. Also, the statistical word clusters we built were all derived from maximizing the likelihood of the data under a bigram model. Different word clustering algorithms should be explored for labeling the corpus with additional sequential annotations. A topic model, for example, would likely assign very different labels and provide novel information.

While Cunei permits translations with gaps, the mechanism for identifying gaps is quite rudimentary and permissive. We hypothesize a gap when there exists a translation instance with a token that diverges from the input sentence. This is likely overgenerating possible gap locations. Because we allow any substitution into the gap (albeit scored for similarity), the large number of gaps come at a considerable expense. We would like to investigate a mechanism for reducing the number of gaps, perhaps based on a threshold of the similarity score for the gap. This in turn may help ensure that the output is well structured by throwing out erroneous translation templates in advance.

In the previous chapters, we introduced several features for scoring context and annotations. Many of these features were based on simple similarity metrics: precision, recall, cosine distance, etc. While it is not clear that more advanced similarity metrics would make much of a difference, it would be helpful to verify whether this is the case experimentally. For simplicity the current similarity features are computed independently over each type (e.g. lexical tokens, part-of-speech labels, and parse tree annotations). There is no technical limitation preventing similarly features from operating jointly over all types, which may better capture particular phenomena. We also currently avoid a similarity score of zero by adding one to the numerator and denominator; this is likely not the best choice and alternatives should be explored.

An open question is whether some of these instance-based features can be calculated prior to translating new input. While we have attempted to make the system efficient, Cunei calculates all of the feature functions during translation. This actually works quite well for a research system where the typical cycle is repeatedly train and test. It is also the natural method for performing instance-based learning. However, this is not ideal in a production environment. It is possible that some features could be pre-computed or cached for frequently-used translation instances. Fundamentally, this is an implementation issue, but it merits investigation for better understanding the computational trade-offs of our approach.

9.3 Conclusion

Communication around the globe began with the telegraph, but it has taken on a whole new dimension with computers and the internet. News, chat rooms, tutorials, and blogs are mere keystrokes away. While the availability of information has exploded, language barriers remain. The continued hope of machine translation is that one day it will provide universal access to information from around the globe. State-of-the art systems have moved us one step closer to this goal, but the traditional SMT model does not adapt well to new domains. We have presented an instance-based model for machine translation that extends the SMT model. Instead of requiring specialized phrase pairs, our model inherently provides generalization capabilities and leverages information from all instances of translation. The model provides a simple and flexible mechanism for integrating new sources of information by scoring instances of translation with a feature-based distance function. The distance function allows the model to easily discriminate between between instances of translation and select those which are most relevant for the particular input sentence and target hypothesis. We demonstrated improvements to machine translation quality in Czech-English and German-English experiments by scoring the alignment, context, and annotations of translation instances. Machine translation–including Cunei–is far from perfect and many challenges remain. However, this dissertation contributes a modeling approach that makes machine translation more flexible and adaptable to varying data conditions.

Appendix A

Examples of Translation

A.1 Baseline Evaluation

Example 1 CzEng v0.9 Test Sentence #124

Moses Baseline	i do n't think *it 's* **maybe this one** tone **shut up** .
Moses without Lexical Reordering	i do n't think *it 's* **maybe this one** tone **shut up** .
Cunei with Moses Phrase Table	i do n't think *it 's* **maybe this one** tone **shut up** .
Cunei Baseline	i do n't think *that* **there is a need** *this* tone **be quiet .**
Reference	i do n't think we need that tone oh , be quiet .

Example 2 CzEng v0.9 Test Sentence #384

Moses Baseline	the settings should match those local area network (lan) **give you a manager** or **supplier internet services .**
Moses without Lexical Reordering	... match those **to give you a manager** local area network (lan) or **supplier internet services .**
Cunei with Moses Phrase Table	... match those **to give you a manager** local area network (lan) or **internet service provider .**
Cunei Baseline	... match those **provided by your** local area network (lan) **administrator** or **internet service provider .**
Reference	the settings should match those provided by your local area network (lan) administrator or internet service provider (isp) .

