# IMPROVED WORD ALIGNMENTS FOR STATISTICAL MACHINE TRANSLATION

by

Alexander Fraser

---

Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(COMPUTER SCIENCE)

December 2007

# Table of Contents

# List Of Tables

# List Of Figures

# Abstract

All state of the art statistical machine translation systems and many example-based machine translation systems depend on an annotation of word-level translational correspondence between sets of parallel sentences. Such an annotation of two parallel sentences is called a “word alignment”. The largest number of manually annotated word alignments currently available to the research community for any pair of languages consists of alignments for only thousands of parallel sentences, even though there are several orders of magnitude more parallel sentences available. For instance, for the task of translating Chinese news articles to English, there are currently on the order of 10 million parallel sentences. This is too many for manual alignment, so they must be automatically word aligned.

Unsupervised word alignment systems generate poor quality alignments, often using statistical word alignment models developed over 10 years ago, but most recent research into improving word alignments has not led to improved translation. There are several reasons for this:

1. There is no good metric which can be used to automatically measure word alignment quality for the translation task.

2. Statistical word alignment models are based on assumptions about the structure of the problem which are incorrect.

3. It is difficult to add new sources of linguistic knowledge because many current systems must be completely reengineered for each new knowledge source.

4. Statistical models of word alignment are most often learned in an unsupervised training process which is unable to take advantage of annotated data.

This thesis remedies these problems by making contributions in the following three areas:

1. We have found a new method for automatically measuring alignment quality using an unbalanced F-Measure metric (Fraser & Marcu, 2007b). We have validated that this metric adequately measures alignment quality for the translation task. We have shown that the metric can be used to derive a loss function for discriminative training approaches, and it is useful for measuring progress during the development of new word alignment procedures.

2. We have designed a new statistical model for word alignment called LEAF (Fraser & Marcu, 2007a), which directly models the word alignment structure as it is

used for machine translation, in contrast with previous models which make unreasonable structural assumptions.

3. We have developed a semi-supervised training algorithm, the EMD algorithm (Fraser & Marcu, 2006), which automatically takes advantage of whatever quantity of manually annotated data can be obtained. The use of the EMD algorithm allows for the introduction of new knowledge sources with minimal effort.

We have shown that these contributions improve state of the art statistical machine translation systems in experiments on challenging large data sets.

# Chapter 1

# Motivation

## 1.1 The Word Alignment Problem

Word alignment is the problem of determining translational correspondence at the word level given a pair of sentences, one of which is a translation of the other. The graph in Figure 1.1 shows a word alignment of a pair of parallel sentences taken from the LDC Canadian Hansards corpus, which consists of English and French documents. In this dissertation we will consider the task of automatically annotating word alignments.

Automatically aligning word level translational correspondence in parallel sentences so as to be able to learn translation rules of high quality is a challenging problem in terms of both accuracy and tractability. Most of the currently successful approaches used in conjunction with state of the art statistical machine translation systems use

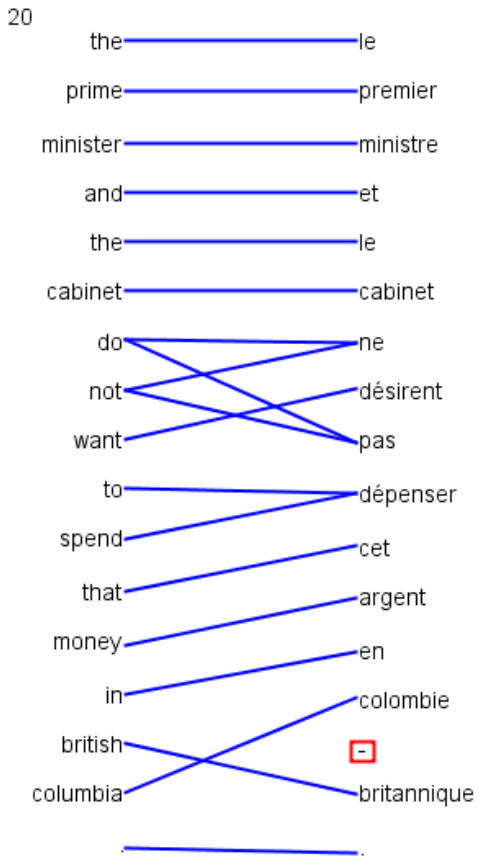


Figure 1.1: French/English gold standard word alignment

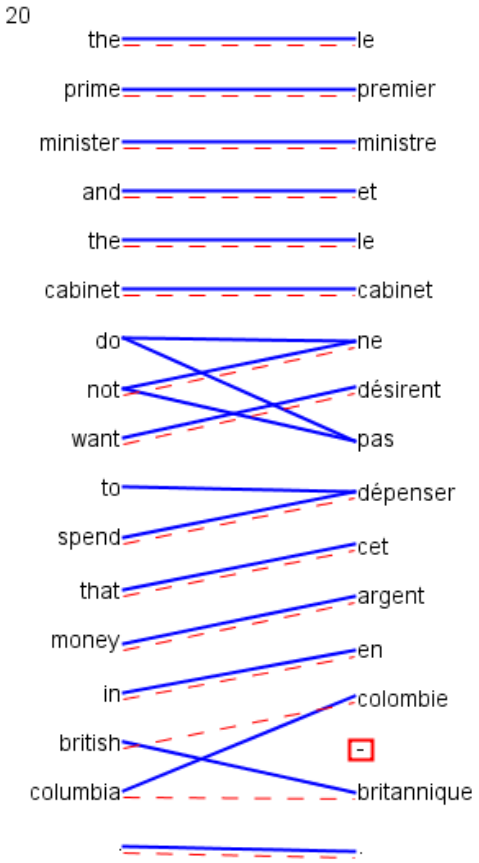


Figure 1.2: French/English gold standard word alignment (solid lines) and system hypothesis (dashed lines)

statistical models of carefully crafted generative stories which are trained using unsupervised learning methods. The task of automatic word alignment is very different from the automatic translation task. In automatic translation, we are trying to generate a reasonable translation, which does not necessarily attempt to mimic all the complexities

of human behavior. In automatic word alignment, on the other hand, we must annotate an original sentence and whatever humans chose to produce as a translation.

The research community has recently become very interested in improving the quality of automatic word alignment, as evidenced by a large number of recent papers beginning with Al-Onaizan et al. (1999), and in particular two workshops featuring shared word alignment tasks, WPT03 (Mihalcea & Pederson, 2003) and WPT05 (Martin et al., 2005). One reason for this is that word alignments are critical to building statistical machine translation (SMT) systems. For instance, the estimation of phrase-based SMT models (Koehn et al., 2003) such as those implemented in the Alignment Templates system (Och & Ney, 2004) and Moses (Koehn et al., 2007) relies on word alignments. Syntactic SMT models (Galley et al., 2004; Galley et al., 2006; Melamed, 2004; Chiang, 2005; Quirk et al., 2005; Zollmann & Venugopal, 2006) also require word alignments. Phrase-based and syntactic SMT models represent the state of the art in SMT, and therefore improving automatic word alignment is an important endeavor.

Word alignment techniques are not only used in translation, but in fact to acquire knowledge in virtually all trans-lingual tasks: Cross-Lingual Information Retrieval (Hiemstra & de Jong, 1999; Xu et al., 2001; Fraser et al., 2002), Trans-lingual Coding (sometimes referred to as annotation projection) (Yarowsky et al., 2001; Hwa et al., 2002), Document Alignment (Resnik & Smith, 2003), Sentence Alignment (Moore, 2002), Extraction of Parallel Sentences from Comparable Corpora (Munteanu et al.,

2004; Fung & Cheung, 2004), etc. Many approaches to monolingual tasks also take advantage of knowledge learned from word alignments. Some examples are summarization (Daumé III & Marcu, 2005), query expansion for monolingual information retrieval (Xu et al., 2002; Riezler et al., 2007), paraphrasing (Pang et al., 2003; Quirk et al., 2004; Bannard & Callison-Burch, 2005), grammar induction (Kuhn, 2004), etc. The focus of this dissertation is on improving translation, but it is likely the work described here will benefit the other tasks mentioned as well. At the current time, the word alignment models developed for annotating translational correspondence are the same models used in approaches to exploiting corpora of parallel sentences for all of these tasks.

Automatic word alignment is not a solved problem. Many MT systems are trained in an alignment process based on the IBM Model 4 word alignment model (Brown et al., 1993). This process involves post-processing the output of Model 4 using heuristics. When evaluated on properly annotated gold standard English/French data, which is a relatively easy language pair for automatic word alignment systems, this approach has only $69\%$ balanced F-measure. F-measure is a trade-off between two factors, called Precision and Recall. Precision is the percentage of the links we hypothesized which are actually correct, and Recall is the percentage of the correct links which we hypothesized. Balanced F-Measure is the geometric mean of these two numbers. The graph in Figure 1.2 shows a gold standard annotation and a hypothesized annotation (marked

by a dashed line). Note the errors. English “do not” should be aligned to French “ne” and “pas” but “not” is aligned to “ne” while “do” is not aligned. The words “to spend” should be aligned to “dépenser”, but only “spend” is aligned to “dépenser”. The word “british” is aligned to “colombie” and “columbia” is aligned to “brittanique”. The Precision of this hypothesized alignment, the number of correctly hypothesized links over the total number of hypothesized links, is $13/15$. The Recall of the hypothesized alignment, the number of correctly hypothesized links over the number of correct links, is $13/19$. Balanced F-Measure (the geometric mean of Precision and Recall) is $77\%$, meaning that this hypothesis is better than the average hypothesis from this system. The desire to improve automatic word alignment systems, so that there are less errors like these and therefore better machine translation performance is obtained, motivates our work.

## 1.2 Problems with Current Practices in Word Alignment

### 1.2.1 Building Translation Systems with Word Alignments

Before we can show the problems with the most widely used unsupervised word alignment approach for statistical machine translation (SMT), we need to briefly outline how SMT systems use word alignments.

SMT systems are usually broken down into two types of model, the translation model, which is a model of translational correspondence between the source and target languages, and the language model, which is a model of well-formed sentences in the target language. To translate a new source sentence, we look for a probability maxima of these two models, i.e. we search for a target string which is both a good mapping of the source string into a target string and is also a well-formed target sentence. The translation model is estimated using a word alignment of a bitext (a corpus of aligned sentences in the source and target languages). The language model is estimated from monolingual target language text. For further details on building SMT systems using alignments see Appendix B.1.

### 1.2.2 Unsupervised Alignments are Not the Best Alignments Possible for Translation

We would like to substantiate the claim that improved alignments will lead to improved MT systems. We show that there exist alignments of a fixed bitext which are significantly better for translation than the alignments generated by our unsupervised baseline system. We generate the improved alignments by using an "oracle", a system which tells our alignment system how to improve the alignments; it knows how to do this by "cheating". We measure statistical machine translation performance both when using our baseline alignment system, and compare this with using a "weak oracle" in Figure

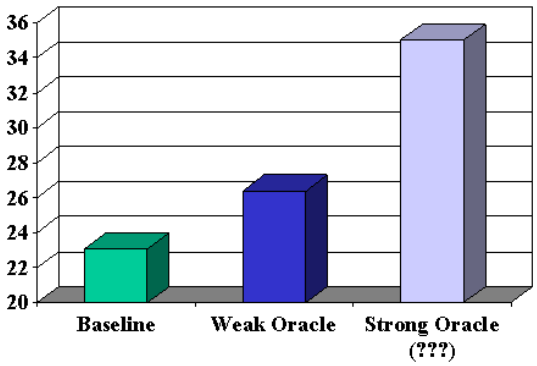


Figure 1.3: Comparison of baseline with a weak oracle showing that it is possible to improve MT performance by improving word alignment

1.3. We do this by using the BLEU metric (Papineni et al., 2001), which is an automatic translation evaluation metric which measures translation quality. BLEU has been shown to correlate well with human judgments of quality. The improved alignments from the "weak oracle" result in a BLEU score of 26.36; this is 3.30 points better than the baseline which is a large improvement. This shows that improving alignments can improve machine translation performance. See Appendix B.2 for a detailed explanation of this experiment. Even determining a good oracle for this problem is difficult. Our "weak oracle" is not the upper bound on performance. Given infinite computational resources we could find a "strong oracle" which would have better performance. We graphically depict this in the figure as well but note that the BLEU score such a "strong oracle" could obtain is unknown. We show later in the dissertation how to obtain improved word alignments without using an oracle.

### 1.2.3 Existing Metrics Do Not Track Translation Quality

There have been many research papers presented at ACL, NAACL, HLT, COLING, WPT03, WPT05, etc, outlining techniques for attempting to increase word alignment quality. However, although there are many results where an alignment system has successfully increased the score according to intrinsic metrics of word alignment quality, very few of these approaches has been shown to result in a large gain in translation performance. We show that this is because the two intrinsic word alignment quality metrics commonly used do not measure how useful alignments are for translation. These metrics are balanced F-Measure (Melamed, 2000) and Alignment Error Rate, or AER, (Och & Ney, 2003). We calculate the correlation between these metrics and the BLEU metric, and show that this correlation is low. A concise mathematical description of correlation is the coefficient of determination ($r^2$), which is the square of the Pearson product-moment correlation coefficient ($r$). For an alignment task using a commonly studied French/English data set, $r^2 = 0.16$ for the Alignment Error Rate (AER) metric, showing a low correlation with BLEU. For the same task and annotation, balanced F-Measure[1] has $r^2 = 0.20$, which also shows a low correlation with BLEU, see Chapter 2 for more details.

[1]This metric is referred to as "balanced F-Measure with Sure/Possible" later in the dissertation, see Chapter 2.

Chapter 2 presents a metric which has a high correlation with BLEU. This metric is shown to allow the derivation of an effective loss function for semi-supervised training in Chapter 4.

### 1.2.4 Existing Generative Models Make False Structural Assumptions

Our objective is to automatically produce alignments which can be used to build high quality machine translation systems. These are presumably close to the alignments that trained bilingual speakers produce. Human annotated alignments often contain M-to-N alignments, where several source words are aligned to several target words and the resulting unit can not be further decomposed. Source or target words in a single unit are sometimes non-consecutive. Unfortunately, existing generative alignment models can not model these alignments, because they make unrealistic assumptions about alignment structure.

Word alignments define minimal single or multi-word units in two parallel sentences which correspond to one another, which we will call “cepts” following Brown et al. (1993). Alignments for two examples (created by shortening sentences observed in “development” data) are shown in Figures 1.4 and 1.5. We concentrate on several interesting minimal translational correspondences listed in Table 1.6. The first two are taken from Figure 1.4 and the second two are taken from Figure 1.5. We now discuss the different alignment structure assumptions which have been made in previous work.

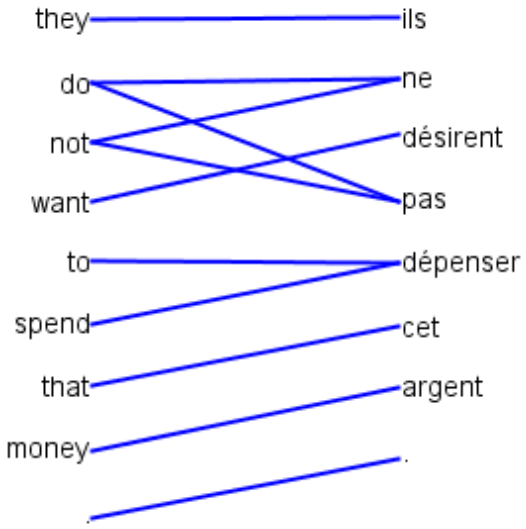


Figure 1.4: French/English gold standard word alignment, example 1

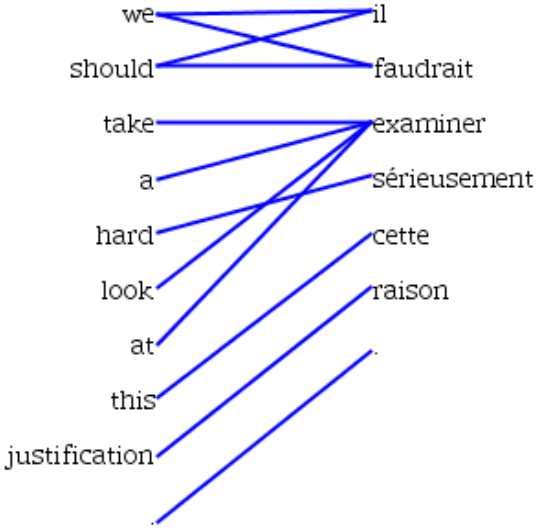


Figure 1.5: French/English gold standard word alignment, example 2

| English Cept | French Cept | 1-to-1 | 1-to-N | M-to-1 | phrase-based | M-to-N discontinuous |
|---|---|---|---|---|---|---|
| do not | ne pas | | | | | ✓ |
| to spend | dépenser | | | ✓ | ✓ | ✓ |
| we should | il faudrait | | | | ✓ | ✓ |
| take a look at | examiner | | | ✓ | | ✓ |

Figure 1.6: The impact of alignment structure assumptions

The use of the 1-to-N assumption is widespread, probably because of the success of the IBM word alignment models (Brown et al., 1993). 1-to-N alignments are alignments where one English word is aligned to zero or more French words, which need not be consecutive. Consider the 1-to-N alignment column in Table 1.6. In the first row, we see an example alignment which the IBM models are not able to model. The English cept: "do not" is aligned to the French cept: "ne ... pas" (which is a French negation construction), this is taken from Example 1 in Figure 1.4. This requires a

many to many, discontinuous alignment. This can not be modeled because under the 1-to-N assumption the English cept “do not” can not be modeled as a unit. In fact, the 1-to-N assumption can not be used to model any of the multi-word phrase mappings we have shown in Table 1.6. Of course, we can flip the direction and train such that one French word is aligned to zero or more English words. However, upon examining the M-to-1 column of Table 1.6, it becomes obvious that this assumption is also unsatisfactory. Many other generative models use the 1-to-N assumption, including the HMM model (Vogel et al., 1996) and other models based on the HMM model, for example the work of Toutanova et al. (2002), Lopez and Resnik (2005) and Deng and Byrne (2005).

What is done in practice in systems using the 1-to-N assumption of the IBM models is that the models are trained in both directions (English to French and French to English) and then “symmetrized” using a heuristic (Och & Ney, 2003; Koehn et al., 2003). If we allow ourselves to consider the best possible 1-to-N and M-to-1 alignments in Figures 1.4 and 1.5, can see several ways we might heuristically combine a 1-to-N alignment with a M-to-1 alignment. However, for more disparate language pairs (or longer French/English sentences), it is increasingly difficult to do this correctly. The use of a symmetrization heuristic also makes it problematic to calculate the probability of a final combined alignment as it is unclear how to combine the probabilities assigned by the two models.

There has also been a large amount of work on generative alignment models which model 1-to-1 word alignment structure (for instance the work of Wu (1997), Melamed (2000), Ahrenberg et al. (2000), Cherry and Lin (2003) and Liang et al. (2006)). None of the examples we have chosen in Table 1.6 can be modeled with this structure. These models have not been shown to perform for translation at the quality level of heuristic symmetrization of the 1-to-N and M-to-1 alignments produced using the IBM models. The claims made about the alignment quality for translation of these techniques are not well founded because they are based only on intrinsic metrics which unfortunately do not track how useful the generated alignments are for translation (as we discussed already in Section 1.2.3). 1-to-1 alignments are not generally used in practice to build machine translation systems.

Another common assumption is the phrase-based assumption, which is also used in translation in phrase-based MT systems (Och & Ney, 2004; Koehn et al., 2003). This assumption allows multiple word units to align to one another, but enforces the constraint that all words must be consecutive. For example, the Joint model (Marcu & Wong, 2002) typically aligns short segments of consecutive words to each other obeying this assumption. These models do not model discontinuous alignments. As shown in Table 1.6, this structure cannot be used to align the “ne ... pas” or “take a ... hard look” cepts in Examples 1 and 2 because they have gaps. Discontinuous alignments are important to achieve the best possible performance in translation. The strong

performance of the Hiero SMT model (Chiang, 2005), which uses such discontinuous alignments directly in the translation process, offers direct evidence to support this. Interestingly, even phrase-based SMT systems, which are already less flexible than hierarchical SMT systems in that they do not allow gaps in their translation rules, fair poorly when they are built from alignments which obey the phrase-based alignment assumption[2]. That even phrase-based SMT systems benefit from discontinuous alignments offers further evidence that discontinuous alignments are important to translation performance.

Since 2005 there have been a number of discriminative models introduced for the word alignment problem. Surprisingly, these models have suffered from the same structural assumptions. These models have either themselves directly required an unreasonable structural assumption, such as the work of Ittycheriah and Roukos (2005), Taskar et al. (2005), Liu et al. (2005), Fraser and Marcu (2006), Blunsom and Cohn (2006) and Lacoste-Julien et al. (2006), or they have used features derived from a generative model implemented with such a structural assumption in order to obtain the best performance, examples include the work of Ayan and Dorr (2006b), Lacoste-Julien et al.

[2]For example, a phrase-based SMT system can not learn both that the English cept "hard" translates as French "serieusement" and that the non-minimal "take a hard look at" translates as "examiner serieusement" in Figure 1.5, unless the alignment is able to represent the gap in the English cept "take a ... look at", which violates the phrasal alignment assumption.

(2006) and Moore et al. (2006). We will discuss discriminative models in detail in Chapter 4 and focus in particular on the structural assumptions made.

The inability of current generative models to model many-to-many discontinuous alignments is an important deficiency. We correct this problem. Our new generative model, LEAF, is able to model alignments which consist of many-to-many non-consecutive minimal translational correspondences directly, without the use of heuristics. LEAF is presented in Chapter 3. We show how to derive features from LEAF for use in a discriminative model in Chapter 4.

### 1.2.5 Many Existing Training Techniques Can Not Take Advantage of Manually Annotated Data

Until recently, start of the art translation systems were trained using an unsupervised training process which did not take advantage of manually annotated data. If we have access to a small amount of annotated word alignment data, we can shift from viewing alignment as an unsupervised problem to viewing alignment as a semi-supervised problem. In the last few years, this has become an active sub-area of word alignment research, but the advances according to various intrinsic word alignment metrics have not been shown to result in increased machine translation performance. Many research groups have continued to use unsupervised techniques to generate word alignments. As

we will show in Chapter 2, this is because the loss criteria being used do not reflect the usefulness of the generated alignments for machine translation.

In Chapter 4 we show that the alignment quality metric we will present in Chapter 2 is useful in the derivation of a loss function for use in semi-supervised training. If we have access to a small aligned set (we use up to 1,000 annotated sentence pairs), we can train a small number of important parameters directly, and discriminatively smooth richer sub-models[3] which would otherwise not be robustly estimated. If we have access to even more annotated data (we recently acquired data where we have up to 25,000 sentences), we can learn more parameters directly, but this is still only a fraction of the total parameters we need to align large corpora (for instance, we currently work on a task which involves aligning 10,000,000 parallel sentences which requires a very large number of parameters, most of which can not be estimated from a small corpus of 25,000 sentences).

We formulate a new model which is trained in a semi-supervised fashion in Chapter 4. This model uses rich features derived from our new generative model LEAF, but also allows for the easy integration of new knowledge sources which would be difficult to add to a generative story. This leads to large increases both in alignment accuracy (up

[3]Sub-models are sometimes also referred to as feature functions in the literature. We call them sub-models in our framework as a reminder that they themselves frequently have parameters which are estimated empirically.

to 9 F-score points) and translation accuracy (improvements of up to 2.8 BLEU points) over strong baselines.

### 1.2.6 It is Difficult to Add New Knowledge Sources to Generative Models

Current generative models depend on complex generative stories which must be completely reengineered each time a new knowledge source is added, blocking the easy introduction of new sources of linguistic knowledge to improve translation.

Consider again Figure 1.2. One problem with the hypothesized alignment is that "british" is aligned to "colombie" and "columbia" is aligned to "brittanique". If we were able to easily incorporate a knowledge source which used string similarity into our alignment model we might be able to overcome this problem. We show in Chapter 4 how to integrate a state of the art transliteration model used for the transliteration of names from Arabic to English. We also show how to incorporate a small fully supervised model estimated from 25,000 sentences, as we discussed in the previous section. Most of the approaches to discriminative word alignment models presented in the last two years, for example the work of Liu et al. (2005), Ittycheriah and Roukos (2005), Taskar et al. (2005), Ayan and Dorr (2006b), Lacoste-Julien et al. (2006), Fraser and Marcu (2006), Blunsom and Cohn (2006) and Moore et al. (2006), have

also addressed the problem of integrating disparate knowledge sources, which shows its importance.

## 1.3 Dissertation Approaches in Brief

We have shown that improvements in word alignment quality can help MT performance in Section 1.2.2. We present the problems we address and the approaches to solving them in brief:

1. As we discussed in Section 1.2.3, existing metrics for word alignment quality do not predict translation quality. To address this shortcoming, we describe a method for automatically measuring alignment quality which is related to improvements in resulting translation quality. Determining how to measure word alignment quality for automatic translation is addressed in Chapter 2.

2. As shown in Sections 1.2.4, existing generative models for word alignment make false structural assumptions. To address this problem, we improve word alignment modeling by designing a statistical model which directly models the full structure of the word alignment problem. Improving word alignment modeling with better structure is addressed in Chapter 3.

3. As discussed in Section 1.2.5 and 1.2.6 respectively, existing training techniques for word alignment models will not allow us to take advantage of manually annotated word alignments, and do not allow for easy integration of new knowledge sources. To address this issue, we develop a new semi-supervised training algorithm. This algorithm automatically takes advantage of whatever quantity of manually annotated data can be obtained, allows for the robust training of powerful models, and enables an easy integration of new knowledge sources. Improving word alignment training using semi-supervised learning is addressed in Chapter 4.

# Chapter 2

# Intrinsic Metrics for Measuring the Quality of Word Alignment for Translation

Automatic word alignment plays a critical role in statistical machine translation. Unfortunately the relationship between alignment quality and statistical machine translation performance has not been well understood. In the recent literature, the alignment task has frequently been decoupled from the translation task and assumptions have been made about measuring alignment quality for machine translation which, it turns out, are not justified. In particular, none of the tens of papers published over the last five years have shown that significant decreases in Alignment Error Rate, AER (Och & Ney, 2003), result in significant increases in translation performance. We explain this state of affairs and present a method for measuring alignment quality in a way which is predictive of statistical machine translation performance.

## 2.1 Introduction

Automatic word alignment (Brown et al., 1993) is a vital component of all statistical machine translation (SMT) approaches. There were a number of research papers presented from 2000 to 2005 at ACL, NAACL, HLT, COLING, WPT03, WPT05, etc, outlining techniques for attempting to increase word alignment quality. Despite this high level of interest, none of these techniques has been shown to result in a large gain in translation performance as measured by BLEU (Papineni et al., 2001) or any other translation quality metric. We find this lack of correlation between previous word alignment quality metrics and BLEU counter-intuitive, because we and other researchers have measured this correlation in the context of building SMT systems that have benefited from using the BLEU metric in improving performance in open evaluations such as the NIST evaluations.[1]

[1]Since in our experiments we use BLEU to compare the performance of systems built using a common framework where the only difference is the word alignment, we make no claims about the utility of BLEU for measuring translation quality in absolute terms, nor its utility for comparing two completely different MT systems. We only assume that BLEU tracks translation quality differences caused by the effects of different word alignments of a fixed bitext. This is a much less general assumption than assuming that BLEU can be used to compare, for instance, a rule-based system and a statistical machine translation system, or two statistical machine translation systems which were trained on differing bitext and/or monolingual text. We argue that any systematic changes to the alignments which result in better BLEU scores on an unseen test set (i.e. changes which are made without examination of that test set) must be viewed as improvements to the alignments for the automatic translation task.

We confirm experimentally that previous metrics do not predict BLEU well and develop a methodology for measuring alignment quality which is predictive of BLEU. We also show that AER is not correctly derived from F-Measure and is therefore unlikely to be useful as a metric.

## 2.2 Experimental Methodology

### 2.2.1 Data

To build an SMT system we require a bitext and a word alignment of that bitext, as well as language models built from target language data. In all of our experiments, we will hold the bitext and target language resources constant, and only vary how we construct the word alignment.

The gold standard word alignment sets we use have been manually annotated using links between words showing minimal translational correspondences. Links which must be present in a hypothesized alignment are called "Sure" links. Some of the alignment sets also have links which are not "Sure" links but are "Possible" links (Och & Ney, 2003). "Possible" links which are not "Sure"[2] may be present but need not be present.

[2] "Sure" links are by definition also "Possible".

We evaluate the translation performance of SMT systems by translating a held-out translation test set and measuring the BLEU score of our hypothesized translations against one or more reference translations. We also have an additional held-out translation set, the development set, which is employed by the MT system to train the weights of its log-linear model to maximize BLEU (Och, 2003). We work with data sets for three different language pairs, examining French to English, Arabic to English, and Romanian to English translation tasks.

The training data for the French/English data set is taken from the LDC Canadian Hansards data set, from which the word aligned data (presented by Och and Ney (2003)) was also taken. The English side of the bitext is 67.4 million words. We used a separate Canadian Hansards data set (released by ISI) as the source of the translation test set and development set. We evaluate two different tasks using this data, a medium task where 1/8 of the data (8.4 million English words) is used as the fixed bitext, and a large task where all of the data is used as the fixed bitext. The 484 sentences in the gold standard word alignments have 4,376 Sure Links and 19,222 Possible links. See alignment set A in Table 2.1 for the data statistics (note that alignment sets B and C will be introduced later).

Table 2.1: French/English Dataset

| | | FRENCH | ENGLISH |
|---|---|---|---|
| MEDIUM TRAINING | SENTENCES | 355,273 | |
| | WORDS | 9,487,633 | 8,438,050 |
| | VOCABULARY | 65,239 | 49,121 |
| | SINGLETONS | 25,622 | 19,253 |
| LARGE TRAINING | SENTENCES | 2,842,184 | |
| | WORDS | 75,794,254 | 67,366,819 |
| | VOCABULARY | 149,568 | 114,907 |
| | SINGLETONS | 60,651 | 47,765 |
| TRANSLATION DEV | SENTENCES | 833 | |
| | WORDS | 20,562 | 17,454 |
| TRANSLATION TEST | SENTENCES | 2,380 | |
| | WORDS | 58,990 | 49,182 |
| ALIGNMENT SET A | SENTENCES | 484 | |
| | WORDS | 8,482 | 7,681 |
| | SURE LINKS | 4,376 | |
| | POSSIBLE LINKS | 19,222 | |
| ALIGNMENT SET B | SENTENCES | 110 | |
| | WORDS | 1,888 | 1,726 |
| | SURE LINKS | 1,037 | |
| | POSSIBLE LINKS | 3,989 | |
| ALIGNMENT SET C | SENTENCES | 110 | |
| | WORDS | 1,888 | 1,726 |
| | SURE LINKS | 2,292 | |

The Arabic/English training corpus is the data used for the NIST 2004 machine translation evaluation[3]. The English side of the bitext is 99.3 million words. The translation development set is the "NIST 2002 Dry Run", and the test set is the "NIST 2003 evaluation set". We have annotated gold standard alignments for 100 parallel sentences using Sure links, following the Blinker guidelines (Melamed, 1998) which calls for Sure links only (there were 2,154 Sure links). Here we also examine a medium task

[3]http://www.nist.gov/speech/tests/summaries/2004/mt04.htm

Table 2.2: Arabic/English Dataset

| | | Arabic | English |
|---|---|---|---|
| Medium Training | Sentences | 482,965 | |
| | Words | 11,218,869 | 12,424,253 |
| | Vocabulary | 185,441 | 77,298 |
| | Singletons | 81,565 | 34,645 |
| Large Training | Sentences | 3,863,718 | |
| | Words | 89,705,083 | 99,326,492 |
| | Vocabulary | 426,746 | 191,349 |
| | Singletons | 143,552 | 77,430 |
| Translation Dev | Sentences | 203 | |
| | Words | 5,039 | 6.4K to 7.0K |
| Translation Test | Sentences | 663 | |
| | Words | 16,491 | 19.0K to 21.7K |
| Alignment Set | Sentences | 100 | |
| | Words | 1,747 | 2,029 |
| | Sure Links | 2,154 | |

using 1/8 of the data (12.4 million English words) and a large task using all of the data. Note that we had four references available for the translation test set and translation development set (used for training Maximum BLEU), which allowed the use of less test sentences than for the other data sets where we used much larger translation development and test sets because we only had access to one reference translation. See Table 2.2 for the data statistics.

The Romanian/English training data was used for the tasks on Romanian/English alignment at WPT03 (Mihalcea & Pederson, 2003) and WPT05 (Martin et al., 2005). We carefully removed two sections of news bitext to use as the translation development and test sets. The English side of the training corpus is 964,000 words. The gold

Table 2.3: Romanian/English Dataset

| | | ROMANIAN | ENGLISH |
|---|---|---|---|
| SMALL TRAINING | SENTENCES | 45,241 | |
| | WORDS | 913,806 | 963,615 |
| | VOCABULARY | 44,390 | 24,918 |
| | SINGLETONS | 18,865 | 8,473 |
| TRANSLATION DEV | SENTENCES | 800 | |
| | WORDS | 15,864 | 16,896 |
| TRANSLATION TEST | SENTENCES | 2,400 | |
| | WORDS | 46,740 | 48,758 |
| ALIGNMENT SET | SENTENCES | 148 | |
| | WORDS | 2,773 | 2,875 |
| | SURE LINKS | 3,181 | |

standard alignment set is the first 148 annotated sentences used for the 2003 task (there were 3,181 Sure links). For the data statistics see Table 2.3.

### 2.2.2 Measuring Translation Performance Changes Caused By Alignment

In phrased-based SMT (Koehn et al., 2003) the knowledge sources which vary with the word alignment are the phrase translation lexicon (which maps source phrases to target phrases using counts from the word alignment) and some of the word level translation parameters (sometimes called lexical smoothing). However, many knowledge sources do not vary with the final word alignment, such as scores assigned using IBM Model 1, N-gram language models and the length penalty. In our experiments, we use a state of the art phrase-based system, similar to (Koehn et al., 2003). The weights of the different knowledge sources in the log-linear model used by our system are trained

using Maximum BLEU (Och, 2003), which we run for 25 iterations individually for each system. Two language models are used, one built using the target language training data and the other built using additional news data.

### 2.2.3 Generating Alignments of Varying Quality

We have observed in the past that generative models used for statistical word alignment create alignments of increasing quality as they are exposed to more data. The intuition behind this is simple; as more co-occurrences of source and targets words are observed, the word alignments are better. If we wish to increase the quality of a word alignment, we allow the alignment process access to extra data which is used only during the alignment process and then removed. If we wish to decrease the quality of a word alignment, we divide the bitext into pieces and align the pieces independently of one another, finally concatenating the results together.

To generate word alignments we use GIZA++ (Och & Ney, 2003), which implements both the IBM Models (Brown et al., 1993) and the HMM word alignment model (Vogel et al., 1996). We use Model 1, HMM, and Model 4 in that order. The output of these models is an alignment of the bitext which projects one language to another. GIZA++ is run end-to-end twice. In one case we project the source language to the target language (producing the “1-to-N” alignment), and in the other we project the target language to the source language (producing the “M-to-1” alignment). The output of

GIZA++ is then post-processed using the three "symmetrization heuristics" described by Och and Ney (2003), "Union", "Intersection" and "Refined". We evaluate our approaches using these heuristics because we would like to account for alignments generated in different fashions. These three heuristics were used as the baselines in virtually all recent work on automatic word alignment, and many of the best SMT systems use these techniques as well.

The "Union" heuristic simply combines the links in the 1-to-N alignment with the links in the M-to-1 alignment, and usually has a higher recall than either of the starting alignments. The "Intersection" heuristic takes only those links occurring in both alignments, and usually results in a higher precision than either of the starting alignments. The "Refined" symmetrization heuristic starts from the intersection of the two alignments and adds links from the union, and usually has higher precision than the union of the 1-to-N and M-to-1 alignments and higher recall than the intersection of these alignments.

We describe the "Refined" symmetrization heuristic in further detail. The first step in applying the heuristic is to take the intersection of the 1-to-N and M-to-1 alignments and store the links into a set $A$. We then take the union of the 1-to-N and M-to-1 alignments and subtract $A$, resulting in a set $A'$ of the links in only one of the two alignments. Each link in $A'$ is then considered for addition to $A$. A link $(i, j)$ connecting the source word at position $i$ with the target word at position $j$ is added to $A$ if a

"neighboring" link is already in $A$, and subject to an additional constraint which we will describe. The "neighboring" links to $(i, j)$ are the links $(i, j + 1)$, $(i, j - 1)$, $(i + 1, j)$ and $(i - 1, j)$. The constraint is that the addition of $(i, j)$ must not result in $A$ containing any link $(i', j')$ such that both the source word at $i'$ and the target word at $j'$ are involved in more than one link in $A$. Once no further link addition can be performed, $A$ is returned as the result. In practice, an implementation expands outwards from each link in the intersection, and requires defining both the order in which the links in the intersection are visited, and the order in which the neighbors to a visited link are checked for addition. The usage of the "Refined" symmetrization heuristic results in a symmetrized alignment consisting of minimal translational correspondences which are either 1-to-N or M-to-1 and consist of consecutive words only.

In this work, when applying the "Union" symmetrization heuristic we take the transitive closure of the bipartite graph created, which results in fully connected components indicating minimal translational correspondence[4]. All of the alignments in Figure 2.1 are equivalent from a translational correspondence perspective and the first two will be mapped to the third in order to ensure consistency between the number of links an alignment has and the translational equivalences licensed by that alignment.

[4]We have no need to do this for the "Refined" and "Intersection" heuristics, because they only produce alignments in which the components are already fully connected.

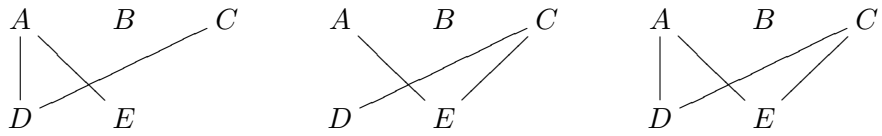


Figure 2.1: All of these alignments are equivalent from a translational correspondence perspective

## 2.3 Word Alignment Quality Metrics

### 2.3.1 Alignment Error Rate is Not a Useful Measure

We begin our study of metrics for word alignment quality by testing Alignment Error Rate (AER) (Och & Ney, 2003). AER requires a gold standard manually annotated set of Sure links and Possible links (referred to as $S$ and $P$). Given a hypothesized alignment consisting of the link set $A$, three measures are defined:

$$\text{Precision}(A, P) = \frac{|P \cap A|}{|A|} \qquad \text{if } (/A/ > 0), \qquad 1 \text{ otherwise} \tag{2.1}$$

$$\text{Recall}(A, S) = \frac{|S \cap A|}{|S|} \qquad \text{if } (/S/ > 0), \qquad 1 \text{ otherwise} \tag{2.2}$$

$$\text{AER}(A, P, S) = 1 - \frac{|P \cap A| + |S \cap A|}{|S| + |A|} \qquad \text{if } ((/S/ + /A/) > 0), \qquad 0 \text{ otherwise} \tag{2.3}$$

Och and Ney's definition of Precision measures the percentage of links in our hypothesized set which are Possible (note that Precision decreases from 1 only as links which are not even Possible are hypothesized, and note that all Sure links are by definition Possible). Recall measures the percentage of the links in the Sure set which have been hypothesized (note that Possible links may either be hypothesized or not hypothesized, this does not affect Recall). In order for a hypothesis to be 100% correct (i.e. have Precision=1 and Recall=1), all of the links in the Sure set must be hypothesized, and any additional links hypothesized must be in the Possible set.

In our graphs, we will present $1 -$ AER so that we have an accuracy measure.

We created alignments of varying quality for the medium French/English training set. We broke the parallel text into separate pieces corresponding to 1/16, 1/8, 1/4 and 1/2 of the original parallel text to generate degraded alignments, and we used 2, 4, and 8 times the original parallel text to generate enhanced alignments. In all cases we use only the alignment of the original parallel text to build a MT system for measuring BLEU. For the "fractional" alignments we report the average AER of the pieces[5].

[5]For example, for 1/16, we perform 16 pairs of alignments (a pair of alignments is a 1-to-N alignment and a M-to-1 alignment), each of which includes the full gold

The graph in Figure 2.2 shows the correlation of $1 -$ AER with BLEU. High correlation would look like a line from the bottom left corner to the top right corner. As can be seen by looking at the graph, there is low correlation between $1 -$ AER and the BLEU score. A concise mathematical description of correlation is the coefficient of determination ($r^2$), which is the square of the Pearson product-moment correlation coefficient ($r$). Here, $r^2 = 0.16$, which is low.

The correlation is low because of a significant shortcoming in the mathematical formulation of AER which to our knowledge has not been previously reported. Och and Ney (2003) state that AER is derived from F-Measure. But AER does not share a very important property of F-Measure, which is that unbalanced precision and recall are penalized, where $S \subset P$ (i.e. when we make the Sure versus Possible distinction, meaning that $S$ is a proper subset of $P$)[6]. We will show this using an example.

We first define the measure "F-Measure with Sure and Possible" using Och and Ney's Precision and Recall formulas together with the standard F-Measure formula (Rijsbergen, 1979). In the F-Measure formula (2.4) there is a parameter $\alpha$ which sets the trade-off between Precision and Recall. When an equal trade-off is desired, $\alpha$ is set to $0.5$.

---

standard text. We perform another 16 pairs of alignments without the gold standard text. We then apply the symmetrization heuristics to each these pairs. We use the symmetrized alignments including the text from the gold standard set to measure AER and take the average. We concatenate the symmetrized alignments not including the gold standard text to build SMT systems for measuring BLEU.

[6]Note that if $S = P$ then 1-AER reduces to balanced F-Measure

$$\text{F-measure with Sure and Possible}(A, P, S, \alpha) = \frac{1}{\frac{\alpha}{\text{Precision}_{(A,P)}} + \frac{(1-\alpha)}{\text{Recall}_{(A,S)}}} \quad (2.4)$$

We compare two hypothesized alignments where $|A|$, the number of hypothesized alignment links, is the same; for instance, $|A| = 100$. Let $|S| = 100$. In the first case, let $|P \cap A| = 50$ and $|S \cap A| = 50$. Precision is $0.50$ and Recall is $0.50$. In the second case, let $|P \cap A| = 75$ and $|S \cap A| = 25$. Precision is $0.75$ and Recall is $0.25$.

The basic property of F-Measure, if we set $\alpha$ equal to $0.5$, is that unbalanced precision and recall should be penalized. The first hypothesized alignment has an F-Measure with Sure and Possible score of $0.50$, while the second has a worse score, $0.375$.

However, if we substitute the relevant values into the formula for AER (Equation 2.3), we see that $1-$AER for both of the hypothesized alignments is $0.5$. Therefore AER does not share the property of F-Measure (with $\alpha = 0.5$) that unbalanced precision and recall are always penalized. Because of this, it is possible to maximize AER by favoring precision over recall, which can be done by simply guessing very few alignment links. Unfortunately, when $S \subset P$, this leads to strong biases, which makes AER not useful as a metric.

Goutte et al. (2004) previously observed that AER could be unfairly optimized by using a bias towards precision which was unlikely to improve the usefulness of the alignments. Possible problems with AER were discussed at WPT 2003 and WPT 2005.

Examining the graph in Figure 2.3, we see that F-Measure with Sure and Possible has some predictive power for the data points generated using a single heuristic, but the overall correlation is still low, $r^2 = 0.20$. We need a measure which predicts BLEU without having a dependency on the way the alignments are generated.

### 2.3.2 Balanced F-Measure is Better, but Still Inadequate

We wondered whether the low correlation was caused by the Sure and Possible distinction. We re-annotated the first 110 sentences of the French test set using the Blinker guidelines (Melamed, 1998), there were 2,292 Sure links. This is alignment set C in Table 2.1. We define F-Measure without the Sure versus Possible distinction (i.e., all links are Sure) in Equation 2.5, and set $\alpha = 0.5$. This measure has been extensively used with other word alignment test sets. Figure 2.4 shows the results. Correlation is higher, $r^2 = 0.67$.

$$\text{F-measure}(A, S, \alpha) = \frac{1}{\frac{\alpha}{\text{Precision}_{(A,S)}} + \frac{(1-\alpha)}{\text{Recall}_{(A,S)}}} \tag{2.5}$$

### 2.3.3 Varying the Trade-off Between Precision and Recall Works Well

We then hypothesized that the trade-off between precision and recall is important. This is controlled in both formulas by the constant $\alpha$. We searched $\alpha = 0.1, 0.2, ..., 0.9$ for the best $r^2$ value. The best results were: $\alpha = 0.1$ for the original annotation annotated with Sure and Possible (see Figure 2.5), and $\alpha = 0.4$ for the first 110 sentences as annotated by us (see Figure 2.6)[7]. The relevant $r^2$ scores were $0.80$ and $0.85$ respectively. With a good $\alpha$ setting, we are able to predict the machine translation results reasonably well. For the original annotation, recall is very highly weighted, while for our annotation, recall is still more important than precision[8]. Our results also suggest that better correlation will be achieved when using Sure-only annotation than with Sure and Possible annotation.

We then tried the medium Arabic training set. Results are shown in figure 2.8, the best setting of $\alpha = 0.1$, and $r^2 = 0.93$. F-Measure is effective in predicting machine translation performance for this set.

We also tried the larger tasks, where we can only decrease alignment quality as we have no additional data. For the large French/English corpus the best results are with

[7]We also checked the first 110 sentences using the original annotation to ensure that the differences observed were not an effect of restricting our annotation to these sentences, see alignment set B in Table 2.1

[8]$\alpha$ less than $0.5$ weights recall higher, while $\alpha$ greater than $0.5$ weights precision higher, see the F-Measure formulas.

$\alpha = 0.2$ for the original annotation of 484 sentences and $\alpha = 0.4$ for the new annotation of 110 sentences with only Sure links (see Figure 2.7). Relevant $r^2$ scores were $0.62$ and $0.64$ respectively. Disappointingly, our measures are not able to fully explain MT performance for the large French/English task.

For the large Arabic/English corpus, the results were better, the best correlation was at $\alpha = 0.1$, for which $r^2 = 0.90$ (see Figure 2.9). We can predict MT performance for this set. It is worth noting that the Arabic/English translation task and data set has been tested in conjunction with our translation system over a long period, but the French/English translation task and data has not. As a result, there may be hidden factors that affect the performance of our MT system which only appear in conjunction with the large French/English task.

One well-studied task on a smaller data set is the Romanian/English shared word alignment task from the Workshop on Parallel Text at ACL 2005 (Martin et al., 2005). We only decreased alignment quality and used 5 data points for each symmetrization heuristic due to the small bitext. The best setting of $\alpha$ was $\alpha = 0.2$, for which $r^2 = 0.94$ (see Figure 2.10), showing that F-Measure is again effective in predicting BLEU.