Example 3 CzEng v0.9 Test Sentence #853

Moses Baseline	**to tackle the problem of** that certain tax payers is tolerated non-payment of taxes and social security contributions .
Moses without Lexical Reordering	**to tackle the problem of** that certain tax payers is tolerated non-payment of taxes and social security contributions .
Cunei with Moses Phrase Table	**to tackle the problem of** that certain tax payers is *no tolerance for the* non-payment of taxes and social security contributions .
Cunei Baseline	**address the phenomenon** that non-payment of taxes and social security contributions is tolerated *for* certain tax payers .
Reference	address the phenomenon that non-payment of taxes and social security contributions is tolerated for certain tax payers .

Example 4 CzEng v0.9 Test Sentence #1091

Moses Baseline	czech airlines 1 march will celebrate *the* sixty years *since the opening of* the regular **flight connection** between prague and *the* german capital berlin .
Moses without Lexical Reordering	... *the* sixty years *since the start of* the regular **flight connection** between prague and *the* german capital berlin .
Cunei with Moses Phrase Table	... sixty years *after the start of* the regular **flight connection** between prague and german capital berlin .
Cunei Baseline	... *the* sixty years *since the start of* the regular **air service** between prague and *the* german capital *of* berlin .
Reference	on 1 march , czech airlines will celebrate sixty years since the launch of scheduled air service between prague and germany ’s capital , berlin .

Example 5 CzEng v0.9 Test Sentence #386

Moses Baseline	if you *want the* windows to try and **setting discover** them , click detect network settings .
Moses without Lexical Reordering	if you *want the* windows to try and **settings discover** them , click detect network settings .
Cunei with Moses Phrase Table	if you *would like* windows to try and **settings discover** them , click detect network settings .
Cunei Baseline	if you *would like* windows to try and **discover** them , click detect network settings
Reference	if you would like windows to try and discover them , click detect network settings

Example 6 German Europarl v6 Test Sentence #583

Moses Baseline	the article does not **at** community aid , but *on* the compatibility of *the* national aid with the laws of the community .
Moses without Lexical Reordering	the article does not **deal with the** community *in* aid , but the compatibility of *the* national aid with the laws ...
Cunei with Moses Phrase Table	the article does not **deal with the** community aid , but the compatibility of *the* national aid with the laws ...
Cunei Baseline	the article does not **relate to** community aid , but *on* the compatibility of national aid with the laws ...
Reference	the article does not relate to community aid , but relates to the compatibility of national aid with community legislation .

Example 7 German Europarl v6 Test Sentence #817

Moses Baseline	the dozens of **dead** of the last few days *i* fear the worst .
Moses without Lexical Reordering	the dozens of **dead** of the last few days *let* fear ...
Cunei with Moses Phrase Table	the dozens of **dead** of the last few days *let* fear ...
Cunei Baseline	the dozens of **deaths** of the last few days *i* fear ...
Reference	the dozens of deaths over recent days make me fear the worse .

Example 8 German Europarl v6 Test Sentence #861

Moses Baseline	the democratic process in côte d'ivoire is now very **got off to a good start .**
Moses without Lexical Reordering	... côte d'ivoire is now very **got off to a good start .**
Cunei with Moses Phrase Table	... côte d'ivoire is now very **got off to a good start .**
Cunei Baseline	... côte d'ivoire is now very **well .**
Reference	the democratic process in côte d'ivoire is well under way .

Example 9 German Europarl v6 Test Sentence #469

Moses Baseline	`the next item is the debate on the following` **`resolutions`** `:`
Moses without Lexical Reordering	`... debate on the following` **`resolution motions`** `:`
Cunei with Moses Phrase Table	`... debate on the following` **`resolutions`** `:`
Cunei Baseline	`... debate on the following` **`motions for resolutions`** `:`
Reference	`the next item is the debate on the following motions for resolution :`

Example 10 German Europarl v6 Test Sentence #764

Moses Baseline	*`i am pleased but also`* `that` **`we are managed`** `,` *`the`* `germ gene therapy put a stop .`
Moses without Lexical Reordering	*`however , i am , too ,`* `that` **`it has managed to us here`** `germ gene therapy put a stop .`
Cunei with Moses Phrase Table	*`i am glad , though ,`* `that` **`we are managed`** `, germ gene therapy put a stop .`
Cunei Baseline	*`i am glad , though ,`* `that` **`we have succeeded`** `, germ gene therapy put a stop` *`to`* `.`
Reference	`i am also pleased , however , that we have succeeded in clamping down on germ-line gene therapy .`

A.2 Incorporating Source Similarity

Example 11 CzEng v0.9 Test Sentence #1096

Cunei Baseline	very significant part of passengers from berlin but is continuing other connect czech airlines flights to the middle east or to south or *southeast* europe .
+ Static Annotations	... flights to the middle east or to south or **south-eastern** europe .
+ Dynamic Annotations	... flights to the middle east or to south or **south-eastern** europe .
+ Sentence Context	... flights towards the *near and* middle east or to south or **southeastern** europe .
+ Document Context	... flights to the middle east or to south or **south-eastern** europe .
All Context Features	... flights to the middle east or to the south or **south-eastern** europe .
Reference	a significant number of the passengers from berlin carry on , beyond prague , on connecting czech airlines flights to the middle east , or to southern or south-eastern europe .

Example 12 CzEng v0.9 Test Sentence #450

Cunei Baseline	the % 1 service announced invalid **the status quo** % 2 .
+ Static Annotations	... announced invalid **the current state** % 2 .
+ Dynamic Annotations	... announced invalid **the current state** % 2 .
+ Sentence Context	... announced invalid **the status quo** % 2 .
+ Document Context	... announced invalid **state of play** % 2 .
All Context Features	... announced invalid **the current state** % 2 .
Reference	the % 1 service has reported an invalid current state % 2 .

Example 13 CzEng v0.9 Test Sentence #491

Cunei Baseline	`call to % 1 over the failed with this` **`mistake`** `: % n % 2`
+ Static Annotations	`call ... failed with this` **`error`** `: % n % 2`
+ Dynamic Annotations	`call ... failed with this` **`mistake`** `: % n % 2`
+ Sentence Context	`call ... failed with this` **`mistake`** `: % n % 2`
+ Document Context	`call ... failed with this` **`error`** `: % n % 2`
All Context Features	`call ... failed with this` **`error`** `: % n % 2`
Reference	`the % 1 call failed with the following error : % n % 2`

Example 14 CzEng v0.9 Test Sentence #645

Cunei Baseline	`in the upper right corner was stamp` **`secret`** `.`
+ Static Annotations	`in the upper right corner was stamp` **`secret`** `.`
+ Dynamic Annotations	`in the upper right corner was stamp` **`secret`** `.`
+ Sentence Context	`in the upper right corner was stamp` **`secret`** `.`
+ Document Context	`in the upper right corner was stamp` **`classified`** `.`
All Context Features	`in the upper right corner was stamp` **`secret`** `.`
Reference	`stamped on the top right was classified .`

Example 15 CzEng v0.9 Test Sentence #868

Cunei Baseline	`-` **`claims`** `which , at the beginning of the calendar year , have not yet been recovered ,`
+ Static Annotations	`-` **`claims`** `which ... have not yet been recovered ,`
+ Dynamic Annotations	`-` **`claims`** `which ... have not yet been recovered ,`
+ Sentence Context	`-` **`debts`** `which ... have not yet been recovered ,`
+ Document Context	`-` **`claims`** `which ... have not yet been recovered ,`
All Context Features	`-` **`debts`** `which ... have not yet been recovered ,`
Reference	`- debts still to be recovered at the beginning of the calendar year ,`

Example 16 CzEng v0.9 Test Sentence #1348

Cunei Baseline	because the french use **the large** roman numerals , when refer to the
+ Static Annotations	because the french use **the large** roman numerals ...
+ Dynamic Annotations	because the french use **the large** roman numerals ...
+ Sentence Context	because the french use **the large** roman numerals ...
+ Document Context	because the french use **the large** roman numerals ...
All Context Features	because the french use **capital** roman numerals ...
Reference	since the french use capital roman numerals to refer to the

Example 17 German Europarl v6 Test Sentence #527

Cunei Baseline	i do not know exactly what the situation in other parts of europe , in south-east england in any event , that is a real and **current** threat .
+ Static Annotations	... that is a real and **current** threat .
+ Dynamic Annotations	... that is a real and **current** threat .
+ Sentence Context	... that is a real and **present** threat .
+ Document Context	... that is a real and **current** threat .
+ All Context Features	... that is a real and **present** threat .
Reference	i do not know exactly the situation across europe but in the south-east of england this is a real and present danger .