## 2.4 Previous Work

Most previous work on measuring alignment quality has focused on comparison of a hypothesis with a gold standard word alignment using some type of distance metric,

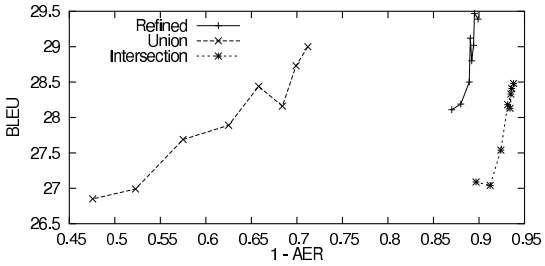


Figure 2.2: French $1-$AER versus BLEU, $r^2 = 0.16$

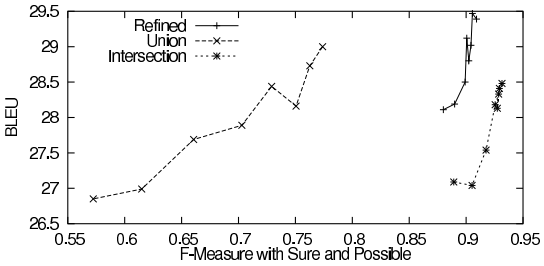


Figure 2.3: French F-Measure with Sure and Possible $\alpha = 0.5$ versus BLEU, $r^2 = 0.20$

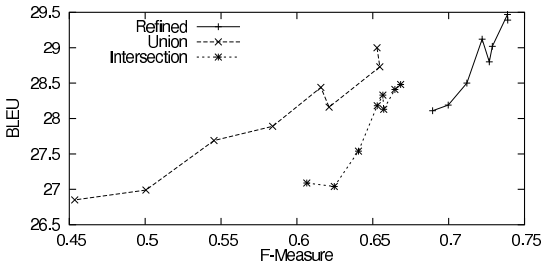


Figure 2.4: French F-Measure $\alpha = 0.5$ versus BLEU, $r^2 = 0.67$

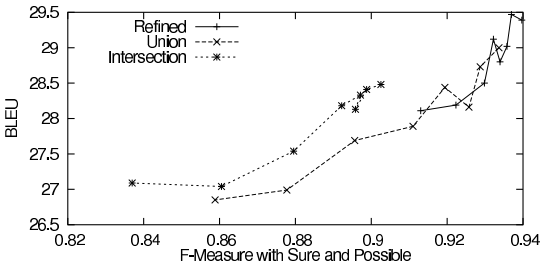


Figure 2.5: French F-Measure with Sure and Possible $\alpha = 0.1$ versus BLEU, $r^2 = 0.80$

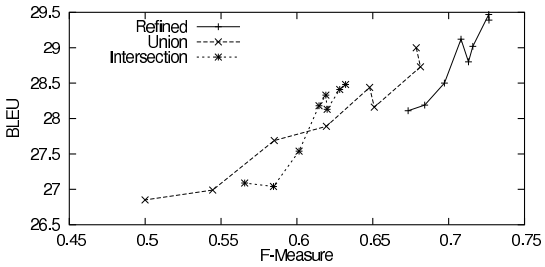


Figure 2.6: French F-Measure $\alpha = 0.4$ versus BLEU, $r^2 = 0.85$

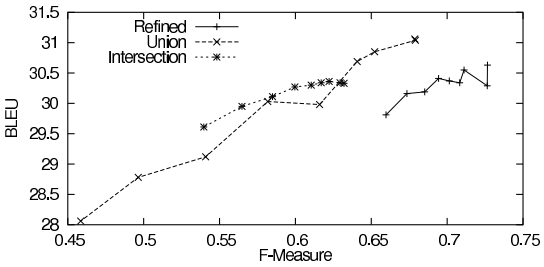


Figure 2.7: Large French F-Measure $\alpha = 0.4$ (110 sentences) versus BLEU, $r^2 = 0.64$

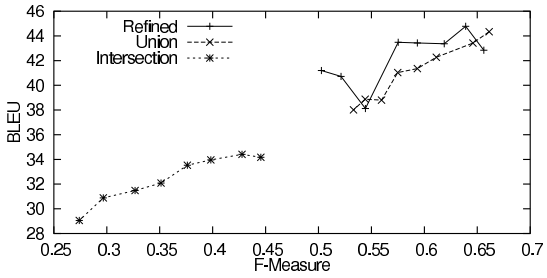


Figure 2.8: Arabic F-Measure $\alpha = 0.1$ versus BLEU, $r^2 = 0.93$

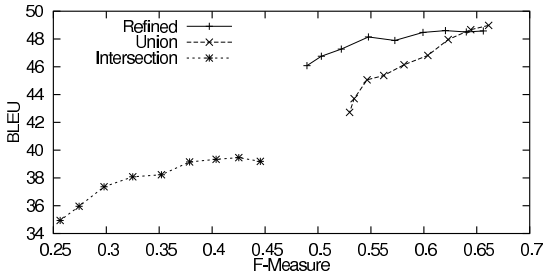


Figure 2.9: Large Arabic F-Measure $\alpha = 0.1$ (100 sentences) versus BLEU, $r^2 = 0.90$

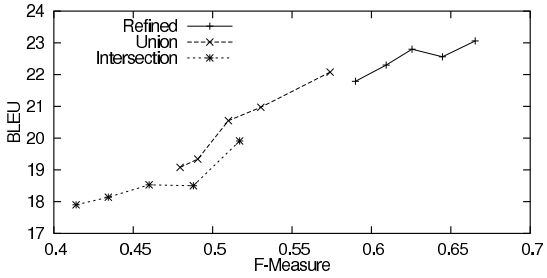


Figure 2.10: Small Romanian F-Measure $\alpha = 0.2$ (148 sentences) versus BLEU, $r^2 = 0.94$

much as our work does. The differences between these studies have focused primarily on the weighting of the links in a single minimal translational correspondence, examining how each of the word level links should be weighted (e.g., should the link in a 1-to-1 correspondence be considered to have equal weight with one of the links in a 1-to-2 correspondence, or should it have the same weight as both combined? How should the links in a 2-to-2 correspondence, which involves four links at the word level, be weighted?). Based on our investigations this does not appear to be as important

as the trade-off between the loss involved in predicting incorrect links versus the loss involved in not predicting correct links which is tuned using $\alpha$ in the F-Measure formula. Melamed (2000) has a formula for weighting the links in large minimal units of translation to avoid giving these units too much weight. The basic idea of this metric is that the sum of all links to a word should have a constant weight. Och and Ney (2003) claim that using the Sure and Possible links defined for F-measure with Sure and Possible helps determine how to correctly weight non-compositional links, but our experiments cast doubt on whether this is necessary because we have shown evidence that F-Measure with Sure and Possible is not more effective than simple F-Measure. Other approaches to dealing with non-compositional links have been tried. Davis (2002) has a metric similar to Melamed's which implements the same weighting idea of words having constant weight, but in a simpler fashion. Ahrenberg et al. (2000) develop simple link precision/recall as the basis for a metric to evaluate the alignment of multiple English words to the large compound words which are common in Germanic languages such as Swedish and German. None of these metrics have been shown to be useful extrinsically, for measuring machine translation performance or measuring performance for any other task. These metrics do have one advantage over F-Measure, which is that they do not require tuning the $\alpha$ parameter for each new task. However, our results show that the best trade-off between Precision and Recall varies by alignment task,

so varying this trade-off will likely be required in any successful approach involving comparison of hypothesized word links with a gold standard.

There are also approaches to measuring word alignment quality which do not involve using a gold standard word alignment of a small sample of parallel sentences, but instead building a translation lexicon from the whole alignment. Wu and Xia (1995) sample the translation lexicon and uses both manual and automatic filters to measure precision. Melamed (2000) takes a sample from the translation lexicon and measures probability weighted precision manually, and then he uses this to estimate probability weighted recall. Koehn and Knight (2002) evaluate a translation lexicon by counting how many of the entries are found in a dictionary, which we find interesting as it is automatic, but it is limited as dictionary entries will likely only exist for matches between the frequent senses of content words (without accompanying function words). The commonality of these approaches lies in using an abstract implicit context, whether that used for the translation dictionary or that used in a manual evaluation, where the evaluators directly judge translational correspondence without observing the context in which the presumed correspondence occurs. Our approach is superior, at least for the task of data driven machine translation, in that it evaluates alignment accuracy in the observed context of parallel sentences where many of the minimal translational correspondences are only contextually motivated and would not apply to all contexts. We expect our translation system to learn not only idealized translations applicable in any

context, which are what is found in a translation dictionary, but also translations which are contextually motivated and may apply only in certain contexts. If we do not learn the latter type of translations we are failing to take full advantage of our (limited) training data.

Appearing somewhat later than our study, two recent papers have looked at the relationship between alignment accuracy and translation performance. Lopez and Resnik (2006) looked at the impact of alignments on phrase-based MT for a Chinese/English task using 30M words of English and 27M words of Chinese. We found this study interesting in that it showed evidence that phrase-based MT systems become less sensitive to alignment quality as training size increases, which we also found in our study. This appears to be due to a saturation of the parameters in basic phrase-based MT models which do not model context as richly as newer approaches such as hierarchical models and supervised syntactic models. Ayan and Dorr (2006a) looked at the same trade-off between Precision and Recall that we examined. They study small alignment tasks for Chinese/English (4.1 M English words) and Arabic/English (1.4 M English words). This work only considered a single lower recall alignment and a single lower precision alignment along with three other alignments. One of the contributions is the definition of an error metric called CPER, which equally weights Precision and Recall over phrases extracted from the hypothesized alignment with respect to phrases extracted from the gold alignment, but unfortunately they were unable to show that this metric is

an effective predictor of MT performance. Both of these studies are limited to generalizations about phrase-based MT models for small to medium sized tasks. As we will show in Chapter 4, our metric can be used to derive a loss function to produce not only improved alignments for phrase-based MT but also improved alignments for hierarchical and supervised syntactic MT models, which use richer context and more structure than phrase-based MT and are therefore more likely to be affected by alignment quality at large training data sizes. Additionally, we have shown that there is not a single best trade-off between Precision and Recall for all alignment tasks, but instead there is a significant difference in the best trade-off depending on the task. For instance, our research shows that the best results for large Chinese/English tasks tend to favor balanced Precision and Recall, a finding which is not inconsistent with the observation of Ayan and Dorr (2006a) on small Chinese/English data tasks. However, obtaining the best results for large Arabic/English tasks requires strongly favoring Recall, which is opposite the conclusion for small Arabic/English tasks reached by Ayan and Dorr (2006a).

Our work invalidates some of the conclusions of recent alignment work which presented only evaluations based on metrics like AER or balanced F-Measure, and explains the lack of correlation in the few works which presented both such a metric and final MT results. A good example of the former are our own results (Fraser & Marcu, 2005). The work presented there had the highest balanced F-Measure scores for the Romanian/English WPT05 shared task, but based on the findings here it is possible that a

different alignment algorithm tuned for the correct criterion would have had better MT performance. Other work includes many papers working on alignment models where words are allowed to participate in a maximum of one link. These models generally have higher precision and lower recall than IBM Model 4 symmetrized using the "Refined" or "Union" heuristics. But we showed that AER is broken in a way that favors precision in Section 2.3.1. It is therefore likely that the results reported in these papers are affected by the AER bias and that the corresponding improvements in AER score do not correlate with increases in phrasal SMT performance. We will show further evidence that F-Measure with a tuned trade-off between Precision and Recall is effective by using this metric to derive a loss criterion in discriminative modeling in Chapter 4.

While we have addressed measuring alignment quality for phrasal SMT, similar work is now required to see how to measure alignment quality for other tasks. For an evaluation campaign the organizers should pick a specific task, such as improving phrasal SMT, and calculate an appropriate $\alpha$ to be used. Individual researchers working on the same phrasal SMT tasks as those reported on here (or very similar tasks) could use the values of $\alpha$ we calculated.

## 2.5 Summary

We have presented an empirical study of the use of simple evaluation metrics based on gold standard alignment of a small number of sentences to predict machine translation performance. Based on our experiments we can now draw the following conclusions:

1. We measured the correlation between our unbalanced F-Measure metric and BLEU. Good correlation was obtained for the medium French and Arabic data sets, the large Arabic data set and the small Romanian data set. We have explained most of the effect of alignment quality on these sets, and if we are given the F-measure of a hypothesized word alignment for the bitext we can make a reasonable prediction as to what the resulting BLEU score will be.

2. We recommend using the Blinker guidelines as a starting point for new alignment annotation efforts, and that Sure-only annotation be used. If a larger gold standard is available and was already annotated using the Sure versus Possible distinction, this is likely to have only slightly worse results.

3. When we make the distinction between Sure and Possible links, AER does not share the important property of F-Measure that unequal precision and recall are penalized, making it easy to obtain good AER scores by simply guessing less alignment links. As a result AER is a misleading metric which should no longer be used.

We suggest comparing alignment algorithms by measuring performance in an identified final task such as machine translation. F-Measure with an appropriate setting of $\alpha$ will be useful during the development process of new alignment models, or as a maximization criterion for discriminative training of alignment models. We will return to the topic of discriminative training in Chapter 4, where we will use our new metric to derive a loss function in conjunction with a semi-supervised training algorithm, and show that this improves translation quality.

## 2.6 Research Contribution

We found an automatic intrinsic metric which measures word alignment quality for the translation task in a better fashion than the currently used metrics.

In addition, this metric will be shown to be useful to derive a loss function for semi-supervised training in Chapter 4.

# Chapter 3

# Improving Structural Assumptions with a New Many-to-Many Discontinuous Generative Alignment Model

Previous generative word alignment models have made unreasonable assumptions about the desired word alignment structure, which do not match the alignment structure used to build statistical machine translation systems. Previous discriminative models have either made such an assumption directly or used features derived from a generative model making one of these assumptions.

Two incorrect word alignment structures are particularly common. The first is the 1-to-N assumption, meaning that each source word generates zero or more target words, which requires heuristic techniques in order to obtain alignments suitable for training a SMT system. The second is the consecutive word-based "phrasal SMT" assumption.

This does not allow gaps in minimal translation correspondences. We discussed the problems with these word alignment structure assumptions in Section 1.2.4, and we will discuss these issues further in Section 3.6, which outlines previous work on generative models of word alignment.

Our objective is to automatically produce alignments which can be used to build high quality machine translation systems. These are presumably close to the alignments that trained bilingual speakers produce. Human annotated alignments often contain M-to-N alignments, where several source words are aligned to several target words and the resulting unit can not be further decomposed. Source or target words in a single unit are sometimes non-consecutive.

We describe a new generative model, LEAF, which directly models M-to-N non-consecutive word alignments.

## 3.1 Introduction

For ease of exposition, the source language for the translation task is referred to as "French", and the target language is referred to as "English", although these can be any language pairs in practice. The translation problem is defined as given a French string $f$, find the English string $\hat{e}$, and is presented in Equation 3.1.

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, Pr(e|f) = \underset{e}{\operatorname{argmax}}\, Pr(e) * Pr(f|e) \tag{3.1}$$

The variable $e$ represents any potential English string made up of English words. $Pr(e)$ represents the true distribution over English strings. $Pr(f|e)$ represents the true distribution over French strings generated from English strings.

Consider $P_\theta(f|e)$ to be a model of $Pr(f|e)$. If we introduce a hidden variable $a$ representing word alignments, we can sum over all possible alignments, as in Equation 3.2.

$$P_\theta(f|e) = \sum_a P_\theta(f, a|e) \tag{3.2}$$

For our task, which is word alignment annotation, we have fixed strings $f$ and $e$, and we wish to select the best alignment according to the model, $\hat{a}$, which we do in Equation 3.3. This alignment is called the Viterbi alignment.

$$\hat{a} = \underset{a}{\operatorname{argmax}}\, P_\theta(a|e, f) = \underset{a}{\operatorname{argmax}}\, P_\theta(f, a|e) \tag{3.3}$$

We will subsequently drop the $\theta$ subscript when calculating probabilities according to the model. Note that generative word alignment models often model the probability of stochastically generating the French string from the English string. This is the reverse direction of the translation task, and is motivated by the noisy channel formulation

which is the right-most term in Equation 3.1. For this reason we will refer to English as the "source" language and French as the "target" language subsequently in this chapter, as is standardly done in the word alignment literature.

## 3.2 LEAF: A Generative Word Alignment Model

### 3.2.1 Generative Story

We introduce a new generative story which enables the learning of non-consecutive M-to-N alignment structure. We use the same notation as the generative story for Model 4 (Brown et al., 1993), which we are extending, where this is possible. The reader may find it useful to consult Appendix A for a discussion of Model 4.

The LEAF generative story describes the stochastic generation of a target string $f$ (sometimes referred to as the French string, or foreign string) from a source string $e$ (sometimes referred to as the English string), consisting of $l$ words. The variable $m$ is the length of $f$. We generally use the index $i$ to refer to source words ($e_i$ is the English word at position $i$), and $j$ to refer to target words.

Our generative story makes the distinction between different types of source words. There are head words, non-head words, and deleted words. Similarly, for target words, there are head words, non-head words, and spurious words. A head word is associated with zero or more non-head words; each non-head word is associated with exactly one

head word. The purpose of head words is to try to provide a robust representation of the semantic features necessary to determine translational correspondence. This is similar to the use of syntactic head words in statistical parsers to provide a robust representation of the syntactic features of a parse sub-tree. However, an important difference is that in current training approaches the head words are not determined using supervision (annotated training data) or hand-written rules, but instead estimated in an unsupervised fashion.

A minimal translational correspondence consists of a linkage between a source head word and a target head word (and by implication, the non-head words which they are associated with). Each head word is involved in exactly one such link. Deleted source words are not involved in a minimal translational correspondence, as they were "deleted" by the translation process. Spurious target words are also not involved in a minimal translational correspondence, as they spontaneously appeared during the generation of other target words.

Figure 3.1 shows a simple example of the stochastic generation of a French sentence from an English sentence, annotated with the step number in the generative story, which we present next.

In specifying the generative story we will introduce some new notation. We use the three word classes $\text{class}_e$, $\text{class}_f$, and $\text{class}_h$ to reduce the dimensionality of the English vocabulary, the French vocabulary and the French head word vocabulary respectively.

To define the distortion model we use two notational tools: $\rho_i$ is the previous English head word to the English head word at $i$; and $c_z$ is the "center" of the French cept, the average of the positions of the words in the cept, whose head word is linked with the English head word at position $z$.

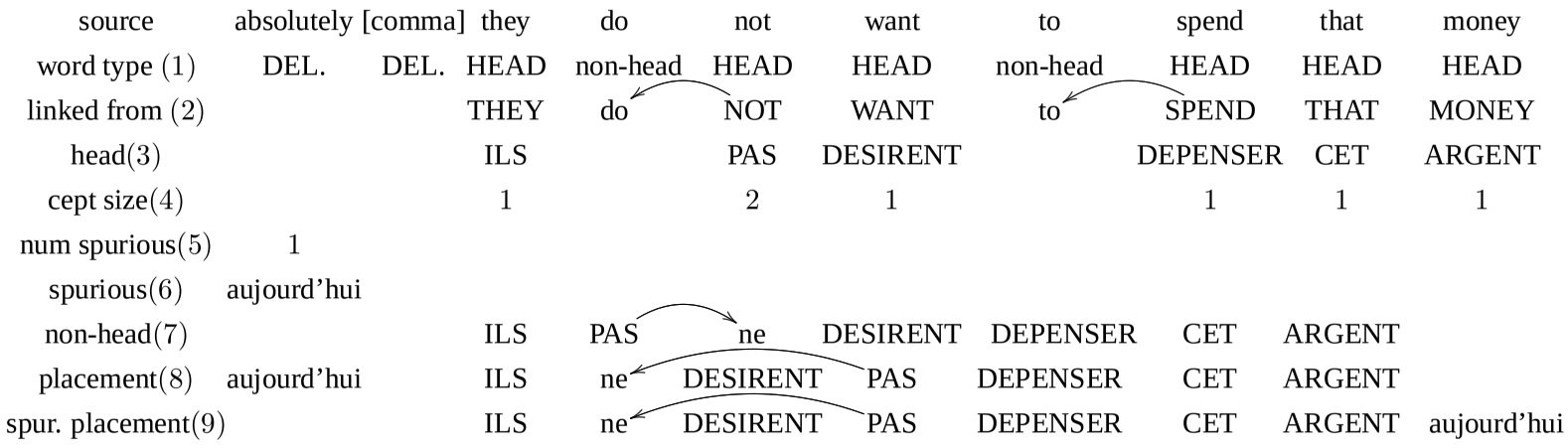


Figure 3.1: Generative story example, (number) indicates step number

1. Choose the source word type.

   for each $i = 1, 2, ..., l$ choose a word type $\chi_i = -1$ (non-head word), $\chi_i = 0$ (deleted word) or $\chi_i = 1$ (head word) according to the distribution $g(\chi_i|e_i)$

   let $\chi_0 = 1$

2. Choose for each non-head word the identity of the head word it is associated with

   for each $i = 1, 2, ..., l$ if $\chi_i = -1$ choose the position of the associated head word $\mu_i$ for the non-head word $e_i$ according to the distribution $w_{-1}(\mu_i - i|\text{class}_e(e_i))$

   for each $i = 1, 2, ..., l$ if $\chi_i = 1$ let $\mu_i = i$

   for each $i = 1, 2, ..., l$ if $\chi_i = 0$ let $\mu_i = 0$

   * for each $i = 1, 2, ..., l$ if $\chi_{\mu_i} \neq 1$ return "failure"

3. Choose the identity of the generated target head word for each source head word

   for each $i = 1, 2, ..., l$ if $\chi_i = 1$ choose $\tau_{i1}$ according to the distribution $t_1(\tau_{i1}|e_i)$

4. Choose the number of words in each target cept. This is conditioned on the identity of the source head word from which the target head word was generated and the source cept size ($\gamma_i$ is 1 if the cept size is 1, and 2 if the cept size is greater than 1)

   for each $i = 1, 2, ..., l$ if $\chi_i = 1$ choose a target cept size $\psi_i$ according to the distribution $s(\psi_i|e_i, \gamma_i)$

for each $i = 1, 2, ..., l$ if $\chi_i < 1$ let $\psi_i = 0$

5. Choose the number of spurious words.

   choose $\psi_0$ according to the distribution $s_0(\psi_0 | \sum_i \psi_i)$

   let $m = \psi_0 + \sum_{i=1}^{l} \psi_i$

6. Choose the identity of the spurious words.

   for each $k = 1, 2, ..., \psi_0$ choose $\tau_{0k}$ according to the distribution $t_0(\tau_{0k})$

7. Choose the identity of the target non-head words associated with each target head word.

   for each $i = 1, 2, ..., l$ and for each $k = 2, 3, ..., \psi_i$ choose $\tau_{ik}$ according to the distribution $t_{>1}(\tau_{ik} | e_i, \text{class}_h(\tau_{i1}))$

8. Choose the position of the target head and non-head words.

   for each $i = 1, 2, ..., l$ and for each $k = 1, 2, ..., \psi_i$ choose a position $\pi_{ik}$ as follows:

   - if $k = 1$ choose $\pi_{i1}$ according to the distribution $d_1(\pi_{i1} - c_{\rho_i} | \text{class}_e(e_{\rho_i}), \text{class}_f(\tau_{i1}))$
   - if $k = 2$ choose $\pi_{i2}$ according to the distribution $d_2(\pi_{i2} - \pi_{i1} | \text{class}_f(\tau_{i1}))$
   - if $k > 2$ choose $\pi_{ik}$ according to the distribution $d_{>2}(\pi_{ik} - \pi_{ik-1} | \text{class}_f(\tau_{i1}))$

   * if any position was chosen twice, return “failure”

9. Choose the position of the spuriously generated words.

   for each $k = 1, 2, ..., \psi_0$ choose a position $\pi_{0k}$ from $\psi_0 - k + 1$ remaining vacant positions in $1, 2, ..., m$ according to the uniform distribution

   let $f$ be the string $f\pi_{ik} = \tau_{ik}$

We note that the steps which return "failure" (the two steps marked with a "*" in the generative story) are required because the model is deficient. Deficiency means that a portion of the probability mass in the model is allocated towards generative stories which would result in infeasible alignment structures. Our model has deficiency in the non-spurious target word placement, just as Model 4 does. It has additional deficiency in the source word linking decisions. Och and Ney (2003) presented results suggesting that the additional parameters required to ensure that a model is not deficient result in inferior performance, but we plan to study whether this is the case for our generative model in future work.

### 3.2.2 Mathematical Formulation

Given $e$, $f$ and a candidate alignment $a$, which represents both the links between source and target head words and the head word connections of the non-head words, we would like to calculate $P(f, a|e)$. The formula for this is:

$$
\begin{aligned}
P(f,a|e) = & [\prod_{i=1}^{l} g(\chi_i|e_i)] \\
& [\prod_{i=1}^{l} \delta(\chi_i,-1) w_{-1}(\mu_i - i|\text{class}_e(e_i))] \\
& [\prod_{i=1}^{l} \delta(\chi_i,1) t_1(\tau_{i1}|e_i)] \\
& [\prod_{i=1}^{l} \delta(\chi_i,1) s(\psi_i|e_i,\gamma_i)] \\
& [s_0(\psi_0|\sum_{i=1}^{l} \psi_i)] \\
& [\prod_{k=1}^{\psi_0} t_0(\tau_{0k})] \\
& [\prod_{i=1}^{l} \prod_{k=2}^{\psi_i} t_{>1}(\tau_{ik}|e_i,\text{class}_h(\tau_{i1}))] \\
& [\prod_{i=1}^{l} \prod_{k=1}^{\psi_i} D_{ik}(\pi_{ik})]
\end{aligned}
$$

where:

$\delta(i,i')$ is the Kronecker delta function which is equal to 1 if $i = i'$ and 0 otherwise.

$\rho_i$ is the position of the closest English head word to the left of the word at $i$ or 0 if there is no such word.

$\text{class}_e(e_i)$ is the word class of the English word at position $i$, $\text{class}_f(f_j)$ is the word class of the French word at position $j$, $\text{class}_h(f_j)$ is the word class of the French head word at position $j$.

$p_0$ and $p_1$ are parameters describing the probability of not generating and of generating a single target spurious word from each non-spurious target word, $p_0 + p_1 = 1$.

$$m' = \sum_{i=1}^{l} \psi_i \tag{3.4}$$

$$s_0(\psi_0|m') = \binom{m'}{\psi_0} p_0^{m'-\psi_0} p_1^{\psi_0} \tag{3.5}$$

$$D_{ik}(j) = \begin{cases} d_1(j - c_{\rho_i}|\text{class}_e(e_{\rho_i}), \text{class}_f(\tau_{ik})) \\ \qquad \text{if } k = 1 \\ d_2(j - \pi_{i1}|\text{class}_f(\tau_{ik})) \\ \qquad \text{if } k = 2 \\ d_{>2}(j - \pi_{ik-1}|\text{class}_f(\tau_{ik})) \\ \qquad \text{if } k > 2 \end{cases} \tag{3.6}$$

$$\gamma_i = \min(2, \sum_{i'=1}^{l} \delta(\mu_{i'}, i)) \tag{3.7}$$

$$c_i = \begin{cases} \text{ceiling}(\sum_{k=1}^{\psi_i} \pi_{ik}/\psi_i) & \text{if } \psi_i \neq 0 \\ 0 & \text{if } \psi_i = 0 \end{cases} \tag{3.8}$$

### 3.2.3 Other Alignment Structures are Special Cases

The alignment structure used in many other approaches can be modeled using special cases of this framework. We can express the 1-to-N structure of models like Model 4 by disallowing $\chi_i = -1$. For 1-to-1 structure we both disallow $\chi_i = -1$ and deterministically set $\psi_i = \chi_i$. We can also specialize our generative story to the consecutive word M-to-N alignments used in "phrase-based" models, though in this case the conditioning of the generation decisions would be quite different. This involves adding checks on source and target connection geometry to the generative story. These checks would check whether the phrase-based constraint is violated. If it is violated, "failure" would be returned. Naturally this would be at the cost of additional deficiency.

### 3.2.4 Symmetricity

The LEAF generative story is symmetric, and so the same alignment structure can be used to evaluate the model in the French to English, or in the English to French direction. In practice, we will estimate the model in both directions, and in unsupervised training we will maximize likelihood in both directions. When determining the Viterbi

alignment, we sum the log costs of the model in both directions. We discuss unsupervised training in the next section.

## 3.3 Unsupervised Training

### 3.3.1 Training LEAF Using Expectation-Maximization

#### 3.3.1.1 Introduction

In this section we present the training of LEAF using the Expectation-Maximization algorithm. Expectation-Maximization (Dempster et al., 1977), or EM, is an algorithm for finding parameter settings of a model which maximize the expected likelihood of the observed and the unobserved data (this is called the complete data likelihood; the incomplete data likelihood is the likelihood of only the observed data). Intuitively, in statistical word alignment, the E-step corresponds to calculating the probability of all alignments according to the current model estimate, while the M-step is the creation of a new model estimate given a probability distribution over alignments (which was calculated in the E-step).

#### 3.3.1.2 E-step

In the E-step we would ideally like to enumerate all possible alignments and label them with $P(f, a|e)$. However, this is not possible when using a word alignment model as

complex as LEAF. As we will see below in the discussion of the M-step, we would at least like to find the most likely alignment of a pair $e$ and $f$ given the model. This is the Viterbi alignment, $\hat{a}$ in this formula:

$$\hat{a} = \operatorname*{argmax}_a P_\theta(a|e,f) = \operatorname*{argmax}_a P_\theta(f,a|e) \tag{3.9}$$

This is a repeat of equation 3.3 which represents the task of finding an approximate Viterbi alignment to output as the final alignment output from the alignment process. Here, in Equation 3.9 we are referring to the search for an alignment during training. We can vary this to be, for instance, the search for the 10 most probable alignments (where a posterior distribution over the 10 alignments would be used for updating the model in the M-step).

Unfortunately, there is no known polynomial time algorithm for finding the Viterbi alignment of LEAF, or even for determining that a particular alignment is the Viterbi alignment. We assume that this is intractable. A similar problem (the calculation of the Viterbi alignment for IBM Model 4) was proven to be NP-hard by Udupa and Maji (2006). So we take the most probable alignment we can find, and assume it is the Viterbi alignment. The algorithms used to solve this search problem are discussed in Section 3.4.

#### 3.3.1.3 M-step

For the M-step, we would like to take a sum over all possible alignments for each sentence pair, weighted by $P(a|e,f)$ which we calculated in the E-step (note that the alignments labeled with probabilities in the E-step must be renormalized to sum to 1 for each $e, f$ pair, as they are estimates of $P(f,a|e)$, and we would like estimates of $P(a|e,f)$). As we mentioned, this is not tractable.

We make the assumption that the Viterbi alignment can be used to update our estimate in the M-step (which we call $p_M(a|e,f)$, the probability of the alignment given the sentence $e$ and the sentence $f$):

$$p_M(a|e,f) = \begin{cases} 1 & \text{if } a = \hat{a} \\ 0 & \text{if } a \neq \hat{a} \end{cases} \tag{3.10}$$

Note that we are abusing the term "Viterbi alignment" to mean the best alignment according to the model that we can find, not the best alignment according to the model that exists.

Although in our experiments we use Viterbi training, neighborhood estimation (Al-Onaizan et al., 1999; Och & Ney, 2003) , "pegging" (Brown et al., 1993) or some other means of creating a set of candidate alignments (whose probabilities are then normalized to sum to one) could be used instead in the M-step.

| | |
|---|---|
| $c_g(\chi_i \mid e_i)$ | source word type |
| $c_\mu(\triangle i \mid \text{class}_e(e_i))$ | head word links (collected if $\chi_i = -1$) |
| $c_{t_1}(f_j \mid e_i)$ | head word translation |
| $c_s(\psi_i \mid e_i, \gamma_i)$ | number of words in target cept |
| $c_{s_0}(\psi_0 \mid \sum_i \psi_i)$ | number of unaligned target words |
| $c_{t_0}(f_j)$ | identity of unaligned target words |
| $c_{t_{>1}}(f_j \mid e_i, \text{class}_h(\tau_{i1}))$ | non-head word translation |
| $c_{d_1}(\triangle j \mid \text{class}_e(e_\rho), \text{class}_f(f_j))$ | movement of target head words |
| $c_{d_2}(\triangle j \mid \text{class}_f(f_j))$ | movement of left-most target non-head word |
| $c_{d_{>2}}(\triangle j \mid \text{class}_f(f_j))$ | movement of subsequent target non-head words |
| (same counts, other direction) | ... |

Table 3.1: Counts used in unsupervised training of LEAF

We estimate new parameters from the Viterbi alignments found during the E-step by simply counting events in the Viterbi alignments, since they are assumed in equation 3.10 to be the only alignments of non-zero probability. We are interested in the counts in Table 3.1 which we simply count in $a$. After collecting the counts, for each condition, we normalize these counts so that the conditional probabilities sum to one, which provides us with the model estimate which is the result of the M-step.

The Viterbi training approximation is related to EM training, which tries to maximize the complete data log likelihood. Neal and Hinton (1998) analyze approximate EM training and motivate this general variant. In future work we would like to try using a probability estimate over a larger set of hypothesized alignments to reestimate the

model, but finding a set to use which helps performance of the estimated models is an open research problem.

### 3.3.2 Bootstrapping

The term "bootstrapping" refers to how we initialize the model. In order to perform unsupervised training of our new model we require an initial probability distribution over alignments. In practice, instantiations of the EM algorithm (including approximate variants) start with a pseudo-M step, where we estimate an initial "iteration 0" model estimate, before the first full iteration of EM. For example, the IBM Models (Brown et al., 1993) were originally specified as a sequence of increasingly complex models which bootstrap from one another in this fashion. The iteration 0 estimate is calculated using the counts necessary for our current model. However, these counts are collected over the alignment distribution (the set of alignments and their probabilities) estimated using the previous model in the bootstrapping chain. In our work, we use Model 1 to start with, bootstrap the HMM Model (Vogel et al., 1996) from Model 1, and then bootstrap LEAF from the HMM Model.

To initialize the parameters of the generative model for the first iteration, we use bootstrapping from a 1-to-N and a M-to-1 alignment. We use the intersection of the 1-to-N and M-to-1 alignments to provide likely candidates for the head word relationship,

the 1-to-N alignment to delineate likely target word cepts, and the M-to-1 alignment to delineate likely source word cepts.

A key concept in our bootstrapping algorithms is whether an initial alignment is feasible under the new model or not. Feasible means that we could set the parameter settings for the model such that this alignment will have probability greater than zero under the model. Infeasible means that no such parameter settings exist.

A problem arises when we encounter infeasible alignment structure where, for instance, a source word generates target words but no link between the target words and the source word appears in the intersection, so it is not clear which target word is the target head word. To address this, we consider each of the N generated target words as the target head word in turn and assign this configuration 1/N of the counts.

## 3.4 Search

For each iteration of training we search for the Viterbi alignment for millions of sentence pairs. Evidence that inference over the space of all possible alignments is intractable has been presented, for a similar problem, by Udupa and Maji (2006). Left-to-right hypothesis extension using a beam decoder (as is typically implemented in phrase-based SMT decoders) is unlikely to be effective because in word alignment reordering can not be limited to a small local window and so the necessary beam would be very large. We are not aware of admissible or inadmissible search heuristics which

have been shown to be effective when used in conjunction with a search algorithm similar to A* search for a model predicting over a structure like ours. Therefore we use a simple local search algorithm which operates on complete hypotheses.

Brown et al. (1993) defined two local search operations for their 1-to-N alignment models 3, 4 and 5. All alignments which are reachable via these operations from the starting alignment are considered. One operation is to change the generation decision for a French word to a different English word (move), and the other is to swap the generation decision for two French words (swap). All possible operations are tried and the best is chosen. This is repeated. The search is terminated when no operation results in an improvement. Och and Ney (2003) discussed efficient implementation.

In our model, because the alignment structure is richer, we define the following operations:

- move French non-head word to new head
- move English non-head word to new head
- swap heads of two French non-head words
- swap heads of two English non-head words
- swap English head word links of two French head words
- link English word to French word making new head words

- unlink English and French head words

These operations are defined and discussed further in the next section. Germann et al. (2004) and Marcu and Wong (2002) introduce some similar operations without the head word distinction.

### 3.4.1 Implementing the Search Operations

We now define the seven operations which transform an alignment $a$ to an alignment $a'$. For each operation we begin by copying $a$ to $a'$ and then apply the operation on $a'$ as specified. The four operations which are applied to non-head words are in Figure 3.2 and the three operations applied to head word links are in Figures 3.3 and 3.4. Note that the operations applied to non-head words are similar to the word level operations in Model 4. The operations applied to head-word links are like the operations in phrase-based alignment such as those defined by Marcu and Wong (2002).

In implementing the search algorithm, we represent an alignment $a$ as a vector $\mu$, a vector $b$ and a vector $h$. $b_j$ is used to indicate the target head word for the target word at position $j$, just as $\mu_i$ indicates the source head word for the source word at position $i$. $h_j$ indicates which source head word at position $i$ generated the target head word at position $j$. $h_j = 0$ if the word at position $j$ is not a head word. If the source word at position $i$ is deleted we set $\mu_i = 0$. Likewise, if the target word at position $j$ is spurious,

we set $b_j = 0$. We also define the function inv$(i)$ which returns the position $j$ for which $h_j = i$ or returns $0$ if there is no such position.

For comparison we note that for 1-to-N models an alignment $a$ is often represented as a vector $v$ where $v_j$ indicates the position of the source word which generated the target word at position $j$, and $v_j = 0$ if the target word is spuriously generated.

We try all possible values of the parameters (see the line "Given" in each operation). Note that "unlink source and target head words", Operation 7 in Figure 3.4, has 3 parameters, rather than 2. To control complexity we restrict the total number of modified alignments considered reachable from an alignment $a$ by applying this operation. This is done by placing restrictions on the parameters $i$ and $j$, which specify the location of the head-words with which to associate the former head words (and non-head words previously associated with these former head words). We only allow for association with nearby head words, or for changing the type of affected source words to "deleted" source word, or affected target words to "spurious" target word.

### 3.4.2 Search Algorithms

Any search algorithm trying to find the Viterbi alignment according to the LEAF model is trying to solve a problem which is most likely intractable. We must align as many as 10,000,000 sentence pairs for a single iteration of training (given the data sets we have at the present time).

**OPERATION 1: move French non–head word**
**Given**: target word positions $j$, $j'$
**if** $b_j \neq j$ and $b_{j'} = j'$ **then**
  let $b_j = j'$
**end if**

**OPERATION 2: move English non–head word**
**Given**: source word positions $i$, $i'$
**if** $\mu_i \neq i$ and $\mu_{i'} = i'$ **then**
  let $\mu_i = i'$
**end if**

**OPERATION 3: swap French head word decisions of two French non–head words**
**Given**: target word positions $j$, $j'$
**if** $b_j \neq j$ and $b_{j'} \neq j'$ **then**
  swap $b_j$ and $b_{j'}$
**end if**

**OPERATION 4: swap English head word decisions of two English non–head words**
**Given**: source word positions $i$, $i'$
**if** $\mu_i \neq i$ and $\mu_{i'} \neq i'$ **then**
  swap $\mu_i$ and $\mu_{i'}$
**end if**

Figure 3.2: LEAF search operations: move and swap non-head words

**OPERATION 5: swap English head word links of two French head words**
**Given**: target word positions $j$, $j'$
**if** $b_j = j$ and $b_{j'} = j'$ **then**
  swap $h_j$ and $h_{j'}$
**end if**

**OPERATION 6: link English word to French word**
{after this operation is performed, source word $i$ and target word $j$ are both head words}
**Given**: source word position $i$, target word position $j$
let $j' = \text{inv}(i)$, let $i' = h_j$
**if** $i' \neq 0$ **then**
  **for** $i'' = 1..l$ **do**
    **if** $\mu_{i''} = i'$ **then**
      let $\mu_{i''} = i$
    **end if**
  **end for**
**end if**
**if** $j' \neq 0$ **then**
  **for** $j'' = 1..m$ **do**
    **if** $b_{j''} = j'$ **then**
      let $b_{j''} = j$
    **end if**
  **end for**
**end if**
let $h_j = i$

Figure 3.3: LEAF search operations: swap and link head words

**OPERATION 7: unlink the link between an English head word and a French head word**
{non-head words whose head words are French $j'$ or English $h_{j'}$ would be "orphaned"}
{parameter $i$ is the English head word (or NULL) to which to attach the English head-word at $h_{j'}$ and any non-head words attached to $h_{j'}$}
{parameter $j$ is the French head word (or NULL) to which to attach the French head-word at $j'$ and any non-head words attached to $j'$)}
**Given**: target word position $j'$, source word position $i$, target word position $j$
let $i' = h_{j'}$
**if** $i' \neq 0$ and $\mu_i = i$ and $b_j = j$ **then**
  let $\mu_{i'} = i$ and $b_{j'} = j$ and $h_{j'} = 0$
  **for** $i'' = 1..l$ **do**
    **if** $\mu_{i''} = i'$ **then**
      let $\mu_{i''} = i$
    **end if**
  **end for**
  **for** $j'' = 1..m$ **do**
    **if** $b_{j''} = j'$ **then**
      let $b_{j''} = j$
    **end if**
  **end for**
**end if**

Figure 3.4: LEAF search operation: unlink head words

To control memory usage, which would be a problem with any search algorithm, we have developed a technique where we restrict the memory used to the parameters we need for a small number of parallel sentences at a cost of refiltering the parameters each time we load a small group of parallel sentences to align.

Because of the time tractability issues, we use a hillclimbing local search. Local search does have one advantage over search algorithms which rely on hypothesis extension, which is that we are always operating on a complete hypothesis. This makes integration of new knowledge sources easier, and in particular allows for knowledge sources which can only be scored over a complete hypothesis, which would be difficult to use if our search involved partial hypothesis extension.

#### 3.4.2.1 Basic Search Algorithm

In the basic search algorithm, we start the search from a starting alignment (for which we use the best alignment from the previous iteration) and exhaustively try each of the operations in Figures 3.2, 3.3 and 3.4 with all possible values for the parameters. We remember which resulting hypothesis was the best, to use as the starting point in the next iteration of search. We terminate the search when no improvement in model score via the search operations in Figures 3.2, 3.3 and 3.4 is possible.

#### 3.4.2.2 New Alignment Search Algorithm

We developed a new alignment algorithm to reduce the numerous search errors[1] made by the basic search algorithm and directly control the time taken:

[1] A search error in a word aligner is a failure to find the best alignment according to the model, i.e. in our case a failure to maximize Equation 3.3.

- The alignment search operates by considering complete hypotheses so it is an "anytime" algorithm (meaning that it always has a current best guess). Timers can therefore be used to control processing, and we set these based on the product of the source and target sentence lengths.

- Alignments which are selected as the starting point at any iteration during a single run of the search algorithm are marked so that they can not be returned to at a future point in the same search run.

- We perform a hillclimbing search (as in the baseline algorithm) but as we search we construct a priority queue of possible other candidate alignments to consider (i.e. the second, third, etc best alignments seen). The search is restarted by drawing the best candidate from this queue after a timer expires. When calculating Viterbi alignments for the entire training corpus we have found it effective to set such a timer 5 or more times, increasing the time limit each time.

The first improvement is important for restricting total time used when producing alignments for large training corpora. The latter two improvements are related to the well-known Tabu local search algorithm (Glover, 1986).

#### 3.4.2.3 Comparing the Two Search Algorithms

One issue of major importance in using local search is the careful control of search errors. A search error is a failure to find the Viterbi alignment under the current model estimate and in a basic hillclimbing search it means that the search ended in a local probability maxima[2].

We present an experiment comparing our two search algorithms for the Model 4 search task. We apply it a French/English task and to an Arabic/English task. The directions evaluated are the French to English and Arabic to English generational directions. We apply both algorithms using the Model 4 search operations described in Appendix A. For each corpus we sampled 1000 sentence pairs randomly, with no sentence length restriction. Model 4 parameters are estimated from the final HMM Viterbi alignment of these sentence pairs. We then search to try to find the Model 4 Viterbi alignment with both the new and old algorithms, allowing them both to process for the same amount of time.

Our experiment evaluates the number of search errors made using the baseline search algorithm and the new search algorithm. The percentage of known search errors is the percentage of sentences from our sample in which we were able to find a more probable candidate by applying our new algorithm using 24 hours of computation

[2]A search error could also mean that we had an error in the implementation of our search algorithm, but we are confident that over the course of experimentation we have effectively removed such errors.

| System | Known Search Errors % |
|---|---|
| Arabic/English Old | 19.4 |
| Arabic/English New | 8.5 |
| French/English Old | 32.5 |
| French/English New | 13.7 |

Table 3.2: Comparison of New Search Algorithm with Old Search Algorithm for Model 4 Alignment

for just the 1000 sample sentences. Table 3.2 presents the results, showing that our new algorithm reduced known search errors to $8.5\%$ for Arabic to English and $13.7\%$ for French to English. This shows that the new algorithm is more effective than the baseline search algorithm.

## 3.5 Experiments

### 3.5.1 Data Sets

We perform experiments on two large alignments tasks, for Arabic/English and French/English data sets. Statistics for these sets are shown in Table 3.3. All of the data used is available from the Linguistic Data Consortium except for the French/English gold standard alignments which we annotated ourselves (and are available from us).

| | | Arabic/English | | French/English | |
|---|---|---|---|---|---|
| | | A | E | F | E |
| Training | Sents | 6,609,162 | | 2,842,184 | |
| | Words | 147,165,003 | 168,301,299 | 75,794,254 | 67,366,819 |
| | Vocab | 642,518 | 352,357 | 149,568 | 114,907 |
| | Singletons | 256,778 | 158,544 | 60,651 | 47,765 |
| Align Discr. | Sents | 1,000 | | 110 | |
| | Words | 26,882 | 37,635 | 1,888 | 1,726 |
| | Links | 39,931 | | 2,292 | |
| Align Test | Sents | 83 | | 110 | |
| | Words | 1,510 | 2,030 | 1,899 | 1,716 |
| | Links | 2,131 | | 2,176 | |
| Trans. Dev | Sents | 728 (4 references) | | 833 (1 reference) | |
| | Words | 18,255 | 22.0K to 24.6K | 20,562 | 17,454 |
| Trans. Test | Sents | 1,056 (4 references) | | 2,380 (1 reference) | |
| | Words | 28,505 | 35.8K to 38.1K | 58,990 | 49,182 |

Table 3.3: Data sets

### 3.5.2 Experimental Results

To build both our baseline and the contrastive alignment systems, we start with 5 iterations of Model 1 followed by 4 iterations of HMM (Vogel et al., 1996), as implemented in GIZA++ (Och & Ney, 2003).

For the LEAF word classes, we use the same set of classes as the baseline system. 50 classes are used for each language. The classes are determined using the "mkcls" program which is supplied with GIZA++. This program starts with a random assignment of the words in a monolingual text to the 50 monolingual classes and then greedily maximizes the likelihood of the monolingual text according to a class-based bigram model by moving words to different classes as described by Och (1999). In our

experiments the classes used for the head classes, $\text{classes}_h$, are the same as those used for all French words, $\text{classes}_f$.

For non-LEAF systems, we take the best performing of the "Union", "Refined" and "Intersection" symmetrization heuristics (Och & Ney, 2003) to combine the 1-to-N and M-to-1 directions resulting in a M-to-N alignment. Because these systems do not output fully linked alignments, we fully link the resulting alignments. The reader should recall that this does not change the set of rules or phrases that can be extracted using the alignment.

We compare the unsupervised LEAF system with GIZA++ Model 4 to give some idea of the performance of the unsupervised model. We made an effort to optimize the free parameters of GIZA++, while for unsupervised LEAF there are no free parameters to optimize. A single iteration of unsupervised LEAF is compared with heuristic symmetrization of GIZA++'s extension of Model 4 (which was run for four iterations). LEAF was bootstrapped as described in Section 3.3.2 from the HMM Viterbi alignments. Note that the timings for the first E-Step of the French/English experiments are presented in Appendix C.1. The current (unoptimized) LEAF search implementation is slow; speeding up search is discussed in the same appendix.

Results for the experiments on the French/English data set are shown in Table 3.4. We ran GIZA++ for four iterations of Model 4 and used the "Refined" heuristic (line 1). We observe that LEAF unsupervised (line 2) is competitive with GIZA++ (line 1).

| System | French/English F-Measure ($\alpha = 0.4$) | Arabic/English F-measure ($\alpha = 0.1$) |
|---|---|---|
| GIZA++ | 73.5 | 75.8 |
| LEAF unsupervised | 74.5 | 72.3 |

Table 3.4: Experimental Results

Results for the Arabic/English data set are also shown in Table 3.4. We used a large gold standard word alignment set available from the LDC. We ran GIZA++ for four iterations of Model 4 and used the "Union" heuristic. We compare GIZA++ (line 1) with one iteration of the unsupervised LEAF model (line 2). The unsupervised LEAF system is worse than four iterations of GIZA++ Model 4. We believe that the features in LEAF are too high dimensional to use for the Arabic/English task (which is more difficult than the French/English task) without the back-offs available in the semi-supervised model which we will discuss in Chapter 4.

We will return to these experiments in Chapter 4 to compare the performance of our unsupervised systems with the semi-supervised systems presented there. In particular, we will present a discriminative model based on sub-models directly derived from the LEAF generative story which we will train using a semi-supervised training algorithm.

## 3.6 Previous Work

The LEAF model is inspired by the literature on generative modeling for statistical word alignment and particularly by IBM Model 4 (Brown et al., 1993). Because of this, we begin our discussion of previous work in generative modeling with the most widely used alignment structure, the 1-to-N structure, which is that used by the IBM Models and the HMM word alignment model. We then continue with other structures, discuss additional issues and conclude.

### 3.6.1 Generative Models of 1-to-N Structure

The 1-to-N structure is not the best alignment structure. See the discussion in Section 1.2.4 and particularly Table 1.6 on Page 10 for an analysis of two example parallel sentences which shows that there are interesting minimal translational correspondences which can not be modeled using this structure.

Most 1-to-N models have the advantage that their parameters can be robustly estimated from relatively small amounts of data. While such models can not directly account for M-to-N discontinuous correspondence, they can use word deletion, where a source word generates nothing (sometimes referred to as “zero fertility” for reasons which will become apparent in the discussion), to try to reduce the effect of this by allowing all of the source words which should appear in a M-to-N relationship to be

deleted except for one source word which generates the N target words[3]. Often models with this structure do a good job of accounting for the cepts in the target language, by robustly decomposing the probabilities associated with these cepts into word level probabilities, and in practice these models can even deal with discontinuous target cepts well. Given decisions about target cepts taken from a 1-to-N alignment, and source cepts taken from a N-to-1 alignment, heuristics can be applied which attempt to generate a M-to-N discontinuous alignment of reasonable quality.