Example 18 German Europarl v6 Test Sentence #603

Cunei Baseline	also , the european commission will in the resolution put to the vote , called for the introduction of a special *with a* green paper on **the issue** .
+ Static Annotations	... of a special green paper on **this question** .
+ Dynamic Annotations	... of a special green paper on **this question** .
+ Sentence Context	... of a special *with a* green paper on **the issue** .
+ Document Context	... of a special *with a* green paper on **the issue** .
+ All Context Features	... of a special *with a* green paper on **the issue** .
Reference	moreover , the text we will be voting on calls on the european commission to draw up a green paper specifically on this question .

Example 19 German Europarl v6 Test Sentence #320

Cunei Baseline	with particular satisfaction , i welcome the new higher education law that creates the prerequisites for that the **albanian language population** group a private university reasons *can* .
+ Static Annotations	... prerequisites for that the **albanian language population** group a private university reasons *can* .
+ Dynamic Annotations	... prerequisites for that the **albanian-speaking population** group a private university reasons *can* .
+ Sentence Context	... prerequisites for that the **albanian-speaking population** group a private university reasons .
+ Document Context	... prerequisites for that the **albanian language population** group a private university reasons *can* .
+ All Context Features	... prerequisites for that the **albanian-speaking citizens** group a private university reasons .
Reference	i welcome with the utmost satisfaction the new law on higher education , which will create the conditions needed for the albanian-speaking community to set up a private university .

Example 20 German Europarl v6 Test Sentence #689

Cunei Baseline	that was the aim of the european parliament in the legislative process on clinical **review** , and i **think** that *today we can say this : this* **objective** has been achieved .
+ Static Annotations	... on clinical **review** , and i **think** that *today we can say this : this* **objective** has been achieved .
+ Dynamic Annotations	... on clinical **trials** , and i **believe** that *we can now say : this* **aim** has been achieved .
+ Sentence Context	... on clinical **review** , and i **think** that *today we can say this : this* **objective** has been achieved .
+ Document Context	... on clinical **trials** , and i **think** that *today we can say this : this* **objective** has been achieved .
+ All Context Features	... on clinical **trials** , and i **believe** that *we can now say : that* **objective** has been achieved .
Reference	this was the european parliament 's objective in the legislative procedure on clinical trials , and i believe that today we can say that this objective has been achieved .

Example 21 German Europarl v6 Test Sentence #655

Cunei Baseline	what we have here is again with an own-initiative report by the committee on employment and social affairs , that , despite the fact that the problem of **additives** health insurance is a burning in *the* committee and also , here in the european parliament , is not uncontroversial , *because they always continue to have* him 12 members on the committee not agreed .
+ Static Annotations	... the problem of **supplementary** health insurance is a burning in *the* committee and also , here in the european parliament , is not uncontroversial , *because they always continue to have* him 12 members ...
+ Dynamic Annotations	... the problem of **additives** health insurance is a burning in *the* committee and also here in the european parliament , is not uncontroversial , *because they always continue to have* him 12 members ...
+ Sentence Context	... the problem of **supplementary** health insurance is a burning in *the* committee and also , here in the european parliament , is not uncontroversial , *as this is* him 12 members ...
+ Document Context	... the problem of **additives** health insurance is a burning in committee and also , here in the european parliament , is not uncontroversial , *because they always continue to have* him 12 members ...
+ All Context Features	... the problem of **additives** health insurance is a burning in *the* committee and also , here in the european parliament , is not uncontroversial , *as this is* him 12 members ...
Reference	(de) we are dealing again with an own-initiative report by the committee on employment and social affairs , which , although supplementary health insurance is a highly topical issue , is not totally endorsed by the committee or here in the plenary of the european parliament since , after all , 12 members did not vote for it in committee .

A.3 Incorporating Target Similarity

Example 22 CzEng v0.9 Test Sentence #188

Cunei Baseline	`**as our** success depends on **the of** surprise and **use the** entire fleet **insurgents .**`
+ Target Context	**still ,** our success depends on **the element of** surprise and **use of the** entire fleet **rebel offensive .**
Reference	still , our success depends on the element of surprise and the use of the entire rebel fleet .

Example 23 CzEng v0.9 Test Sentence #374

Cunei Baseline	*the removal of all the* temporary files
+ Target Context	*removes any* temporary files **used**
Reference	removes any temporary files used

Example 24 CzEng v0.9 Test Sentence #484

Cunei Baseline	dns domain this computer was **amended** from % 1 to % 2 .
+ Target Context	the dns domain of this computer was **changed** to % 1 on % 2 .
Reference	the dns domain assigned to this computer has been changed from % 1 to % 2 .