In practice, the main disadvantage of this alignment structure is the need for heuristic symmetrization in order to obtain M-to-N discontinuous alignments. Heuristic symmetrization was introduced by Och and Ney (2003) and extended by Koehn et al. (2003). The choice of symmetrization heuristic which is most effective changes from task to task. It is not only dependent on the language pair being aligned, as well as the translation direction of the final translation task, but it is also dependent on the training data size (for instance, see the graphs in Chapter 2 on page 36). Appendix A contains further information on heuristic symmetrization, including specific details of how it is used in our baseline. LEAF does not require use of these heuristics.

We now discuss specific 1-to-N alignment models, beginning with the IBM models.

[3]However, In general many variants approximating an M-to-N minimal translational correspondence will be possible. For instance if M=N such a model will often align the words 1-to-1. But it is important to remember that none of these variants is correct and it is easy to find contexts where the translation rules licensed by such variants would be harmful.

#### 3.6.1.1 The IBM Models

Brown et al. (1993) developed five statistical models of translation, IBM Models 1 through 5, and parameter estimation techniques for them. These models all use the 1-to-N alignment structure. The models were designed to be used in a pipeline, where each model is bootstrapped from the previous model.

Model 1 is the first model in the pipeline. It makes very strong conditional independence assumptions on word placement and generation (all French words are generated and placed independently). Three probability distributions are involved in generating a French sentence from a English sentence using steps which define an alignment. These are a distribution over the length of the French sentence, a distribution over the alignment decision for each French word position (denoting the position of the English word which generated it), and a distribution over the translation decision (which stochastically selects the lexical identity of the French word given the English word which generated it).

The formula in Model 1 for the joint probability of an alignment and a French string, given an English string, is in Equation 3.11. Note the three components of the model. The length distribution is the numerator of the term before the product. The alignment position model is simply $1/(l+1)^m$, a uniform distribution over the English positions (including position $0$ which if selected would indicate that the French word

is spuriously generated). The translation model is inside the product so it is evaluated once for each of the $m$ French words.

$$P(f, a|e) = \frac{p(m|l)}{(l+1)^m} \prod_{j=1}^{m} p(f_j|e_{a_j}) \tag{3.11}$$

When Model 1 is trained to maximize likelihood using EM the likelihood is convex, but in practice Och and Ney (2003) suggest that stopping before convergence increases performance. The estimation of the parameters for a single iteration can solved without a complex search operation, and the calculation of the Viterbi alignment for a fixed $e$ and $f$ is trivial (the highest generation probability for each French word is selected). This makes Model 1 a popular choice for applications which do not require a strong model of translational correspondence but instead a rough indication of whether two sentences should be considered parallel, such as sentence alignment (Moore, 2002). Model 1 is also used as a smoothing method for higher order translation models (Och et al., 2004).

Model 2 relaxes one of the assumptions of Model 1, by making the location of the English word which generated each French word dependent on the absolute locations of the two words. The equation for Model 2 is in Equation 3.12. The first term is again the length distribution. Within the product, the first term is the alignment position model (the conditional probability that the French word at position $j$ is generated by the English word at position $a_j$). The translation model is identical with Model 1. Like Model

1, the estimation of the parameters for a single iteration can solved without a complex search operation, and the calculation of the Viterbi alignment is also simple (the product of the alignment position model and the translation model is simply maximized for each French word in turn).

$$P(f, a|e) = p(m|l) \prod_{j=1}^{m} p(a_j|j, l, m) p(f_j|e_{a_j}) \tag{3.12}$$

Models 1 and 2 are both weak models of translational correspondence which were designed to be used for bootstrapping Models 3, 4 and 5. The advantage of these models is the tractability of both estimating the models and making predictions using the models.

Models 3, 4 and 5 are considerably more complex. These models are discussed in detail in Appendix A. They are referred to as the "fertility" models. An English word's fertility is the number of French words generated by it. The use of a fertility model requires inverting the alignment position model. Models 3 and 4 use a simple alignment position model which introduces deficiency into the estimation. Deficiency means that the model wastes probability mass on predictions which are impossible. In this case the deficiency lies in the placement decisions for French words (an example is that the probability that two French words are placed in the same position can be non-zero). Och and Ney (2003) presented evidence that this form of deficiency is not a problem in practice.

Model 3 introduces the "fertility" distribution. The alignment model still uses absolute positions as in IBM Model 2, but is inverted so that we calculate the probability of placing a French word given an English word's position (rather than vice versa, as was the case for Model 2). Model 3 is not generally used in practice. The reader interested in Model 3 is referred to the Model 3 tutorial (Knight, 1999), which is also good background for understanding Model 4 (as well as providing a good first view of statistical word alignment and SMT in general).

The good performance of Model 4 is the basis for the work on modeling in this thesis, and Model 4 is used in much of the work in Statistical Machine Translation published in the last several years. Model 4 is a generalization of Model 3 where the alignment model uses relative positions rather than absolute positions. The alignment model is again inverted from that used by Model 1 and Model 2. The reader is referred to Appendix A for a full presentation of Model 4 including a discussion of the generative story with examples. LEAF suffers from the same deficiency as Model 4 and introduces additional deficiency in the source non-head word linking decisions, but we have seen no evidence that this causes problems in practice.

Model 5 is the last model in the chain of IBM models. Model 5 is similar to Model 4, except that Model 5 is not deficient. Model 5 is not typically used because avoiding the deficiency of Model 4 requires a much larger number of parameters than Model 4

has, and because Model 5 has not been shown to perform better than Model 4, despite Model 4's deficiency (Och & Ney, 2003).

The advantages of Model 4 over Model 1 and Model 2 come from the more powerful model which better captures translational correspondence, but this comes at a high price. Both estimation and search over the full distribution of alignments becomes intractable. In practice, a local hillclimbing search is used during the E-step (as discussed for Model 4 in Appendix A.2.5.2, note that this is similar to the "basic" search algorithm used with LEAF discussed in Section 3.4), to find a small set of probable alignments, and the model is re-estimated using only this set (i.e. with the assumption that alignments outside this set have probability $0$). LEAF also requires local hillclimbing search and re-estimation from a small set of probable alignments.

The unsupervised baseline in this thesis involves first training Model 1, then training the HMM word alignment model (the HMM has similarities to Model 2 but performs better than Model 2; it is described in the next section), and then Model 4. Appendix A includes a presentation of the baseline unsupervised system which uses the GIZA++ implementation of Model 4 in both directions (the 1-to-N direction and the M-to-1 direction), followed by the application of symmetrization heuristics to produce the final M-to-N discontinuous alignment.

LEAF improves on Model 4 by providing a generative story which allows the modeling of M-to-N discontinuous alignment structure rather than the 1-to-N structure modeled by the IBM Models. This is a better structure of translational correspondence than that modeled in the IBM models. In practice, this means that LEAF has the important advantage that it does not require heuristic symmetrization and is able to model the full range of translational correspondences we are interested in directly. LEAF can provide a posterior distribution over likely M-to-N discontinuous alignment hypotheses, which is impossible to obtain from Model 4 without using both symmetrization heuristics and heuristic combination of the 1-to-N and M-to-1 posterior probabilities.

#### 3.6.1.2 HMM Word Alignment Models

Much of the additional work on generative modeling of 1-to-N word alignments is based on the HMM word alignment model (Vogel et al., 1996), which is itself a generalization of ideas presented by Dagan et al. (1993). The HMM word alignment model uses an alignment model which has relative positions, like IBM Model 4, rather than using an alignment model involving absolute positions which are used with models like IBM Model 2. We observe the French words, which are the emissions of the HMM, and we know that there are $l$ states, the English words. The transition parameters are tied by distance. For example, suppose we already emitted the French word at position $j$ from state $i$. The transition probability of transitioning from state $i$ to $i'$ (which would

mean that we would emit the French word at position $j + 1$ from $i'$) is conditioned on the signed distance $i' - i$.

Many research groups are interested in the HMM because it can be efficiently trained using the Forward-Backward algorithm, and inference is also tractable. One important difference with Model 4 is that the HMM does not have a fertility distribution. The fertility distribution is important to the good performance of Model 4, and there have been several attempts to at least partially overcome the lack of a fertility distribution in the HMM (without losing the benefits of tractable inference) as we will discuss further below.

Och and Ney (2003) presented extensions to the HMM word alignment model which allow NULL (which emits spurious target words) to be modeled using $l$ additional states (recall that $l$ is the length of the English sentence). The choice of this state encodes the position $i$ of the previous non-NULL English word (the state from which we transitioned into a NULL state). This allows the appropriate NULL state to "remember" the previous non-NULL English word, so that transition probabilities out of the NULL states can be based on the previous English word. If this were not done, and we have only one NULL state, this state would "forget" where in the English sentence the last non-spurious word was emitted from. This formulation adds one additional free parameter, the probability of a jump to the appropriate spurious word state (in fact, the formulation requires $l$ free parameters, but in practice these are tied). An additional free

parameter is used to control an interpolation of the relative position alignment model with a uniform position alignment model, which is used to smooth the relative position alignment model. These two parameters must be optimized on held out data. In practice, we have found the parameter controlling the jump to the NULL states to be particularly important for good performance. Och and Ney (2003) also proposed lexicalizing the non-NULL jump probabilities with word classes to create a class-based HMM.

Toutanova et al. (2002) and Lopez and Resnik (2005) presented a variety of refinements of the HMM word alignment model particularly effective for low data conditions. Toutanova et al. (2002) reported on extending the HMM word alignment model in three ways: using POS-based translation probabilities, making the jump to NULL dependent on the identity of the English word and conditioning the generation of spurious French words on the following French word. Lopez and Resnik (2005) introduced syntactically motivated jump distance features based on the distance within a dependency parse and improved initialization of both the translation and alignment position models.

The model which was presented by Deng and Byrne (2005) is an extension of the HMM which modifies the HMM to be able to emit a phrase of words at each state (recall that a state is an English word). Optionally, a word-level bigram formulation can be used to model which words are in a phrase, otherwise a word-level unigram model is used. A free parameter is used to tune whether longer or shorter phrases

are desired. Like the extension of the HMM presented by Och and Ney (2003), the state space is multiplied by two to model spurious target generation (though here we are referring to spurious phrases rather than spurious words), and the probability of outputting a spurious phrase is a free parameter. To more robustly model the alignment position distribution a linear interpolation of the usual HMM relative position model is performed with an absolute position model (like Model 2's alignment model) and a simple uniform position model. This interpolation of these three quantities introduces another two free parameters. These four free parameters must be optimized against held out data, which, given our experience with the HMM, is likely to be important to performance. The structure modeled by Deng and Byrne (2005) is 1-to-N. When trained in both training directions (using different settings for the free parameters of the two directions), the improvements in the model were competitive with Model 4 (for the special case of monotone translation). However, the largest improvements were obtained by using a technique which "second guesses" the final symmetrized alignment, which is easier to do with models which support exact inference (like the HMM) than with LEAF or Model 4. This "second guessing" provides translations for phrases which were not covered by the symmetrized alignment, we will discuss this in detail in Section 3.6.7.

#### 3.6.1.3 Discussion

Model 4 and the HMM share one important characteristic. The reordering models (also called "distortion" models) use relative positions, i.e. that there is a greater than zero order dependency on word placement. The "homogeneous" HMM word alignment model has a first order dependency (the position of the next placed word is conditioned only on the position of the previously placed word). The extended HMM word alignment model of Och and Ney (2003) remembers the location of the previously placed non-NULL word. Model 4 conditions the alignment model on the location of the previous "cept center" for the first word (from the left) generated from an English word, on the position of the previous word generated from an English word if the word being placed is not the first word generated, and also uses word classes (see Appendix A for more details). These models appear to be successively more powerful. LEAF uses a similar ordering model to Model 4 with the important difference that the distortion is relative to explicitly chosen head words[4].

The lack of fertility in the HMM is a strong difference with Model 4. Toutanova argues for using a probability of "staying" in a source word to try to indirectly model fertility. Deng and Byrne use phrase length probabilities for each emission. Both of these can not directly model fertility because the state can be returned to multiple times,

[4]The placement of the third and subsequent words in a cept is relative to the placement of the previous word, which is more similar to the modeling of distortion in Model 4.

but they may provide a useful bias which partially makes up for the lack of an explicit fertility model. Model 4's fertility model is its main strength over the HMM, as it provides a more robust global model of generation (e.g. in order for an English word to generate words in two very different parts of the sentence it pays both a distortion cost and a fertility cost; for the HMM this is just a distortion cost which is easily offset by avoiding a low probability translation). LEAF has an explicit model of fertility which is similar to Model 4's but is also conditioned on $\gamma$ which indicates whether the source cept is a single word. We have experimented with conditioning this decision as well on the word class of the target head word, but found that performance degraded, indicating that such a distribution can not be robustly estimated with the amount of data we currently have available[5].

In general, LEAF improves on the HMM by providing a generative story which allows the modeling of M-to-N discontinuous alignment structure rather than the 1-to-N alignment structure modeled by the HMM. As in the case of Model 4, the predictions of the HMM word alignment model are 1-to-N, which requires heuristic symmetrization of predictions in both training directions. However, an important difference with both LEAF and Model 4 is that HMM word alignment models support tractable exact estimation and prediction, which explains their interest to the research community. We

[5]One approach to remedying this might be to use fewer head word classes, we currently use 50.

bootstrap both LEAF and our baseline Model 4 system from the HMM as implemented in GIZA++.

A disadvantage which both Model 4 and the HMM variants have in common is the existence of several free parameters which must be optimized on held out data in an expensive end-to-end heuristic search which is either manually done, or often simply not done at all (in which case parameters optimized for a different task are used). Unsupervised LEAF has the advantage that it requires no free parameters, but this lack of direct control over important parameters contributes to poor performance if the bootstrap distributions are not well estimated (this appears to be the case for the unsupervised Arabic/English experiment we reported in Section 3.5). In Chapter 4 we show how a small number of parameters can be trained using as few as 100 sentences of annotated data as an integral part of a semi-supervised training process. This can be viewed as a practical way to avoid the manual optimization process required when using such free parameters while still obtaining the benefits of such an optimization.

#### 3.6.1.4 Other Generative Models of 1-to-N Structure

Moore (2004) reported on modifications to the training of IBM Model 1, which serve to improve the quality of the Viterbi alignment of Model 1. Moore noted that using techniques which may reduce the accuracy of the full distribution over possible alignments in favor of strongly sharpening the Viterbi estimate, may be counter productive

if the model is subsequently used to bootstrap, as is done with both LEAF and our baseline. However, Moore motivated his work by discussing applications other than word alignment which use Model 1, including sentence alignment (Moore, 2002) in particular.

Och and Ney (2003) presented "Model 6", which is a log-linear combination of Model 4 and the HMM. The motivation for this combination is that the distortion (re-ordering) model for the HMM is in the inverse direction of that of Model 4, and so combining their predictions may be more robust. In practice, Model 6 is not used to create alignments for state of the art SMT systems. Symmetric LEAF calculates a relative distortion model in both directions, and uses a differently parameterized model for determining source non-head word to head word links (again in both directions), so it captures this same effect in a stronger fashion.

### 3.6.2 Generative Models of 1-to-1 Structure

Another popular choice has been to use the 1-to-1 alignment structure. The discussion in Section 1.2.4 and particularly Table 1.6 on Page 10 shows that this structure is inadequate in accounting for translational correspondence. However, search over this type of structure is simple. Wu (1997) and Melamed (2000) and Cherry and Lin (2003) all used this structure. It is possible that 1-to-1 alignment structure may be of some interest for applications other than machine translation with a strong emphasis on precision,

such as the extraction of single word translation lexicons for use in Cross-Lingual Information Retrieval (Xu et al., 2001), but further study is needed to determine whether this is in fact the case or whether the low recall of 1-to-1 alignment approaches causes problems.

Wu (1997) invented hierarchical alignment, using operations on parallel binary trees, which were modeled as hidden variables, and a word level lexicon to establish translational correspondence. This allows for highly tractable estimation and inference, but has not been used effectively to improve translation.

Melamed (2000) introduced "competitive linking" which is a heuristically motivated combined modeling/search approach which involves a greedy 1-to-1 matching of English and French words. Cherry and Lin (2003) used a probabilistic model similar to Melamed (2000) and two constraints, the 1-to-1 constraint and the no crossing dependencies ("cohesion") constraint. Two sets of features are used in their model, "adjacency" features (which rewards groups of words for clustering together) and "dependency" features (a word movement penalty based on dependency trees generated using the MiniPar dependency parser). LEAF's placement models encode knowledge similar to Cherry and Lin's non-syntactic features here, but the syntactic features may capture a generalization that is of interest in the semi-supervised approach we present in Chapter 4.

Yamada and Knight (2001) presented a tree-to-string alignment model. The model is trained using English syntactic trees generated from a high quality syntactic parser and Japanese strings. A particular generative story applies operations to the English tree to generate the Japanese string, and this induces an alignment. The operations on the tree allowed by the generative story include three kinds of operations, the reordering of English constituents within an English constituent phrase, translation actions mapping English to Japanese, and insertion of NULL words. This model was not used to try to generate a good Viterbi alignment, but instead to directly learn a good estimate for $p(f_{\text{string}}|e_{\text{tree}})$ which is then applied during translation (translation is the recovery of an English tree given a Japanese string), in conjunction with a language model which models the probability of an English tree. This model uses a 1-to-1 structure for the majority of the translation actions, which are translations of the leaves of the English parse tree, but was later extended to allow phrasal translations of constituents in the parse tree (however, this was not implemented in the alignment model). Gildea (2003) extended this model to tree-to-tree alignment and enhanced both tree-to-string and tree-to-tree generative stories with an operation called “clone” which allows models to be more powerful and less tied to the original tree structure (or structures). LEAF induces a roughly dependency-like relationship in the links between a single head word and multiple non-head words, but this is more semantically motivated than syntactically motivated.

1-to-1 alignments make very few predictions, so they have a bias toward high precision but low recall. Estimation (and prediction) using 1-to-1 alignment structure is highly tractable, but unfortunately this structure is not a good choice for building MT systems. As we showed in Chapter 2, the AER metric unfairly favors high precision alignments, which has encouraged research using this structure, but none of this research has been shown to improve machine translation quality.

### 3.6.3 Generative Models of "Phrase-based" Structure

The phrase-based (consecutive word) alignment structure has also been used in several alignment models, though it is more often used in translation models. The discussion in Section 1.2.4 and particularly Table 1.6 on Page 10 shows that the phrase-based assumption is also not a good choice of alignment structure, and we mention again that even phrase-based SMT models do not perform ideally with alignments generated using a phrase-based alignment structure.

#### 3.6.3.1 General consecutive word alignment models

Marcu and Wong (2002) defined the Joint model, which modeled consecutive word M-to-N alignments. When used as a translation model, the Joint model is interesting because it uses a distribution over phrase translations directly, rather than estimating it from a Viterbi alignment. The model has a strong memorization capability and seems to

match the assumptions behind phrase-based SMT closely. However, this memorization capability leads to problems in generalization and in tractability. In the Joint model, unlike in LEAF, overlapping phrases do not share parameters. For instance, the probability of the French cept “homme” translating to the English cept “man” is not directly related to the probability of the French cept “homme” translating as the English cept “the man”. This leads to a large blow-up in the number of parameters, causing the intractability problems, and leads to poor generalization. The Joint model also does not have the ability to deal with non-parallelism (which is annotated using NULL alignments in most other translation models). Kumar et al. (2006) used the phrase-based version of the alignment templates translation framework of Och and Ney (2004) to build an alignment model which is similar to the Joint model.

The problem with the blow-up in parameter space involved in models like the Joint model is addressed in LEAF by using the head word structure to allow the phrase probabilities to decompose into smaller units. In particular, this appears to provide a good trade-off between robustness and expressiveness given the amount of training data currently available. The M-to-N discontinuous alignment structure using the head word assumption is also faster to search than a pure phrase-based structure as the translation dependencies on one side are only dependent on the head word on the other side (and $\gamma$ which is a flag indicating whether the cept on the other side contains just one word). The decomposition of costs using the head word assumption means that adding

a non-head word to a head word is an operation which incurs additional cost but does not cause all of the other costs incurred by that cept to be reevaluated. In phrase-based models any change to a cept causes all costs to be reevaluated. LEAF also provides us with a path to easily increase the power of the model by simply reducing reliance on word classes and further relaxing conditional independence assumptions.

#### 3.6.3.2 Other "phrasal" models

Other alignment structures have been tried which are loosely phrasal. Wang and Waibel (1998) introduced a generative story based on extension of the generative story of Model 4. The alignment structure modeled was "consecutive M to non-consecutive N", and the parameters were trained using EM. LEAF has some similarities with this model in that they are both based on generative stories which are extensions of Model 4. However, LEAF allows the full range of M-to-N discontiguous alignments.

Tiedemann (2003) created an algorithm similar to Melamed's competitive linking algorithm, but allowing adjacent word connections. This structure has similarities to the "Refined" heuristic symmetrization metric of Och and Ney (2003) which we discussed in Chapter 2. A variety of features were used including features based on POS tags and similarity heuristics. We will propose a semi-supervised training algorithm which could use these types of features in Chapter 4.

### 3.6.4 Generative Models of 1-to-N and M-to-1 Structure

Matusov et al. (2004) presented a model capable of modeling 1-to-N and M-to-1 alignments (but not M-to-N alignments) which was bootstrapped from Model 4. The technique used for bootstrapping is to use state occupation probabilities. State occupation probabilities can be exactly determined for the HMM but only approximately determined for Model 4; this involves using a sample of the Model 4 posterior distribution which is calculated over a small set of alignments which are hopefully near to the best alignment. We suspect that this is not more powerful than simply estimating a model directly from the Model 4 Viterbi alignment (and could even be inferior), but these two options have never been directly compared. The state occupation probabilities are then used in combination with the Hungarian algorithm to solve a bipartite covering problem which derives a 1-to-N and M-to-1 alignment. However the decisions made are based only on the state occupation probabilities which don't model the global context well[6].

[6]This is easiest to illustrate with an example. Suppose an estimate of Model 4 prefers to assign the French word at the beginning of a particular French sentence to the first English word 50% of the time and the French word at the end of the French sentence to the same English word 50% of the time. This can easily be captured in the state occupation probabilities. But this fails to capture any interaction between these two alignment decisions. If the highly probable alignments which have the first French word aligned to the first English word never contain an alignment of the last French word to the first English word (because the distortion probabilities involved in making a placement at the beginning of the French sentence and at the end of the French sentence of words generated from the same English word are low), this interaction would be lost using state occupation probabilities.

Because of this, we doubt that using HMM or Model 4 state occupation probabilities would be as effective as bootstrapping LEAF from the HMM.

### 3.6.5 Generative models of M-to-N Discontinuous Structure

LEAF is the only general purpose alignment model which models M-to-N discontinuous structure which we are aware of. However, May and Knight (2007) defined a model which can be used to re-align given a high quality word alignment and an English parse tree. This work uses the GHKM translation model (Galley et al., 2006) as an alignment model.

May and Knight (2007) used this model to re-align from a starting alignment and a fixed parse tree. The parse tree is treated as a fixed hard constraint. First an inventory of treelet/alignment pairs is created from the starting alignment and the fixed parse tree. Then EM is used to find better treelet/alignment pairs for maximizing the likelihood of the training data then were originally used (note that all of the treelet/alignment pairs considered for a particular sentence must have been observed in the starting minimal Viterbi derivation of either the sentence in question or a different training sentence). Finally the Viterbi treelet/alignment derivation is found for each sentence pair. This work allows a generative model to take advantage of syntactic information. However, it has some of the same issues with overlapping rules as phrasal systems do. This is partially addressed by adding a “rule size” distribution which is analogous to a fertility

distribution (but is over rule size rather than the number of words generated). We would be interested in taking advantage of syntactic information in LEAF, but as the parse tree is not perfect (it is generated by a probabilistic parser, which makes errors) we think the appropriate way to do this would be to define syntactically motivated sub-models in our semi-supervised formulation, which will be discussed in Chapter 4.

### 3.6.6 Symmetrization

One important aspect of LEAF is its symmetry. Och and Ney (2003) invented heuristic symmetrization of the output of a 1-to-N model and a M-to-1 model resulting in a M-to-N alignment, this was extended by Koehn et al. (2003). Zens et al. (2004) introduced symmetrized lexicon training. Liang et al. (2006) showed how to train two HMM word alignment models, a 1-to-N model and a M-to-1 model, to agree in predicting all of the links generated, resulting in a 1-to-1 alignment with occasional rare 1-to-N or M-to-1 links. We have used insights from these works to help determine the structure of our generative model.