Example 25 CzEng v0.9 Test Sentence #760

Cunei Baseline	`sadi looked quizzically at garion ,` **`in his hands was ready for`** `his thin and a small knife .`
+ Target Context	`sadi looked quizzically at garion ,` **`holding ready`** `his thin and a small knife .`
Reference	`sadi looked inquiringly at garion , holding up his slim little knife suggestively .`

Example 26 CzEng v0.9 Test Sentence #1335

Cunei Baseline	`more specifically , it ’s the average distance` **`values data`** `from their average .`
+ Target Context	`more specifically , it ’s the average distance` **`data values`** `from their average .`
Reference	`more precisely , it is a measure of the average distance of the data values from their mean .`

Example 27 CzEng v0.9 Test Sentence #1348

Cunei Baseline	`because the french use` **`the large`** `roman numerals , when refer to the`
+ Target Context	`because the french use` **`capital`** `roman numerals , when refer to the`
Reference	`since the french use capital roman numerals to refer to the`

Example 28 German Europarl v6 Test Sentence #5

Cunei Baseline	`for some unknown reason ,` **`appears`** `my name is not included in the list of those present .`
+ Target Context	`for some unknown reason , my name is not included in the list of those present .`
Reference	`for some strange reason , my name is missing from the register of attendance .`

Example 29 German Europarl v6 Test Sentence #192

Cunei Baseline	`let us hope that` **`we in future`** `, at least these guarantees can achieve .`
+ Target Context	`let us hope that` **`in the future we`** `at least , these guarantees can achieve .`
Reference	`let us hope that in the future we will at least be able to achieve those guarantees .`

Example 30 German Europarl v6 Test Sentence #457

Cunei Baseline	the **public access** to information **has been** improved .
+ Target Context	the **access of the public** to *the* information **has** improved .
Reference	public access to information has improved .

Example 31 German Europarl v6 Test Sentence #562

Cunei Baseline	consequently , we must this industry areas the greatest efforts to reduce emissions **companies** .
+ Target Context	consequently , we must this industry areas the greatest efforts to reduce emissions .
Reference	it is therefore these industries which ought to be making a major effort to reduce emissions .

Example 32 German Europarl v6 Test Sentence #685

Cunei Baseline	the explanations of vote **are closed** .
+ Target Context	**that concludes** the explanations of vote .
Reference	that concludes the explanations of vote .

Example 33 German Europarl v6 Test Sentence #903

Cunei Baseline	according to my information **have** but all other groups your **resolution applications withdrawn** .
+ Target Context	according to my information , but all other groups their **withdrawn the motions for resolutions** .
Reference	but , according to my information , all the other groups have withdrawn their motions .

A.4 Incorporating Corpus Annotations

Example 34 CzEng v0.9 Test Sentence #217

Cunei Baseline	`do you think the hong kong is a dangerous place , to` **`life`** `?`
+ Annotations without Lexical Divergence	`... is a dangerous place to` **`live`** `?`
+ Annotations with any Divergence	`... is a dangerous place to` **`live`** `?`
+ Annotations with Divergence & Replacement	`... is a dangerous place to` **`live`** `?`
Reference	`do you think hong kong is a dangerous place to live ?`

Example 35 CzEng v0.9 Test Sentence #226

Cunei Baseline	`go to this` **`row`**
+ Annotations without Lexical Divergence	`go to this` **`line`**
+ Annotations with any Divergence	`go to this` **`line`**
+ Annotations with Divergence & Replacement	`go to this` **`line`**
Reference	`navigate to this line`

Example 36 CzEng v0.9 Test Sentence #518

Cunei Baseline	`there will be necessary to strengthen ,` *`particularl`*`y in` **`the context of`** `the` **`eu`** `.`
+ Annotations without Lexical Divergence	`... to strengthen` *`in particular`* `in` **`connection with`** `the` **`advent of eu enlargement`** `.`
+ Annotations with any Divergence	`... to strengthen` *`in particular`* `in` **`connection with`** `the` **`advent of eu enlargement`** `.`
+ Annotations with Divergence & Replacement	`... to strengthen` *`in particular`* `in` **`connection with`** `the` **`enlargement of the eu`** `.`
Reference	`particularly in connection with enlargement , this confidence will need to be strengthened .`

Example 37 CzEng v0.9 Test Sentence #566

Cunei Baseline	`if you` **`got`** `pipe with me , sit down and` **`nacpěte from`** `me !`
+ Annotations without Lexical Divergence	`if you` **`have a`** `pipe with me , sit down and` **`let me kill`** `me !`
+ Annotations with any Divergence	`if you` **`have a`** `pipe with me , sit down and` **`let me kill`** `me !`
+ Annotations with Divergence & Replacement	`if you` **`have a`** `pipe with me , sit down and` **`let me kill`** `me !`
Reference	`if you have a pipe about you , sit down and have a fill of mine !`

Example 38 CzEng v0.9 Test Sentence #719

Cunei Baseline	`- article 4 of the` **`agreement bulgaria - spain`**
+ Annotations without Lexical Divergence	`- article 4 of the` **`bulgaria - spain`**
+ Annotations with any Divergence	`- article 4 of the` **`morocco - spain agreement ;`**
+ Annotations with Divergence & Replacement	`- article 4 of the` **`bulgaria - spain`**
Reference	`- article 4 of the bulgaria - spain agreement ;`

Example 39 CzEng v0.9 Test Sentence #1015

Cunei Baseline	**`as`** `people` **`are`** *`of a particular`* `goods and` *`what`* `the product` **`marks`** `know ?`
+ Annotations without Lexical Divergence	**`how`** `people` **`are`** *`specific`* `goods and the product` **`brands`** *`they`* `know ?`
+ Annotations with any Divergence	`people` **`are`** *`specific`* `goods and the product` **`brands`** *`they`* `know ?`
+ Annotations with Divergence & Replacement	**`how do`** `people` **`choose`** *`a specific`* `goods and the product` **`brands`** `know ?`
Reference	`how do they choose particular goods in the store and which product brands they know ?`

Example 40 CzEng v0.9 Test Sentence #1345

Cunei Baseline	`film` **`evaluation`** `system in hong kong uses category i , iia , iib and iii , based on roman číslicích .`
+ Annotations without Lexical Divergence	*`the`* `film` **`rating`** `system in hong kong uses ...`
+ Annotations with any Divergence	*`the`* `film` **`rating`** `system in hong kong uses ...`
+ Annotations with Divergence & Replacement	`film` **`rating`** `system in hong kong uses ...`
Reference	`the motion picture rating system in hong kong uses categories i , iia , iib , and iii based on roman numerals .`

Example 41 German Europarl v6 Test Sentence #255

Cunei Baseline	`we all hope , of course , including the greek colleagues here that this dispute soon ,` **`will now be resolved`** `.`
+ Annotations without Lexical Divergence	`... that this dispute soon` **`to be resolved`** `.`
+ Annotations with any Divergence	`... that this dispute soon .`
+ Annotations with Divergence & Replacement	`... that this dispute` **`will be settled`** `soon .`
Reference	`of course we all hope - and that includes the greek meps here - that this dispute will soon be settled .`

Example 42 German Europarl v6 Test Sentence #272

Cunei Baseline	`we now expect the necessary` *`aid`* `for the` **`conversion`** `of the tetovo university in a private university , whose diplomas then` *`also needs to be`* `recognised .`
+ Annotations without Lexical Divergence	`we now expect the necessary` *`assistance`* `for the` **`transformation`** `of the tetovo university ... whose diplomas then` *`are also`* `recognised .`
+ Annotations with any Divergence	`we now expect the necessary` *`assistance`* `for the` **`transformation`** `of the tetovo university ... whose diplomas then` *`are also`* `recognised .`
+ Annotations with Divergence & Replacement	`we now expect the necessary` *`assistance`* `for the` **`transformation`** `of tetovo university ... whose diplomas then` *`are also`* `recognised .`
Reference	`we are now awaiting the necessary aid for the transformation of tetovo university into a private university whose diplomas will then be recognised .`

Example 43 German Europarl v6 Test Sentence #363

Cunei Baseline	`ultimately was after some tough negotiations , a` **`final outcome reached defended deserves .`**
+ Annotations without Lexical Divergence	`ultimately , after some tough negotiations , a` **`final outcome , which deserves to be defended .`**
+ Annotations with any Divergence	`ultimately , after some tough negotiations , a` **`result which deserves to be defended .`**
+ Annotations with Divergence & Replacement	`ultimately , after some tough negotiations , a` **`result that deserves to be defended .`**
Reference	`ultimately , after some tough negotiating , an outcome was achieved that is worth defending .`