Various models have attempted to gain the advantages of using these symmetrization heuristics, but most have been required to deal with 1-best predictions (or with state occupation probabilities). LEAF uses the head word structure in a symmetric fashion inside of the generative story, which seems to be a better way to model the desired

structure. In particular, this allows for a posterior distribution over more than the 1-best alignment without the use of heuristics.

### 3.6.7 Different Rule/Phrase Extraction

The work reported in this thesis used translation systems which extract translation rules from a single word alignment (Koehn et al., 2003). One promising area of translation modeling research is work on extracting translations rules from richer representations than a single word alignment. The IBM models (Brown et al., 1993) and the Joint model (Marcu & Wong, 2002) were designed to estimate parameters (for 1-to-N and phrase-based translation models respectively) directly without requiring the use of a Viterbi alignment. Venugopal et al. (2003) invented a generalized technique for using lower order alignment models such as Model 1 to generate phrase pairs given a source language test set and an unaligned bitext.

Deng and Byrne (2005) described an approach which is used as a post-process for finding translations of phrases in a translation test set which did not have translation candidates indicated in the symmetrized alignment. This is a form of "second guessing" the symmetrized alignment. It involves using a modified Forward algorithm for estimating the posterior probability of each possible phrase pair (according to symmetrically trained phrase-based HMM models). They used this approach together with symmetrized phrase-based HMM alignments to obtain improved BLEU

scores over just using the symmetrized phrase-based HMM alignments. They also obtained improved BLEU scores when using the posteriors calculated over symmetric phrase-based HMM models to extract translations for phrases which were not covered in symmetrized Model 4 alignments. The implementation of this approach requires the calculation of quantities similar to the state occupation probabilities of Matusov et al. (2004). This relaxation of the Viterbi alignment assumption for phrasal or hierarchical rule extraction seems to us to be a logical extension of our current approach. Implementing this for LEAF would require modifications to the model to allow it to generate the most probable alignment subject to the constraint that at least one translation of a certain phrase can be extracted; we will discuss this further in Chapter 5.

### 3.6.8 Discussion

We have outlined some of the important previous work on word alignment. We chose to break this work down by the alignment structure modeled, as our choice of a better alignment structure was critical to the design of LEAF.

However, there are other dimensions on which we could expand. One very important dimension is the treatment of syntactic phenomena. In designing LEAF, we were not only inspired by Model 4, but also by dependency-based alignment models. We discussed some of the dependency-based word alignment models in the sections on 1-to-1, phrase-based and M-to-N discontinuous structures. In contrast with their approaches,

we have a very flat, one-level notion of dependency, which is semantically motivated and learned automatically from the parallel corpus. This idea of dependency has some similarity with hierarchical SMT models such as the Hiero model (Chiang, 2005).

## 3.7 Summary

Our new generative model, LEAF, is able to model alignments which consist of M-to-N non-consecutive minimal translational correspondences. We presented the generative story and mathematical formulation.

We then discussed the training of LEAF using an approximate Expectation-Maximization training algorithm. We discussed the E-step, the M-step, and bootstrapping (performing the initial M-step).

We use a local search algorithm to search for likely alignments. We presented the permutation operators used and discussed how to use them in a basic hillclimbing algorithm. We also derived an improved hillclimbing algorithm using "Tabu" alignments and restarts, and performed a simple experiment showing that it is effective.

We conducted experiments on large French/English and Arabic/English data sets which show that LEAF is comparable with our baseline, GIZA++, when LEAF is trained in an unsupervised fashion.

We then discussed the extensive body of previous work on generative modeling of word alignment. We broke the discussion down by the alignment structure modeled,

with the two most important structures being the "1-to-N" structure as used in the IBM models and the HMM, and the "phrase-based" (consecutive word) structure as used in phrase-based models. We contrasted LEAF's M-to-N non-consecutive alignment structure with both of these structures and discussed the advantages of the head word assumption, and in particular how this approach solves the phrasal segmentation problem of phrase-based models, where overlapping phrases cause problems with both tractability and robustness. We also discussed two other issues, symmetricity and approaches to building translation systems which use more than just the Viterbi word alignment.

In conclusion, we have found a new structure over which we can robustly predict which directly models translational correspondence commensurate with how it is used in hierarchical SMT systems. Surprisingly, this is also a more suitable structure for general phrase-based SMT systems than the phrase-based alignment structure. Our model, LEAF, is comparable with a strong baseline when it is trained in an unsupervised fashion. In Chapter 4 we will decompose LEAF to derive the sub-models of a powerful semi-supervised model and show that this model has significantly better performance than two strong baselines.

## 3.8 Research Contribution

We designed a new generative model which models the structure of the word alignment problem directly.

We also developed a high performance distributed local search algorithm.

# Chapter 4

# Minimum Error / Maximum Likelihood Training for Automatic Word Alignment

## 4.1 Introduction

The technique of using labeled data and unlabeled data together for training is called semi-supervised training. We are interested in developing a semi-supervised training technique for the word alignment problem: we have a large number of parameters to estimate, a large amount of unlabeled data, and a small amount of labeled data. We have a structured generative model, LEAF, which can be trained in an unsupervised fashion on the unlabeled data, and now we would like to take advantage of the labeled data.

When we refer to labeled data for the automatic word alignment problem, we mean parallel sentences for which a correct word alignment has been annotated by humans.

Unlabeled data refers to a pair of sentences which we assume are parallel (as they were chosen using a sentence alignment program which is known to have high accuracy in making this determination). Unlabeled data do not have human annotated word alignments associated with them, which is why we call them unlabeled.

We first show how to discriminatively rerank the output of a generative model to minimize the errors on the labeled data. We then present a new semi-supervised training approach called Minimum Error / Maximum Likelihood training which incorporates steps which alternatively minimize error with respect to the final performance criterion and maximize the likelihood of the underlying generative model.

## 4.2 Discriminative Reranking for Generative Word Alignment Models

The idea behind applying discriminative training to generative models is to enable us to use a discriminative criterion to access knowledge which can not be directly integrated into the generative model (because of the need to reengineer the generative story).

Discriminative reranking of the output of a generative model uses a representation of the guesses of the generative model. If this representation explicitly enumerates the best N complete hypotheses, it is called an N best list. The hypotheses are ranked by their probabilities. Discriminatively reranking an N best list allows the use of additional

knowledge which would be difficult to incorporate directly into the generative model to produce a new ranking (i.e. different probability scores for the hypotheses in the N best list). If additional knowledge sources are effectively combined with the knowledge sources in the original generative model, this ranking will be better than (or at least as good as) the ranking output by the original model.

We present a new discriminative reranking method which we will apply to an N best list generated using LEAF. After presenting relevant previous work on discriminative reranking, we will generalize this to a new semi-supervised training approach.

### 4.2.1 Reinterpreting LEAF as a Log-Linear Model

In this section we will reinterpret LEAF as a log-linear model. This form of model will allow us to use the distributions which make up LEAF in a discriminatively trained model, as we will explain in the next two sections.

We use the term "sub-model" to refer to the components of our models. This emphasizes that most of these "sub-models" are in fact models which are estimated from data. These "sub-models" often have parameters and rely on what we normally think of as "features" for their parameterization. However, not all of our sub-models will have parameters (for instance, we could imagine defining a sub-model which is simply the percentage of the French words which are unaligned). A sub-model is simply a function applied to an alignment which outputs a real number (we hope that the reader

who prefers to call this a “feature function” or “feature” will simply mentally translate “sub-model” to their preferred term). An effective sub-model can be used to tell us whether to prefer one hypothesized alignment over another. If we view the numbers output by a sub-model as negative log probabilities, then a high number (cost) assigns the alignment a low probability, while a low number assigns the alignment a high probability.

In this section we reinterpret the probability distributions of LEAF listed in Table 4.1 as sub-models of a log-linear model and estimate the weights associated with each sub-model. The model formulation is given in Equation 4.1. We reinterpret the new generative model as having ten sub-models in the source to target direction, and ten sub-models in the target to source direction, for a total of twenty sub-models, which are listed in Table 4.1. Each sub-model $m$ has an associated weight $\lambda_m$. Our approach can also be applied to additional sub-models which are not part of the original generative model, which will be discussed in Section 4.8.1.

$$p_\lambda(a, f|e) = \frac{exp(\sum_i \lambda_i h_i(f, a, e))}{\sum_{f',a'} exp(\sum_i \lambda_i h_i(f', a', e))} \tag{4.1}$$

Given a vector of weights $\lambda$, the alignment search problem, i.e. the search to return the best alignment $\hat{a}$ of $e$ and $f$ according to the model, is in Equation 4.2.

| | | |
|---|---|---|
| 1 | $g(\chi_i \mid e_i)$ | source word type |
| 2 | $w_{-1}(i - \mu_i \mid \text{class}_e(e_i))$ | choosing a head word |
| 3 | $t_1(f_j \mid e_i)$ | head word translation |
| 4 | $s(\psi_i \mid e_i, \gamma_i)$ | $\psi_i$ is number of words in target cept |
| 5 | $s_0(\psi_0 \mid \sum_i \psi_i)$ | number of unaligned target words |
| 6 | $t_0(f_j)$ | identity of unaligned target words |
| 7 | $t_{>1}(f_j \mid e_i, \text{class}_h(\tau_{i1}))$ | non-head word translation |
| 8 | $d_1(\triangle j \mid \text{class}_e(e_\rho), \text{class}_f(f_j))$ | movement for target head words |
| 9 | $d_2(\triangle j \mid \text{class}_f(f_j))$ | movement for left-most target non-head word |
| 10 | $d_{>2}(\triangle j \mid \text{class}_f(f_j))$ | movement for subsequent target non-head words |
| 11-20 | | (same features, target to source direction) |

Table 4.1: Sub-models derived from LEAF

$$\hat{a} = \underset{a}{\text{argmax}}\, p_\lambda(a|f,e) = \underset{a}{\text{argmax}}\, p_\lambda(a,f|e) = \underset{a}{\text{argmax}}\, exp(\sum_i \lambda_i h_i(f,a,e)) \quad (4.2)$$

### 4.2.2 Discriminative Training Algorithm

Given a hypothesized alignment $a$, a gold standard alignment $g$, and the English and French sentences, we can calculate an error function, $E(a, g, e, f)$. We would like to minimize the error function by finding the best $\lambda$ settings. This is a supervised learning problem, the discriminative training problem, listed in Equation 4.3.

$$\underset{\lambda}{\text{argmin}}\, E(\hat{a}, g, e, f) \text{ where } \hat{a} \text{ is as defined in Equation 4.2} \quad (4.3)$$

Because this is a structured learning problem over the enormous space of $\lambda$ vectors, exact inference is intractable. We will instead develop an iterative process for solving

Equation 4.3. We will learn optimal weights over a (growing) set of hypotheses for a small number of parallel sentences for which we have gold standard alignments. We use $1 -$ F-measure$(\alpha)$ as our error function, comparing hypothesized word alignments for the discriminative training set (often referred to as the "development" or "dev" set) with the gold standard.

The discriminative reranking algorithm is initialized with the parameters of the sub-models $\theta$ (which are the final distributions estimated during unsupervised training of the generative model), an initial choice of the $\lambda$ vector, gold standard word alignments (labels) for the alignment discriminative training set, the constant N specifying the size of the N best list[1], and an empty master set of hypothesized alignments. The algorithm consists of repeatedly running a loop which consists of three main steps:

LOOP:

1. Produce an N best list using $\lambda$ by solving Equation 4.2). If all of the hypotheses in the N best list are already in the master set of hypotheses, the algorithm has converged, so terminate the loop. Otherwise add new hypotheses to the master set of hypotheses.

2. In this step, we choose the best $\lambda$ vector to minimize error from a set of candidates. The candidates are our current $\lambda$ vector, any $\lambda$ vectors which were chosen

[1] $N = 128$ for our experiments

previously in Steps 2 and 3, and 999 randomly generated vectors. Given these candidate $\lambda$ vectors we apply each of them to the master set of hypotheses in order to determine the top ranked alignment $a'$, and and then evaluate the error function $E(a', g, e, f)$. We set $\lambda$ to the $\lambda$ vector which resulted in the alignments with the lowest error (i.e. the highest F-measure$(\alpha)$ score since we use $1 -$ F-measure$(\alpha)$ as our error criterion), so we have solved Equation 4.3.

3. Run a "city block" error minimization step which results in a new vector $\lambda$. This minimization also involves solving Equation 4.3, but is more complex than simply evaluating the error of several $\lambda$ candidates. The implementation of "city block" minimization for our problem is discussed in detail below.

Step 3 of the algorithm tries to find the best $\lambda$ setting over the set of hypotheses for the sentences in the discriminative training set using numerical optimization. This is an M-dimensional optimization problem (where M is the number of sub-models). Minimizing error for all of the weights at once is not computationally feasible. We initially applied Powell's Method (Press et al., 2002), using Brent's Method (Press et al., 2002) for line minimization, but found this to be ineffective. This is might be because the assumption that the error surface is quadratic was violated and the line minimization was then quickly trapped in local error minima which were much worse than the global error minima.

Och (2003) has described an efficient exact one-dimensional error minimization technique for a similar search problem, which we will adapt to our problem. This involves calculating a piecewise constant function. This function, which is calculated for a fixed sub-model $m$, is a function of one variable $x$. The function directly evaluates the error of the hypotheses which would be picked by equation 4.2 if we hold all weights constant, except for the weight ($\lambda_m$ for some $m$) under consideration, which is set to $x$. The formula for such a function for sub-model $m$, which we call $f_m(x)$ is given in Equation 4.4.

$$f_m(x) = E(\underset{a}{\operatorname{argmax}}\, exp(x * h_m(f, a, e) + \sum_{i \neq m} \lambda_i h_i(f, a, e)), g, e, f) \tag{4.4}$$

We implement "city block" minimization by first calculating the M functions. Once we have calculated an explicit representation of each of the functions $f_m$, we can quickly find the error minima (the $x$ value resulting in lowest error) for each $f_m$. We then choose the sub-model $m$ and the value $x$ resulting in the lowest error minima and set $\lambda_m = x$. We iterate this process until no further reduction in error can be found.

We can in fact generalize Equation 4.4 to calculate a function for any line in the M-dimensional space (not just the M unit vectors). It would seem obvious that we should use exact line minimizations in place of Brent's method and apply Powell's

method. However, counter-intuitively, we have found that in practice Powell's method is quickly trapped in local error minima even with the exact line minimizations. We have instead found it more effective to perform "city block" minimization over just the M unit vectors.

In automatic word alignment problems using a large number of sub-models, the outcome of Step 3 is sensitive to the starting point. If we consider just steps 2 and 3, then we can define a search error as a failure to find the best $\lambda$ value for minimizing the error of the hypothesis chosen from the current master set of hypotheses using Equation 4.2. Performing step 2, which vets both the $\lambda$ vectors which were found useful previously and a large number of random $\lambda$ vectors, and then using the best result as the starting point of the "city block" minimization in step 3 seems to reduce search errors to an acceptable level, but we believe that in future work we will be able to improve on this.

## 4.3 Previous Work in Discriminative Training

Discriminative reranking has been used successfully in many areas of NLP. A good example area is syntactic parsing. For parsing, discriminative reranking was introduced by (Collins, 2000). He starts with an underlying generative model which models the joint generation of a sentence and its parse-tree. Given a new sentence to parse, he first selects the best N parse-trees according to his generative model. Then he scores new features, which could not be easily integrated into a new generative story because

their roles in generation would overlap, and learns discriminatively how to rerank the parses in his N best list. He uses a greedy feature selection technique to determine which features are important. Recently a very large number of different approaches to discrimininative reranking have been applied to syntactic parsing, and there have also been a large number of more general discriminative training algorithms used. One discriminative training algorithm of particular interest to us is training using the averaged perceptron (Collins, 2002), which was refined and applied to word alignment by Moore (2005); this will be discussed in Section 4.4.3.

Discriminative reranking has also been applied to machine translation. Och et al. (2003) and Och et al. (2004) used a large number of feature functions and the discriminative training technique defined by Och (2003) to rerank N best lists of hypothesized English translations for Chinese sentences to improve the quality of translations. Shen and Joshi (2005) evaluated maximum margin approaches for the same task.

Other approaches to discriminative training based on an underlying generative model have been applied in NLP. We present work in the area of machine translation, as it is relevant to the discriminative training approach we will take. Och and Ney (2002) introduced a log-linear model for translation composed of a collection of sub-models which are estimated using various techniques. These included several sub-models estimated by taking the relative frequency of consecutive word phrases extracted from the one-best output of symmetrized Model 4 alignments and also included sub-models

which backed off estimation of phrase-to-phrase translation probabilities to a word-level translation lexicon. Both the maximum mutual information (MMI) and the minimum classification error (MCE) criteria were tried. Och (2003) introduced direct error minimization for statistical machine translation using the same log-linear model, and showed that discriminative training to the final performance criterion, BLEU, is superior to training using MMI or MCE. Other optimization techniques are possible with log-linear models. For instance Zens and Ney (2004) used the downhill simplex method to train weights for both phrase-based and alignment-template-based translation, and Cettolo and Federico (2004) used the downhill simplex method to train weights for a log-linear model involving a reinterpretation of the Model 4 sub-models for translation.

The approaches to discriminative reranking and discriminative training for Machine Translation which we have discussed use a log-linear model to integrate sub-models of widely varying granularity. The log-linear model is trained either to a criterion which maximizes entropy, or to directly maximize the final performance criterion. Och (2003) showed that the latter performs well in practice. When training to the final performance criterion is chosen, two approaches to discriminative training are generally used. The simpler approach is to generate candidate vectors of weights and evaluate the results; the down simplex optimization method (Press et al., 2002) is commonly applied here. We apply this type of approach in step 2 of our discriminative algorithm in an even simpler fashion, by simply generating random vectors and evaluating them. The other

approach, introduced for translation by Och (2003), is to optimize over N best lists using exact line minimizations. This puts the performance criterion inside the optimization. We use exact line minimizations in the "city block" minimization which is performed in step 3 of our algorithm.

## 4.4 Previous Work in Discriminative Modeling for Word Alignment

Previous work on discriminative modeling for word-alignment differs most strongly from the log-linear approach in that it generally views word-alignment as a supervised task. However, all of the state of the art approaches depend on using features from an unsupervised generative model in order to obtain their best results because of the small amount of gold standard word alignments available (Liu et al., 2005; Ittycheriah & Roukos, 2005; Taskar et al., 2005; Ayan & Dorr, 2006b; Lacoste-Julien et al., 2006; Fraser & Marcu, 2006; Blunsom & Cohn, 2006; Moore et al., 2006).

We are most interested in discriminative models which allow the use of many-to-many non-contiguous alignment structure. We are less interested in discriminative models using 1-to-N structure, as the use of 1-to-N requires a heuristic step following the discriminative training to obtain the M-to-N discontiguous alignments actually used to build SMT systems. The use of such a heuristic step means that alignment

quality can not be directly optimized. We will show in Section 4.8 that optimizing F-measure($\alpha$) for 1-to-N and M-to-1 alignment models separately (and then combining their predictions using a symmetrization heuristic such as "Union") is inferior to directly optimizing F-measure($\alpha$) for our M-to-N alignment model.

We are not aware of previous work on discriminative models with a "phrase-based" contiguous M-to-N structure, and given the recent success of hierarchical SMT models (which support gaps in the translation rules) we doubt this is would have strong performance for most data sets. However, it would be simple to implement this to test this assumption. As we discussed in Section 3.2.3, phrase-based structure can be modeled as a special case of LEAF (however, it is important to remember that the conditioning of the generation decisions would be on the head words rather than on the full phrase). EMD could then be applied without modification to a log-linear model using the sub-models derived from this special LEAF model.

### 4.4.1 Discriminative Models of 1-to-1 Structure

After Brown et al. (1993), much of the initial work on generative modeling was done using 1-to-1 structure. This structure is not a good choice for maximizing SMT performance, but is an interesting starting point for researchers who then go on to work on more highly structured output spaces. In particular, search limited to a 1-to-1 alignment structure is fairly simple even for models which use very complex features.

Taskar et al. (2005) took a similar approach to the models of Melamed (2000) and Cherry and Lin (2003), but in a discriminative context, casting the word alignment problem as a maximum weighted bipartite matching problem, which is estimated within the large margin framework using a quadratic program. They use such features as DICE score, orthographic similarity and proximity of (absolute) positions.

Liu et al. (2005) built a log-linear model using the IBM Model 3 alignment score in both directions and discriminatively reranked it. Additional sub-models were a POS-based lexicon model, and a dictionary based lexicon model. They showed small improvements in balanced F-measure with Sure/Possible over symmetrized Model 4, but did not show what the effect is on translation quality. Their discriminative reranking approach is similar to ours, but with important differences. They did not decompose the underlying generative model, which is IBM Model 3. Instead, they used two features based on the score of the full model. These features model 1-to-many and many-to-1 alignments respectively, so they can not directly model many-to-many alignments. One of these two feature functions must have a value of zero unless the hypothesized alignment is a 1-to-1 alignment. The other main difference is that they trained to the Maximum Entropy criterion rather than maximizing the final performance criterion, though they indicate interest in doing this and they use heuristics to try to pick local maxima of the Maximum Entropy training which are better according to the final performance criterion.

### 4.4.2 Discriminative Models of 1-to-N Discontinuous Structure

The 1-to-N structure, used initially in the generative models defined by Brown et al. (1993), has a long and distinguished history. Discriminative approaches which adopt the 1-to-N structure are a logical extension of this.

Berger et al. (1996) defined a word level lexicon model which used varying amounts of context up to 3 words in each direction from the word being translated, and discussed how to train this representation. García-Varea et al. (2002) implemented this in an alignment package. This work defined the lexicon using both word contexts and word class contexts. The system was built by first completely training the IBM models to obtain both the 1-to-N Viterbi alignments in a single direction and the sub-models representing fertility and distortion. The weights of the features for the special lexicon were trained using the Viterbi alignments as training data and the maximum entropy criterion. The fertility and distortion models were then retrained, holding the special lexicon model fixed. Finally the presumed Viterbi alignment was calculated, and this was returned as the final discriminatively reranked result. This work resulted in small gains in balanced F-measure over Model 4 and has not been shown to improve translation quality.

Kumar and Byrne (2002) presented a framework for searching to minimize the Bayes Risk, applied to word alignment. The work presented used IBM Model 3 without a reordering model (i.e., translation and fertility were modeled as in Model 3, but

distortion was modeled as a uniform distribution). The insights in this work could be applied in our framework in the future, once we have a better posterior distribution over word alignments.

Ittycheriah and Roukos (2005) presented a 1-to-N discriminative model trained using the Maximum Entropy criterion specifically for the task of Arabic/English word alignment. They showed balanced F-Measure results which were competitive with 1-to-N GIZA++, and are one of the few works which also compared the resulting MT performance, where they had inconsistent gains over 1-to-N GIZA++ (unfortunately there was no comparison with heuristically symmetrized GIZA++, which would have been a stronger baseline). They invested significant effort in sub-model engineering (producing both sub-models specific to Arabic/English alignment and sub-models which would be useful for other language pairs), while we use sub-models which are derived from LEAF and a few heuristic features. In contrast to their work, all of the sub-models we have presented are language independent.

Blunsom and Cohn (2006) created a Conditional Random Field (CRF) model for the 1-to-N alignment task, and trained it to minimize AER. The model structure was similar to the HMM model in that there was a first-order Markov assumption, but because they were using a CRF they were able to integrate overlapping features (lexica based on string similarity, words and POS tags were all scored for the same link), which would have been difficult to integrate into a generative story.

Previous to our work with LEAF, we used 1-to-N structure within the work we did on training a log-linear model using a mix of features derived from IBM Model 4 and heuristics (Fraser & Marcu, 2006). In this work we optimized the F-Measure($\alpha$) of models in both directions independently, but at each iteration of training we estimated additional word-level lexicons by heuristically symmetrizing the Viterbi alignments taken from both training directions. This is similar to the symmetrized lexicon training of Zens et al. (2004). We will compare the current approach using sub-models derived from LEAF with our previous approach using sub-models derived from Model 4 in the experiments in Section 4.8.