Example 44 German Europarl v6 Test Sentence #752

Cunei Baseline	children and the disabled are vulnerable **people** , which **is one of the widest possible protection need .**
+ Annotations without Lexical Divergence	children and the disabled are vulnerable **persons , the possible far-reaching need of protection .**
+ Annotations with any Divergence	children and the disabled are vulnerable **persons** , which **need adequate protection , as far as possible .**
+ Annotations with Divergence & Replacement	children and disabled **people** are vulnerable **persons** , which **need adequate protection , as far as possible .**
Reference	children and disabled people are fragile people who need maximum protection .

Example 45 German Europarl v6 Test Sentence #865

Cunei Baseline	**it seems to me ,** the concept of the ivorität **completely justified** to be **.**
+ Annotations without Lexical Divergence	**it seems to me ,** the concept of the ivorität **perfectly right** to be .
+ Annotations with any Divergence	**it seems to me** that the concept of ivorität **perfectly right** to be .
+ Annotations with Divergence & Replacement	the concept of ivorität **, it seems to me** to be **fully justified .**
Reference	the concept of ivorian nationality would appear to me to be perfectly well founded .

Example 46 German Europarl v6 Test Sentence #873

Cunei Baseline	the commission is also **on the recent violence very concerned .**
+ Annotations without Lexical Divergence	the commission is also **very concerned about the recent violence .**
+ Annotations with any Divergence	the commission is also **very concerned on the recent violence .**
+ Annotations with Divergence & Replacement	the commission is also **very concerned on the recent violence .**
Reference	the commission is also very concerned by the recent violence .

Example 47 German Europarl v6 Test Sentence #905

Cunei Baseline	`please allow me to confirm whether the other groups` **`your resolution applications was indeed have withdrawn .`**
+ Annotations without Lexical Divergence	`please allow me to confirm whether the other groups` **`have withdrawn their motions indeed .`**
+ Annotations with any Divergence	`please allow me to confirm whether the other groups` **`have withdrawn their motions indeed .`**
+ Annotations with Divergence & Replacement	`please allow me to confirm whether the other groups` **`have withdrawn their motions indeed .`**
Reference	`please be so kind as to confirm that the other groups have indeed withdrawn their motions .`

Appendix B

Bibliography

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, Ann Arbor, MI.

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, MI.

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.

Bojar, O. and Žabokrtský, Z. (2009). CzEng 0.9: Large parallel treebank with rich annotation. *Prague Bulletin of Mathematical Linguistics*, 92.

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, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

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. (2000). Automated generalization of translation examples. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 125–131, Saarbrücken, Germany.

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. Springer.

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, MI.

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, NY.

Carbonell, J., Klein, S., Miller, D., Steinbaum, M., Grassiany, T., and Frei, J. (2006). Context-based machine translation. In *Proceedings of the 7th Conference of the Association for Machine Translation of the Americas*, pages 19–28, Cambridge, MA.

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, MI.

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.

Chappelier, J.-C. and Rajman, M. (1998). A generalized CYK algorithm for parsing stochastic CFG. In *Proceedings of the First Workshop on Tabulation in Parsing and Deduction*, pages 133–137, Paris, France.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Ann Arbor, MI.

Chiang, D. (2012). Hope and fear for discriminative training of statistical translation models. *Journal of Machine Learning Research*, 13. To appear.

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, CO.

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.

Daelemans, W. and van den Bosch, A. (1992). Generalization performance of backpropagation learning on a syllabification task. In Drossaers, M. F. J. and Nijholt, A., editors, *Proceedings Twente Workshop on Language Technology 3: Connectionism and Natural Language Processing*, pages 27–38, Enschede, Netherlands. University of Twente.

Daelemans, W., Zavrel, J., Berck, P., and Gillis, S. (1996). MBT: A memory-based part of speech tagger-generator. In Ejerhed, E. and Dagan, I., editors, *Proceedings of the Fourth Workshop on Very Large Corpora*, pages 14–27, Copenhagen, Denmark.

Denkowski, M. and Lavie, A. (2010). METEOR-NEXT and the METEOR paraphrase tables: Improved evaluation support for five target languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 339–342, Uppsala, Sweden.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram cooccurrence statistics. In *Proceedings of the Human Language Technology Conference*, pages 128–132, San Diego, CA.

Fix, E., H. J. (1951). Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, TX.

Foster, G. and Kuhn, R. (2007). Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 128–135, Prague, Czech Republic.

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, OH.

Green, T. (1979). The necessity of syntax markers: Two experiments with artificial languages. *Journal of Verbal Learning and Behavior*, 18:481–496.

Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell System Technical Journal*, 29:147–160.

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.

Klein, D. and Manning, C. D. (2003a). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Klein, D. and Manning, C. D. (2003b). Fast exact inference with a factored model for natural language parsing. In S. Becker, S. T. and Obermayer, K., editors, *Advances in Neural Information Processing Systems*, volume 15, pages 3–10, Cambridge, MA. MIT Press.

Koehn, P. (2004a). Pharaoh: a beam search decoder for phrase-based statistical machine translation models. 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 115–124, Washington, DC. Springer.

Koehn, P. (2004b). Statistical significance tests for machine translation evaluation. In *2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain.

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, PA.

Koehn, P. and Hoang, H. (2007). Factored translation models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague, Czech Republic.

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.

Li, Z., Callison-Burch, C., Dyer, C., Khudanpur, S., Schwartz, L., Thornton, W., Weese, J., and Zaidan, O. (2009). Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translatio*, pages 135–139, Athens, Greece.

Liang, P., Bouchard-Côté, A., Klein, D., and Taskar, B. (2006a). An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Sydney, Australia.

Liang, P., Taskar, B., and Klein, D. (2006b). 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, NY.

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

Locke, W. N. and Booth, A. D., editors (1955). *Machine Translation of Languages*. The Technology Press of the Massachusetts Institute of Technology, Cambridge, MA.

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.

Lu, Y., Huang, J., and Liu, Q. (2007). Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 343–350, Prague, Czech Republic.

Mahalanobis, P. C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Science of India*, 2(1):49–55.

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. Society for Industrial and Applied Mathematics.

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.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Moore, R. C. and Lewis, W. (2010). Intelligent selection of language model training data. In Hajič, J., Carberry, S., Clark, S., and Nivre, J., editors, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 220–224, Uppsala, Sweden.

Nagao, M. (1984). A framework of a mechanical translation between Japanese and English by analogy principle. In Elithorn, A. and Banerji, R., editors, *Artificial and Human Intelligence*, pages 173–180. Elsevier Science Publishers.

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. (1999). An efficient method for determining bilingual word classes. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics*, pages 71–76, Bergen, Norway.

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, PA.

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, PA.

Phillips, A. and Brown, R. (2011). Training machine translation with a second-order Taylor approximation of weighted translation instances. In *Proceedings of the 13th Machine Translation Summit*, pages 40–47, Xiamen, China.

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. (2010). The Cunei Machine Translation Platform for WMT '10. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 155–160, Uppsala, Sweden.

Phillips, A. B. (2011). Cunei: Open-source machine translation with relevance-based models of each translation instance. *Machine Translation*, 25(2):161–177.

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.

Rafferty, A. N. and Manning, C. D. (2008). Parsing three German treebanks: lexicalized and unlexicalized baselines. In *Proceedings of the Workshop on Parsing German*, PaGe '08, pages 40–46, Stroudsburg, PA.

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition.

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.

Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423.

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.

Shepard, D. (1968). A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524, New York, NY.

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.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation of the Americas*, pages 223–231, Cambridge, MA.

Somers, H. (1999). Review article: Example-based machine translation. *Machine Translation*, 14:113–157.

Stolcke, A. (2002). SRILM - an extensible language modeling toolkit. In *7th International Conference on Spoken Language Processing*, pages 901–904, Denver, CO.

Veale, T. and Way, A. (1997). Gaijin: A bootstrapping, template-driven approach to example-based MT. In *Proceedings of 2nd International Conference on Recent Advances in Natural Language Processing*, pages 239–244, Tzigov Chark, Bulgaria.

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.

Zavrel, J. and Daelemans, W. (1997). Memory-based learning: Using similarity for smoothing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 436–443, Madrid, Spain.

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.

Zipf, G. K. (1932). *Selective Studies and the Principle of Relative Frequency in Language.* Harvard University Press, Cambridge, MA.