### 4.4.3 Discriminative Models of 1-to-N and M-to-1 Discontinuous Structure

Lacoste-Julien et al. (2006) created a discriminative model restricted to 1-to-1, 1-to-2 and 2-to-1 alignments. This work extends the framework of Taskar et al. (2005) to the "quadratic" case, where there are features on pairs of edges rather than individual edges, allowing them to robustly model 1-to-2 and 2-to-1 alignments. Parameter estimation can be solved exactly as a quadratic assignment problem, but can also be relaxed to be solvable as a quadratic program. Prediction is solved as an integer linear program, but can this also be relaxed. The (relative) tractability of search in this framework is attractive, but this is at the cost of the unreasonable 1-to-2 and 2-to-1 assumptions and

weaker features than the features derived from LEAF. This work valued tractability over the richness of the features, which is at odds with our approach. The approach also requires the use of Hamming loss as the training criterion. Hamming loss has been shown to be effective in reducing AER, but no work has been done to show that it is effective for optimizing a metric which correlates well with machine translation performance. The best results were obtained using features based on intersected Model 4 and symmetric HMMs trained to agree (Liang et al., 2006). The generated alignments were not evaluated in a statistical machine translation system.

Moore et al. (2006) introduced a sequence of two discriminative models called Stage 1 and Stage 2. The final alignments generated are 1-to-1, 1-to-2, 1-to-3, 2-to-1 or 3-to-1 alignments. Unlike the work of Lacoste-Julien et al. (2006), there is nothing in the framework which inherently restricts the N and M variables in the 1-to-N and M-to-1 alignments modeled, and we assume that the choice of 3 for both of these variables was a good choice to minimize AER for the French/English alignment task considered. The Stage 1 model is estimated from the unannotated full training data and the annotated discriminative training set. The Stage 2 model is estimated using the predictions of Stage 1. The features used in Stage 1 include alignment geometry, exact string match, lexical features (for words occurring two or more times in the small discriminative training set), and a ranking induced from the log likelihood ratio calculated over cooccurences of words occurring in parallel sentences in the full training data. The

stage 2 model uses statistics taken from the stage 1 model's predictions on the full training set, in particular an empirically estimated feature which models the probability of a single source word being aligned to a bag of up to 3 target words (or vice versa) and an empirically estimated jump distance feature. The model is trained using the averaged perceptron which requires a heuristic search to find the most probable alignment just as ours does, but a beam decoder is used rather than a hillclimbing search. The averaged perceptron training was compared with using a support vector machine formulation which is designed for structured prediction, and the two approaches had similar performance. The conclusion of this work, that the richness of the features is more important than the discriminative training technique, matches our intuition. Similarly to the work of Lacoste-Julien et al. (2006) the best results were obtained using intersected Model 4 and HMMs trained to agree, and MT performance was not evaluated. We view both of these works as providing an interesting study of features, some of which we intend to try adding to our model in future work.

### 4.4.4 Discriminative Models of M-to-N Discontinuous Structure

Ayan et al. (2005) used transformation based learning to expand the 1-to-1 and 1-to-N discontinuous alignments generated from generative statistical alignment models to general M-to-N discontiguous alignments. They used a small gold word alignment set to learn effective transformations (additions or deletions to the alignment) which

used context modeled using closed-class words, POS tags, and dependency trees. This work integrates interesting features which we will consider using in the future in our semi-supervised approach.

Ayan and Dorr (2006b) used a Maximum Entropy classifier to combine the predictions of several alignment systems. Based on features over the input alignment set geometry and POS tags, they learned to classify whether a particular link that is predicted by at least one of the input alignments should be included in the final alignment. These decisions were made for each link independently as they are conditioned only on the input and not the output. The experiments performed included combining Model 4 and the HMM extensions of Lopez and Resnik (2005). They showed significant improvements in MT quality over heuristic symmetrization for small data sets. Our approach, in contrast, involves a powerful model where alignment links are not considered independently, but maximizing this model requires a search over possible alignment bigraphs of the whole sentence. We could add the predictions of other models into our model in a similar fashion to their work. We have in fact tried combining information in a similar fashion using alignments generated from the HMM Viterbi alignments (which are also what we bootstrap from) in conjunction with using three heuristic symmetrization metrics and found this to be ineffective when using sub-models derived from LEAF (although we note that these same sub-models were effective in our previous 1-to-N log-linear model (Fraser & Marcu, 2006)).

## 4.5 Semi-Supervised Learning

During our discussion of semi-supervised training, we draw a distinction between discriminative training and semi-supervised training, as applied to generative models. In discriminative training we rerank the predictions of a generative model to obtain predictions of higher quality. There is no mechanism so that the discriminative criterion can affect the estimates of the underlying generative model. Discriminative training (when applied to an underlying generative model) can be viewed as a weak form of semi-supervised learning which is missing this important feedback loop.

Most approaches to semi-supervised learning require that the labeled data be sufficient to make a good initial estimate which is then refined using unlabeled data (Seeger, 2000). In fact, the problem of semi-supervised learning is often defined as "using unlabeled data to help supervised learning" (Seeger, 2000). Most work on semi-supervised learning uses underlying generative models which have distributions with a relatively small number of parameters. An initial model is estimated in a supervised fashion using the labeled data, and this supervised model is used to attach labels (or a probability distribution over labels) to the unlabeled data, then a new supervised model is estimated, and this is iterated.

For instance, both Nigam et al. (2000) and Miller and Browning (2003) train an initial supervised classifier and then use EM to improve the initial estimate of posterior class membership probabilities. In cases where there are only a small number of labels

available but a very large number of parameters must be estimated, such as when the number of parameters increases as training data increases, this is not practical. If this technique is applied in these cases, it will lead to the so-called "overconfident pseudo-labeling problem" (Seeger, 2000), where the initial labels of very poor quality assigned to the unlabeled data will at the best have no effect, and at the worst dominate the initial model estimated in the M-step causing convergence to a local minima of very poor quality (with respect to the final performance criterion).

We present the following alternative, which alternatively minimizes error and maximizes likelihood. Our new approach applies in cases where the amount of labeled data is not sufficient to do supervised estimation of an initial model of reasonable quality, but we have large amounts of unlabeled data and a generative model which can be trained in an unsupervised fashion. We call our training approach "Minimum Error / Maximum Likelihood Training", and we introduce the "EMD" semi-supervised training algorithm to perform the training.

## 4.6 Minimum Error / Maximum Likelihood Training

We extend approximate EM training to perform a new type of training which we call Minimum Error / Maximum Likelihood Training. The intuition behind this approach to semi-supervised training is that we wish to obtain the advantages of both discriminative training (error minimization) and approximate EM (which allows us to estimate

a large numbers of parameters effectively even though we have too few gold standard word alignments to do this in a supervised fashion). We introduce the EMD algorithm, in which discriminative training is used to control the contributions of sub-models (thereby minimizing error), and a procedure similar to one iteration of approximate EM is used to estimate the large number of sub-model parameters, by using steps which increase likelihood.

Intuitively, in approximate EM training for word alignment (Brown et al., 1993), the E-step corresponds to calculating the probability of all alignments according to the current model estimate, while the M-step is the creation of a new model estimate given the probability distribution over alignments calculated in the E-step.

In the E-step ideally all possible alignments should be enumerated and labeled with $p(a|e, f)$, but this is intractable. For the M-step, we would like to count over all possible alignments for each sentence pair, weighted by their probability according to the model estimated at the previous step. Because this is not tractable, we make the assumption that the single assumed Viterbi alignment can be used to update our estimate in the M-step. This approximation is called Viterbi training. Neal and Hinton (1998) analyze approximate EM training and motivate this type of variant.

The basic intuition behind our approach to semi-supervised learning is that we wish to obtain the advantages of both discriminative training and approximate EM. We use

discriminative training to control the contributions of sub-models, which vary in granularity from large numbers of parameters to a single parameter (this can be a single parameter in the original generative model, which we are training discriminatively here). We use a sub-procedure very similar to approximate EM to train the often very large numbers of parameters of the sub-models themselves.

Here is an initial brief outline of the approach. We first determine a decomposition of the generative model into sub-models. We then add additional sub-models which were not in the generative model.

A single iteration of EMD training consists of a step which resembles the E-step in EM, followed by a step which resembles the M-step in EM, followed by a "discriminative step", which we call the D-step. In the step which resembles the E-step, we use the weights $\lambda$ and the estimates of all sub-models (both the sub-models in the generative model and those sub-models which are not in the generative model) to predict alignments for the entire training set. In the step which resembles the M-step, we reestimate the sub-models dependent on the hypothesized alignments (for example, the sub-models which are distributions from the generative model). The D-step estimates the weight vector $\lambda$ which minimizes error. It does this by repeatedly reranking the output of the generative model for a small set of sentences for which we have labels. This completes one iteration of training.

We start the EMD algorithm by estimating the sub-models taken from the generative model by bootstrapping as in the unsupervised case. We then carry out an initial D-step. After this "iteration 0", complete iterations of EMD training are performed, starting with iteration 1.

### 4.6.1 EMD Algorithm

A sketch of the EMD algorithm applied to our extended model is presented in Figure 4.1. Parameters have a superscript $t$ representing their value at iteration $t$. The parameters of the iteration dependent sub-model $m$ at time $t$ are $\theta_m^t$, while the parameters of the sub-model $m$ which is iteration independent is denoted $\theta'_m$. We initialize the algorithm with the gold standard word alignments (labels) of the word alignment discriminative training set, an initial $\lambda$, N, the starting alignments (the final HMM Viterbi alignment), and the parameters of the heuristic sub-models which are iteration independent ($\theta'$). In line 2, we make iteration 0 estimates of the sub-models whose parameters are estimated from the current Viterbi alignment (these are sub-models 1 to $M'$, and include the sub-models based on distributions used in LEAF). In line 3, we run discriminative training using the algorithm from Section 4.2.2. In line 4, we measure the error of the resulting $\lambda$ vector. In the main loop in line 7 we align the full training set (similar to the E-step of EM), and in line 8 we estimate the iteration-dependent sub-models (similar to the

1: **Algorithm** EMD(labels, $\lambda'$, N, starting alignments, $\theta'$)
2: bootstrap $\theta^0_m$ for $m = 1$ to $M'$
3: $\lambda^0$ = Discrim($\theta^0$, $\theta'$, $\lambda'$, labels, N)
4: $e^0$ = Error($\lambda^0$)
5: $t = 1$
6: **loop**
7: align full training set using $\lambda^{t-1}$, $\theta^{t-1}$ and $\theta'$
8: estimate $\theta^t_m$ for $m = 1$ to $M'$
9: $\lambda^t$ = Discrim($\theta^t$, $\theta'$, $\lambda^{t-1}$, labels, N)
10: $e^t$ = Error($\lambda^t$)
11: **if** $e^t >= e^{t-1}$ **then**
12: terminate loop
13: **end if**
14: $t = t + 1$
15: **end loop**
16: return hypothesized alignments of full training set

Figure 4.1: Sketch of the EMD algorithm

M-step of EM). Then we perform discriminative reranking in line 9 and check for convergence in lines 10 and 11 (convergence has been reached if error was not decreased from the previous iteration). The output of the algorithm is hypothesized alignments of the entire training corpus (calculated in line 7).

In the general word alignment problem, the entire search space can not be enumerated, which is the reason we have to do multiple iterations of the loop of the "Discrim" subroutine (which was presented in Section 4.2.2). For each iteration $i$ of the the "Discrim" subroutine, we find a new vector $\lambda$ which then causes us to enumerate a different portion of the search space in Step 1 of the "Discrim" subroutine. We could run this process until we no longer search a different portion of the search space (i.e., we find

no new N best list entries), at which point we would assume we have converged. In practice we stop when the error does not decrease. Note that if EMD is used for a different problem where the entire search space can be explicitly enumerated, the code inside the loop of the "Discrim" algorithm would only need to be executed once per outer loop iteration $t$.

When re-estimating the generative model we use the hypothesized labels for the discriminative training set, rather than the gold standard labels. Otherwise we would overfit the labels on the discriminative set and so we would be unable to continue using predictions to determine good weights.

It is important to emphasize that we are not presenting just a discriminative reranking step but instead a fully integrated approach, taking advantage of the fact that the power of each sub-model changes over the training process (i.e., from iteration to iteration of training). It is the ability to determine how discriminative each sub-model is at each iteration of semi-supervised training and the ability to directly train a few sub-model parameters directly at each iteration of semi-supervised training which gives us performance superior to discriminative reranking (where these two things can only be done once, after the estimation of the generative model).

## 4.7 Previous Work on Semi-Supervised Learning

Previous approaches for using EM for combining labeled and unlabeled data have often been applied to unstructured classification. An initial classifier is learned from labeled data, and then this classifier is used to label unlabeled data with posterior class membership probabilities. EM is then used to improve the initial estimate of posterior class membership probabilities. For labeled data, the probability of the correct class is maximized, and this improves estimates of class membership for the unlabeled data. For unlabeled data the maximum a posteriori (MAP) solution is selected.

There is a large body of work on semi-supervised learning with parameterized distributions that are described by a small number of parameters; we present a few examples. Miller and Uyar (1997) used unlabeled data and EM to augment a mixture of experts. Miller and Browning (2003) used an extension of the EM algorithm for a task modeled as a mixture of Gaussians. Their algorithm is similar to the algorithm we propose in that they extended the EM algorithm by incorporating an additional separate optimization for training a small number of parameters, but they trained these parameters to maximize complete data log likelihood rather than the final performance criterion.

There has also been some work on semi-supervised learning when a much larger number of parameters must be estimated. Nigam et al. (2000) addressed a text classification task where each class is modeled as multiple mixtures over the entire vocabulary.

They estimated a Naive Bayes classifier over the labeled data and used it to provide initial MAP estimates for unlabeled documents. They then ran EM as described above. They introduced a single mixing parameter to attempt to control problems with the estimates from the unlabeled data washing out the estimates from the labeled data. Their approach would not work if applied to our scenario as the number of labeled examples is small, so the initial labellings of the unlabeled data would be very poor, causing the "overconfident pseudo-labeling problem" we already mentioned in Section 4.5.

Callison-Burch et al. (2004) performed a preliminary study of the issue of semi-supervised training for word alignment. They addressed their lack of manually annotated data by using automatically annotated data as a replacement for human annotated data and looking at the effect of semi-supervised learning on both AER and BLEU, following the work of Nigam et al. (2000). However, their simulated supervised data was annotated using GIZA++, which, as we have already shown, can be further improved substantially, so we do not believe that they succeeded in realistically simulating having large amounts of manually annotated data. However, their experiments on combining higher and low quality automatically generated alignments did result in an important finding. They showed that it is important to ensure that the larger amount of low quality annotations do not "wash out" the parameters estimated from the higher quality annotations, which is an insight we will use in the experimental section.

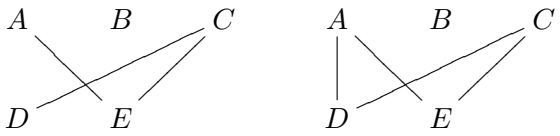


Figure 4.2: Two alignments with the same translational correspondence

Two approaches that are more similar in spirit to our work involve the use of labels in reinforcement learning and the use of labels in clustering. Ivanov et al. (2001) used discriminative training in a reinforcement learning context in a similar way to our adding of a discriminative training step to an unsupervised context. A large body of work uses semi-supervised learning for clustering by imposing constraints on the clusters. Basu et al. (2004) is a good example, where the system was supplied with lists of pairs of instances labeled as belonging to the same or different clusters. Our work can be motivated in a similar fashion to theirs, but the details are quite different. We are solving a difficult structured prediction problem which involves a search over bigraphs for each parallel sentence pair.

## 4.8 Experiments

We perform experiments on the two large alignments tasks from Chapter 3, for Arabic/English and French/English data sets. Statistics for these sets are shown in Table 3.3 on page 74. All of the data used is available from the Linguistic Data Consortium

except for the French/English gold standard alignments which are available from the authors.

We showed that F-Measure is effective in predicting BLEU in Chapter 2. Therefore, we use $1 -$ F-Measure$(\alpha)$ as our error criterion in discriminative training. We established that it is important to tune $\alpha$ (the trade-off between Precision and Recall) to maximize performance.

We remind the reader of the problem we discovered in Chapter 2, which is that two alignments which have the same translational correspondence can have different F-Measures. An example is shown in Figure 4.2. To overcome this problem we fully interlinked the transitive closure of the undirected bigraph formed by each alignment hypothesized by our baseline alignment systems[2]. This operation maps the alignment shown to the left in Figure 4.2 to the alignment shown to the right. Recall that this operation does not change the collection of phrases or rules extracted from a hypothesized alignment.

The best settings of $\alpha$ were $\alpha = 0.1$ for the Arabic/English task and $\alpha = 0.4$ for the French/English task, , see Chapter 2 for details of the process used to choose these constants.

[2]All of the gold standard alignments were fully interlinked as distributed. We did not modify the gold standard alignments.

| | | | |
|---|---|---|---|
| 1 | $g(\chi_i \vert e_i)$ source word type | 9 | $d_2(\triangle j \vert \text{class}_f(f_j))$ movement for left-most target non-head word |
| 2 | $w_{-1}(i - \mu_i \vert \text{class}_e(e_i))$ choosing a head word | 10 | $d_{>2}(\triangle j \vert \text{class}_f(f_j))$ movement for subsequent target non-head words |
| 3 | $t_1(f_j \vert e_i)$ head word translation | 11 | $t(f_j \vert e_i)$ translation without dependency on word-type |
| 4 | $s(\psi_i \vert e_i, \gamma_i)$ $\psi_i$ is number of words in target cept | 12 | $t(f_j \vert e_i)$ translation table from final HMM iteration |
| 5 | $s_0(\psi_0 \vert \sum_i \psi_i)$ number of unaligned target words | 13 | $s(\psi_i \vert \gamma_i)$ target cept size without dependency on source head word $e$ |
| 6 | $t_0(f_j)$ identity of unaligned target words | 14 | $s(\psi_i \vert e_i)$ target cept size without dependency on $\gamma_i$ |
| 7 | $t_{>1}(f_j \vert e_i, \text{class}_h(\tau_{i1}))$ non-head word translation | 15 | target spurious word penalty |
| 8 | $d_1(\triangle j \vert \text{class}_e(e_\rho), \text{class}_f(f_j))$ movement for target head words | 16-30 | (same features, other direction) |

Table 4.2: Sub-models used together with the EMD algorithm

### 4.8.1 Evaluating EMD+LEAF

We present an experiment which evaluates the efficacy of the EMD training algorithm when applied to a log-linear model. We decompose LEAF, presented in Section 3.2, in both translation directions to provide the initial feature functions for the log-linear model, features 1 to 10 and 16 to 25 in Table 4.2.

To provide additional robustness, we use back-offs for the translation decisions (features 11 and 26), the HMM translation tables (features 12 and 27) and back-offs for the target cept size distributions (features 13, 14, 28 and 29 in Table 4.2). We also use heuristics which directly control the number of unaligned words we generate (features 15 and 30 in Table 4.2), which allows us to control the trade-off between Precision and Recall which is required to optimize any particular $\alpha$ used with F-Measure($\alpha$).

We perform one main comparison, which is of semi-supervised systems. This is also what we will use to produce alignments for evaluating SMT performance. We compare semi-supervised LEAF with our previous state of the art semi-supervised system (Fraser & Marcu, 2006) which also uses the EMD algorithm but separately optimizes 1-to-N and M-to-1 translation performance using sub-models derived from Model 4 and a larger number of heuristic models than are used with LEAF. We perform translation experiments on the alignments generated using semi-supervised training to verify that the improvements in F-Measure result in increases in BLEU. Note that the timings for the first E-Step of the French/English experiments are presented in Appendix C.1. The current (unoptimized) LEAF search implementation is slow, speeding up search is discussed in the same appendix.

In order to have the results in a single table, we also compare the unsupervised LEAF system with GIZA++ Model 4. This gives an idea as to the performance of the unsupervised model, and is a repeat of the results from Section 3.5. The reader is referred there for further explanation.

To build all alignment systems, we start with 5 iterations of Model 1 followed by 4 iterations of HMM (Vogel et al., 1996), as implemented in GIZA++ (Och & Ney, 2003), and use the final iteration of HMM to perform the bootstrap. To generate the final output for all non-LEAF systems, we take the best performing of the “Union”, “Refined” and “Intersection” symmetrization heuristics (Och & Ney, 2003) to combine

the 1-to-N and M-to-1 directions resulting in a M-to-N non-consecutive alignment. Because these systems do not output fully linked alignments, we fully link the resulting alignments. Once again, the reader should recall that this does not change the set of rules or phrases that can be extracted using an alignment.

Results for the experiments on the French/English data set are shown in Table 4.3. We ran GIZA++ for four iterations of Model 4 and used the "Refined" heuristic (line 1). We ran the baseline semi-supervised system for two iterations (line 2), and in contrast with Fraser and Marcu (2006) we found that the best symmetrization heuristic for this system was "Union", which is most likely due to our use of fully linked alignments. We observe that LEAF unsupervised (line 3) is competitive with GIZA++ (line 1), and is in fact competitive with the baseline semi-supervised result (line 2). We ran the LEAF semi-supervised system for two iterations (line 4). The best result is the LEAF semi-supervised system, with a gain of 1.8 F-Measure over the LEAF unsupervised system and a gain of 2.8 F-Measure over GIZA++.

For French/English translation we use a state of the art phrase-based MT system similar to those of Och and Ney (2004) and Koehn et al. (2003). The translation test data is described in Table 3.5.1. We use two trigram language models, one built using the English portion of the training data and the other built using additional English news data. The BLEU scores reported are calculated using lowercased and tokenized data. For semi-supervised LEAF the gain of 0.46 BLEU over the semi-supervised baseline is

| System | French/English | | Arabic/English | |
|---|---|---|---|---|
| | F ($\alpha = 0.4$) | BLEU | F ($\alpha = 0.1$) | BLEU |
| GIZA++ | 73.5 | 30.63 | 75.8 | 51.55 |
| Fraser and Marcu (2006) | 74.1 | 31.40 | 79.1 | 52.89 |
| LEAF unsupervised | 74.5 | | 72.3 | |
| LEAF semi-supervised | 76.3 | 31.86 | 84.5 | 54.34 |

Table 4.3: Experimental Results

not statistically significant (a gain of 0.78 BLEU would be required), but LEAF semi-supervised compared with GIZA++ is significant, with a gain of 1.23 BLEU. We note that a gain of 1.23 BLEU shows a large gain in translation quality over that obtained using GIZA++ because for the French/English task BLEU is calculated using only a single reference (a gain of 1.23 BLEU using a single reference is a larger gain than a gain of 1.23 BLEU when using four references).

Results for the Arabic/English data set are also shown in Table 4.3. We used a large gold standard word alignment set available from the LDC. We ran GIZA++ for four iterations of Model 4 and used the "Union" heuristic. We compare GIZA++ (line 1) with one iteration of the unsupervised LEAF model (line 3). The unsupervised LEAF system is worse than four iterations of GIZA++ Model 4. We believe that the features in LEAF are too high dimensional to use for the Arabic/English task without the back-offs available in the semi-supervised models. The baseline semi-supervised system (line 2) was run for three iterations and the resulting alignments were combined with the "Union" heuristic. We ran the LEAF semi-supervised system for two iterations. The

best result is the LEAF semi-supervised system (line 4), with a gain of 5.4 F-Measure over the baseline semi-supervised system and a gain of 8.7 F-Measure over GIZA++.

For Arabic/English translation we train the state of the art hierarchical model Hiero (Chiang, 2005) using our Viterbi alignments. The translation test data used is described in Table 3.5.1. We use two trigram language models, one built using the English portion of the training data and the other built using additional English news data. The test set is from the NIST 2005 translation task. LEAF had the best performance scoring 1.43 BLEU better than the baseline semi-supervised system and scoring 2.79 BLEU better than GIZA++, both of which are statistically significant.

The success of training our new log-linear model, based on sub-models derived from LEAF, to minimize the $1-$F-Measure$(\alpha)$ error criterion using the semi-supervised EMD training algorithm combines the main contributions of this thesis. The BLEU score increases achieved by this system are large for both tasks[3]. We now have a principled model over the alignment structure in which we are interested, and we can obtain a posterior probability distribution over likely alignments rather than being restricted to heuristically combining the 1-best predictions of a 1-to-N and M-to-1 model as was previously done, which will enable new directions for future research. We have shown that the predictions of our new model substantially improve state of the art machine translations systems on some of the largest, most challenging, data sets available.

[3]We remind the reader that the French/English result is based on BLEU calculated using only a single reference, for which a gain of 1.2 BLEU% is large.

### 4.8.2 Giving GIZA++ Access to Human Annotated Alignments

We performed an additional experiment for the French/English alignment task aimed at understanding the potential contribution of the word aligned data without the new model and training algorithm. Like Ittycheriah and Roukos (2005), we converted the alignment discriminative training corpus links into a special corpus where the parallel "sentences" consist only of the single English and French word involved in each link. We found that the information in the links was "washed out" by the rest of the data and resulted in no change in the alignment test set's F-Measure. Callison-Burch et al. (2004) showed in their work on combining alignments of lower and higher quality that the alignments of higher quality should be given a much higher weight than the lower quality alignments. Using this insight, we found that adding 10,000 copies of this special corpus to our training data resulted in the highest alignment test set gain, which was a small gain of $0.3$ F-Measure. This result suggests that while the link information is directly useful for improving F-Measure, our semi-supervised training method is producing much larger improvements.

### 4.8.3 Integrating an Arabic Name Transliteration Model

We report in this section on integrating an Arabic Name transliteration model, developed by Ulf Hermjakob. This model reads parallel sentences and outputs any likely

transliteration matches between a single Arabic token and one or more English tokens along with a confidence score.

The interesting aspect of integrating this as a sub-model is that it can not be directly integrated as a phrase to phrase matching. This is because even when there is a likely transliteration match, this match often does not fully account for the complete translational correspondence involved.

For instance, suppose that in the Arabic sentence of a parallel Arabic/English sentence pair the Arabic word “Mohammed” occurs. If the English word “Mohammed” occurs twice in the English sentence, a transliteration model is unable to determine which one to match or whether to match both. We solve this problem by providing a constraint on the alignment. We say that the alignment must align at least one of the English “Mohammed” tokens with the Arabic “Mohammed” token, or a penalty is paid. We train a penalty sub-model in the log-linear model which pays a fixed cost for violating such constraints, which has the effect of setting a decrease in cost which must be obtained from other sub-models in order for an alignment in which the constraint is violated to more probable than one obeying the constraint. Note that this type of “OR” constraint would be very difficult to specify in the LEAF generative story.

A similar case occurs where the combination of an English transliteration of an Arabic content word and one or more English function words should be aligned as a unit to the single Arabic content word. The transliteration model has a limited ability to

determine where English function words should be aligned, but for more complicated decisions the decision requires knowledge which can be found in the other sub-models which can determine whether alignment geometry is probable, likely non-head words to attach to the English head word, etc. This is again implemented as a constraint, which is placed on the alignment of the content word.

Adding constraints determined by the transliteration package lead to an increase of 0.2 F-Measure over the system without these constraints. The fit on the development corpus was 0.5 F-Measure better, indicating that some overfitting likely occurred.

The transliteration model only suggests a constraint for a few words in each of roughly one quarter of the parallel sentences in our training corpus. The sub-model added a constant for each constraint violation. We also tried using one minus the confidence score as the penalty which did not improve performance.

The successful integration of a feature of this type shows that our approach is not limited to sub-models which are similar to those in the generative story but can in fact be used with any sub-model which can be scored over a hypothesized alignment of a parallel sentence pair. We believe that improving the reliability of the confidence score and decreasing overfitting will increase the performance obtained by adding this sub-model further.

### 4.8.4 Integrating Supervised Sub-models

The EMD algorithm can also integrate supervised knowledge. We recently obtained a larger hand aligned alignment set from LDC for Arabic/English. After eliminating possible overlap with our discriminative training and test sets, there were hand generated alignments for 25,930 new sentences. We decided to estimate two small supervised sub-models directly from this data and add these sub-models to the EMD+LEAF system.

We estimated translation tables directly from this data. There were about 230,000 entries in the translation tables, which are tables containing an English word, an Arabic word, and a probability. This is a low number of parameters. For instance, compare this with the HMM translation tables, where each table has about 34,000,000 entries, (these tables are features 12 and 27 in the semi-supervised model, see Table 4.2).

We added the two supervised translation table sub-models to our baseline LEAF+EMD alignment system. This lead to an increase of 1.8 F-Measure over a system without this supervised knowledge. This shows that it is possible to easily integrate supervised knowledge into the system.

## 4.9 Discussion

The literature on semi-supervised learning generally addresses how to augment supervised learning tasks with unlabeled data. Here we augment an unsupervised learning task with labeled data. This is useful in a wide diversity of tasks where we do not have enough unlabeled data for supervised estimation of an initial model.

We have presented an algorithm applicable in the case that we have few labels and a generative model with acceptable performance when trained in an unsupervised fashion. We determine a decomposition of the generative model into sub-models and then reinterpret these sub-models as being combined into a log-linear model. We can add additional sub-models which were not in the original generative story, and we use this to add both backed off forms of the sub-models derived from the original generative story, and heuristic sub-models which are not directly related to the original generative story.

It is important to note that with this training algorithm we are not taking steps to strictly maximize likelihood, even though the vast majority of parameters are estimated in the likelihood maximization framework. Instead we are finding local maxima of likelihood which are better with respect to the final performance criterion. These are better than other reachable maxima with respect to the final performance criterion, but they could possibly be worse with respect to likelihood under the original generative model.

We have shown that the reinterpretation of our new model as a log-linear model and the derivation of a semi-supervised training algorithm which can be used to train it is an excellent way forward to integrating knowledge sources which could not be captured in the original generative model.

The semi-supervised learning literature generally addresses augmenting supervised learning tasks with unlabeled data (Seeger, 2000). In contrast, we augmented an unsupervised learning task with labeled data. We hope that Minimum Error / Maximum Likelihood training using the EMD algorithm can be used for a wide diversity of tasks where there is not enough labeled data to allow supervised estimation of an initial model of reasonable quality.

## 4.10 Summary

We began this chapter by redefining LEAF as a log-linear model. We showed how to discriminatively rerank N best lists which are taken from this model. We then generalized this to a semi-supervised training algorithm called "EMD" which implements "Minimum Error / Maximum Likelihood" training. We trained EMD using the original sub-models of LEAF along with more robust backed off sub-models and heuristically derived sub-models which directly control the trade-off between Precision and Recall.

The EMD algorithm, when coupled with features derived from our LEAF model and trained to maximize F-Measure, leads to increases between 3 and 9 F-score points

in alignment accuracy and 1.2 and 2.8 BLEU points in translation accuracy over strong French/English and Arabic/English baselines. This strongly validates all three main contributions of the thesis. We additionally performed experiments showing that we can add sub-models which are very different from those derived from LEAF.

## 4.11 Research Contribution

We developed an effective semi-supervised training algorithm for automatic word alignment which is capable of using manually annotated data and of integrating sub-models which are not in our original generative model.

# Chapter 5

# Conclusion

We present the contributions of the thesis, discuss lessons learned, and then present a section combining shortcomings and suggested future work.

## 5.1 Contributions

1. We have found a new method for automatically measuring alignment quality using an unbalanced F-Measure metric (Fraser & Marcu, 2007b), which has a good correlation with BLEU. We have experimentally validated that this metric adequately measures alignment quality for the translation task.

2. We have designed a new statistical model for word alignment, called LEAF (Fraser & Marcu, 2007a), which directly models the word alignment problem without making unreasonable assumptions about the structure of the resulting

alignments. When LEAF is trained in an unsupervised fashion using approximate EM, it is comparable with our baseline. Unlike our baseline, unsupervised LEAF does not require the use of heuristics to generate the final alignment which is used to build a SMT system. The LEAF model can be decomposed to provide rich sub-models which can be used in a log-linear model for semi-supervised training.

3. We have developed a semi-supervised training algorithm, the EMD algorithm (Fraser & Marcu, 2006), which automatically takes advantage of whatever quantity of manually annotated data can be obtained. This algorithm allows for the introduction of new knowledge sources with minimal effort. We formulated a new log-linear model using the original sub-models of LEAF along with more robust backed off sub-models and two heuristically derived sub-models which directly control the important trade-off between Precision and Recall. We applied the EMD algorithm to train this model using a loss function derived from our unbalanced F-Measure metric. The EMD algorithm, when coupled with sub-models derived from our LEAF model, leads to increases between 3 and 9 F-score points in alignment accuracy and 1.2 and 2.8 BLEU points in translation accuracy over strong French/English and Arabic/English baselines.

## 5.2 Lessons Learned

### 5.2.1 Quality

The most widely used error metric in word alignment, Alignment Error Rate, (AER) (Och & Ney, 2003) is not correctly derived from F-Measure and should not be used.

The trade-off between Precision and Recall is very important. We have shown that the setting of the parameter $\alpha$, controlling this trade-off, varies with the task.

Using fully connected alignments is important, see Figure 2.1 on Page 29. Without using fully connected components we have unnecessary ambiguity where two alignments which have the same translational correspondences have different scores according to most intrinsic metrics of quality.

Extrinsic evaluation is important. Some word alignment research directed towards minimizing AER, such as research on 1-to-1 alignment models, is not useful for increasing translation performance. This is an important lesson for Natural Language Processing systems which are not generally extrinsically validated. An example is statistical parsing where, at least until recently, a higher priority has been assigned to increasing performance on the Section 23 test set of the Penn Treebank than to ensuring robust performance in clearly identified tasks. The latter would almost always

involve parsing sentences which are drawn from a distribution which is not well correlated with that of the Penn Treebank, and gains in the robustness required to do this accurately may not be well correlated with small gains on Section 23.

### 5.2.2 Modeling

M-to-N discontiguous alignments allow us to learn the translational correspondences we are interested in. These are the most general correspondences which can be used by current hierarchical translation systems such as Hiero (Chiang, 2005) and GHKM (Galley et al., 2006). Even phrase-based (consecutive word) SMT models can benefit from alignments which do not make the consecutive word alignment structure assumption.

The quality of search is an important consideration where we are unable to do tractable inference. It is important to both directly control search errors and directly control the time taken.

The beam decoding algorithm, widely used in phrase-based decoders, does not work for word alignment models with complex structure. Unlike phrase-based decoding, left-to-right hypothesis extension using a beam decoder is unlikely to be effective because in word alignment reordering is not limited to a small local window and so the necessary beam would be very large. We are not aware of admissible or inadmissible search heuristics which have been shown to be effective when used in conjunction with a search algorithm similar to A* search for a model predicting over a structure like ours.

The problem with the blow-up in parameter space involved in phrase-based models such as the Joint model (Marcu & Wong, 2002) is partially solved by using the head word structure. In particular, this appears to be a realistic assumption given the amount of data we now have, and we also have a straight-forward path to increase the richness of the sub-models, in response to additional training data, by simply reducing reliance on word classes and further relaxing conditional independence assumptions. The M-to-N discontiguous alignment structure using the head word assumption is also faster to search than a pure phrase-based structure as the translation dependencies on one side are only dependent on the head word on the other side (and $\gamma$ which is a flag indicating whether the cept on the other side contains just one word). In phrase based approaches translational correspondence is calculated using the full identity of both cepts. The decomposition of costs using the head word assumption means that adding a non-head word to a head word is an operation which incurs additional cost but does not cause all other costs incurred by that cept to be reevaluated. In phrase-based models any change to a cept causes all costs to be reevaluated.

### 5.2.3 Semi-supervised Training

Combining discriminative training in a loop with steps derived from EM which increase likelihood is an effective approach to semi-supervised training of models which were traditionally trained in an unsupervised fashion using EM.

Symmetrization heuristics are surprisingly powerful, but are no longer required for LEAF, which directly models the desired alignment structure. We were initially surprised that the predictions of the symmetrization heuristics were no longer useful as a sub-model, but in retrospect it makes sense as they are the product of simple rules which are effectively subsumed in the LEAF model.

Deriving an appropriate training criterion is important. As we showed in Chapter 2 AER is not a good training criterion, which shows why our initial experiments in discriminative training, (Fraser & Marcu, 2005), failed to produce an improvement in BLEU.

Backing off the rich features of LEAF is important, particularly for difficult language pairs like Arabic/English, and combining the original rich feature with a backed off version in a log-linear model is an effective way of doing this.

Directly tuning the trade-off between Precision and Recall is important when working with F-Measure. This has an analogue in translation, which is the optimization of the BLEU length penalty (Koehn et al., 2003), which is required to obtain good performance using BLEU.

Scoring full hypotheses allows for the integration of very rich features scored over the full alignment, a subject we have only scratched the surface of with the integration of the name transliteration feature.

The search performance dramatically affects the performance of our discriminative algorithm. We have found that search performance is much more important during the D-step than it is when predicting Viterbi alignments for the entire training corpus. Fortunately we only have to execute search for the discriminative development set during discriminative training. For instance, we search for the Viterbi alignment for only the 1,000 sentences in the development set for the Arabic/English task (the search is performed once for each iteration of the loop inside of the D-step, see Figure 4.1). Because of this we can spend a significantly longer time on each sentence pair during discriminative training than when we perform the E-step (which requires finding the Viterbi alignment of 6.6 million sentence pairs in the Arabic/English case).

## 5.3 Shortcomings and Future Work

### 5.3.1 Problem Definition: What is a Word?

We have implicitly specified that a word is a space-separated token output by a tokenizer. The tokenizer's primary purpose is to separate punctuation from words. The tokenizer additionally performs light deterministic processing of morphological phenomena. For instance, the French tokenizer we use separates obvious clitics from the words they are attached to (e.g. "n'est" is mapped to "ne est") and maps masculine and feminine articles to a single token (which is acceptable for translation to English

which does not make this distinction). However this approach is too simple for many language pairs.

The LEAF generative story generalizes well to the case that information in one language is expressed lexically and not present in an easily accessible fashion in the other language. For instance, for the application of Chinese/English machine translation LEAF's "head-word" concept seems to work well. An English phrase such as "the man" is often translated as a single Chinese word meaning "man", while the definiteness of this word is usually marked by syntactic phenomena which would be difficult to model. A good LEAF alignment would be a head-word link between English "man" and the Chinese word for "man", and then an association from English non-head word "the" to English head word "man". The $g$ distribution in LEAF is capable of modeling that the word "the" has a high probability of being a non-head word, while the $w_{-1}$ distribution can model that non-head words in the word class which "the" is in have a high probability of being associated with a head word which is one word to the right.

Chinese (and other Asian languages such as Japanese) additionally require word segmentation, which separates short sequences of Chinese characters into "words" (this is because Chinese is written without the use of spaces to separate words). Automatic word segmentation is itself an active area of research. A Chinese word segmenter is typically trained in a supervised fashion from a gold standard segmentation specified by human annotators, but it is not been carefully studied whether existing segmentations

are a good choice for machine translation purposes. In fact, it may be possible to create a new generative story by adding a few steps to the LEAF generative story which allow the Chinese word segmentation to be modeled simultaneously with word alignment, rather than handled as a preprocess as is currently done. This would have the interesting effect of allowing word segmentation choices to be informed by the English words in the parallel text. Most likely an initial segmentation (or segmentation knowledge source) would need to be initialized using supervised knowledge, but the segmentation could then be allowed to vary during the alignment process, and this might determine a final segmentation which is more useful for translation than existing segmentations.

Unfortunately, the LEAF generative story does not model the information systematically present in "pieces" of words (e.g. morphological phenomena, including particularly clitics). Such generalization would require a source of morphological knowledge. For instance, consider again English "the man", but this time consider how it should be aligned with Arabic. English "the man" might be aligned with the single Arabic token "al-rajul", where the prefix "al-" is "the", and "rajul" means "man". Here again the $g$ distribution in LEAF is capable of modeling that words like "the" have a high probability of being non-head words; again, the $w_{-1}$ distribution can model that non-head words in the word class of "the" are often associated with head words one word to the right. But LEAF can not learn that the "al-" in this case indicates that it is more likely that "the" should be in the English cept aligned with "al-rajul". Modeling this

in a language independent fashion would be difficult. Ittycheriah and Roukos (2005) defined sub-models which model this type of information for the Arabic/English word pair case and showed that this is effective. We could similarly define language pair specific sub-models to do this. However, we would be more interested in finding a general framework to solve this problem. Such a framework would ideally be language independent, but might require supervised training data (in the same way that integrating Chinese segmentation might require access to a supervised knowledge source, as we already discussed). We would be interested in developing a language independent extension of the LEAF generative story which is able to consider phenomena like the "al-" in "al-rajul" (and possibly align such morphemes separately), but we recognize that this is both conceptually and computationally difficult without access to very highly accurate sources of morphological knowledge.

### 5.3.2 Quality

One shortcoming of our work on quality metrics is that we have provided a metric with a tunable parameter. This necessitates experimentation to determine how to evaluate with each new task. We would be interested in understanding the dependency of the $\alpha$ parameter more fully. For instance, we could study whether there is something about the language pairs involved, the quality and style of the gold standard annotation, or

even the quantity of training data which helps to explain why a particular $\alpha$ setting works best.

Ideally we would like to derive a metric which does not have a tunable parameter but has the same performance as unbalanced F-Measure does when $\alpha$ is appropriately tuned. CPER (Ayan & Dorr, 2006a) is an interesting step in this direction. CPER calculates balanced F-Measure over the phrase pairs extracted from a hypothesized alignment (these are the same phrase pairs as are extracted for use in the translation model of a phrase-based MT system), comparing them with the phrase pairs extracted from a gold standard alignment. Unfortunately, CPER has not been shown to predict MT performance. It seems likely to us that there should be a trade-off between Precision and Recall in comparing phrases extracted as well, but possibly this trade-off will be less important than in the case of word links. We would be satisfied if we were able to use a single $\alpha$ parameter in conjunction with a CPER-like metric if $\alpha$ were constant for all of our tasks.

Another shortcoming of our work is that we only tested the BLEU metric. The BLEU metric shows that is likely that our noise and oracle models, used for artificially degrading and improving alignments, produce regular changes in the quality of machine translation systems built from these alignments, but we could obtain even stronger evidence. Ideally we would like to use human annotators to judge the output of MT systems built using the alignments, but this would be prohibitively expensive and is

probably not necessary in this case. Instead, METEOR (Banerjee & Lavie, 2005) is a promising automatic metric which we would be interested in trying as it has been shown to have better correlation with human judgments than BLEU.

Our work on quality is dependent on measuring the quality of a single predicted alignment, such as the Viterbi alignment of the LEAF model. However, there are approaches to building MT systems which are trying to utilize the full distribution over alignments rather than the most likely single alignment. As this body of work matures, we would be interested in deriving a quality criterion for a distribution over alignments which is finer grained than simply taking the most likely prediction and scoring it. This new quality criterion should allow us to evaluate the quality of the entire distribution.

### 5.3.3 Modeling

One large disadvantage of the LEAF model is the intractability of exact search. Model 4 has the same problem. We need to solve search problems during both parameter estimation and prediction of the final Viterbi alignment. As we have discussed previously, existing models with tractable exact search make unrealistic assumptions about alignment structure which do not model the word alignment problem with sufficient fidelity. We have defined a local search algorithm which results in good F-Measure scores, by taking steps to apply some of the knowledge gained by the research community in solving problems such as the Traveling Salesman Problem in our implementation of a

restarting “Tabu” search (Glover, 1986). However, our current implementation is very slow (see Appendix C.1 for detailed timings and a discussion of how to program a faster implementation). We are also hopeful that we could use a dynamic programming approach which would consider many more alignments (see Appendix C.2).

Another disadvantage of the LEAF model is the Viterbi approximation used to carry out the M-step. In previous experiments using GIZA++ we have found that using the Viterbi assumption is usually not worse than using the “neighborhood” assumption, which involves calculating the probabilities of alignments which are one search operation away from the Viterbi alignment. However, there is reason to believe that this might not be the best we could do. In our work with LEAF we have significantly reduced search errors, which means that the alignments we find are of higher quality. It is likely that the N best lists we generate are a better approximation of the search space than the neighborhood of the Viterbi used by GIZA++. In the short term, it would be interesting to try estimating LEAF using a normalized N best list of a large size similar to those generated during the D-step (but in this case calculated over the entire training corpus). In the longer term, it would be interesting to estimate LEAF by solving the alignment problem such that very large N best lists or an alternative efficient representation of many hypotheses can be used.

One aspect of the LEAF model we have not fully investigated is the use of word classes. We use source word classes derived using a greedy maximization of the probability of the monolingual source corpus (Och, 1999), and follow the same procedure to derive word classes for the monolingual target corpus. These are the same word classes as are used in our baseline. We would be interesting in conducting a study to see if better word classes, for instance derived from Part-of-Speech tags, might help the performance of LEAF.

### 5.3.4 Semi-supervised Training

Our approach to semi-supervised training, Minimum Error / Maximum Likelihood training using the EMD algorithm, has been shown to work well, but it could be further improved. We would be interested in conducting studies to determine the point at which the current algorithm begins to overfit the discriminative training set. It would also be helpful to determine at which point adding additional sub-models begins to tax the current optimization's ability to find a local maxima reasonably close to the global maxima.

A closely related problem is the problem of feature selection. In our current implementation several of the features receive very low scores and sometimes the 1 best choice (taken by rescoring the final hypothesis list from the D-step) is not changed by removing these features. A principled approach to feature selection would mark

these features as ineffective earlier in the process and this might systematically result in convergence to better solutions for the discriminative training problem.

One obvious area of improvement for our semi-supervised alignment model is to use language specific sub-models as we already mentioned. In particular, interesting work has been done for morphology in connection with word alignment. Corston-Oliver and Gamon (2004) describes an approach for normalizing the inflectional morphology of German and English to gain an improvement in alignment quality measured by AER. We documented a simple approximative stemming algorithm, (Fraser & Marcu, 2005), which was used to gain an improvement in AER. Niessen and Ney (2004) provides an interesting approach to integrating morphology in word alignment by interpolating lemma and inflected word probabilities in a principled fashion. The IBM research group has used Model 1 training combined with sophisticated morphological segmentation of Arabic to train Arabic/English word alignments (Lee, 2004), and more recently defined a discriminative word alignment model specifically for Arabic integrating morphological components (Ittycheriah & Roukos, 2005). These works and several others point to the possibility of integrating morphological modeling with word alignment. One could integrate features either just into the word alignment model, or possibly into both the word alignment model and the translation model in a coordinated fashion.

We are also interested in the integration of more powerful sub-models which can be drawn from other areas covered in the natural language processing literature. We suggest three examples here. Drabek and Yarowsky (2004) showed that syntactic rules can be used to reorder the corpus so as to decrease problems in aligning syntactic clause level phenomena, and Collins et al. (2005) has generalized this approach further. Our model is likely to benefit from the use of dependency parses to help determine likely head word relationships in a manner similar to work reported by Cherry and Lin (2003), but instead implemented as a sub-model added to semi-supervised LEAF. Work on determining multi-word units, which is often done using unsupervised models, may provide interesting features for helping to inform which words might be grouped together as a translational unit, though this decision is ultimately a bilingual decision which will be made differently for different language pairs (e.g. the English words grouped together would differ for the English/Arabic and English/German cases). Work of this type can be easily integrated into our framework as we always score complete hypotheses, and so no limitations requiring the decomposing of features over small pieces of the alignment are necessary.

Finally, we would like to apply the EMD algorithm to problems outside of word alignment. There is a tremendous interest in algorithms which work well with very small quantities of labeled data and larger quantities of unlabeled data. EMD solves this problem, but in its current formulation is tied to the word alignment problem. We would

be interested in providing a more general formulation of EMD. Another application of EMD, perhaps outside of the area of natural language processing, is an opportunity we would be very interested in pursuing.

### 5.3.5 Using Word Alignment

We would be interested in applying the EMD algorithm in conjunction with LEAF to generate alignments for applications other than MT. Two obvious applications which come to mind are Cross-Lingual Information Retrieval (CLIR) (Hiemstra & de Jong, 1999; Xu et al., 2001; Fraser et al., 2002) and paraphrasing (Pang et al., 2003; Quirk et al., 2004; Bannard & Callison-Burch, 2005). It would be interesting to see if the F-Measure criterion derived for translation tasks is useful for these tasks as well. Our intuition tells us it that it should be, but this must be empirically verified. We would need to calculate an appropriate $\alpha$ for each task. As an example, we were interested to observe that in work by Riezler et al. (2007) the authors reported that they needed to manually increase the number of NULL alignments on one side of a specialized corpus they were aligning for use in query expansion. We expect that these sort of trade-offs could be handled automatically in our framework by providing a small number of gold standard word alignments and appropriately adjusting $\alpha$. It would also be very interesting to try using alignments generated following our approach to build resources

for CLIR and paraphrasing, and these applications might provide another source of extrinsic validation for our work.

We also envision modifying LEAF's generative story to better model other applications. For instance, LEAF could be modified to directly model the problem of summarization, in a fashion similar to work by Daumé III and Marcu (2005). This requires a generative story which allows large amounts of deletion in aligning the document to the summary. A similar problem is the modeling of the generation of closed captions for television.

The present best practice of extracting translation rules (or phrase pairs in a phrase-based SMT system) from a single alignment (such as the LEAF Viterbi alignment) is well established. But as we discussed in Section 3.6.7 research has begun into estimating the translation model from a distribution over alignments. A first approximation of this approach might be to estimate rules from the N best lists we can currently generate, weighted by the posterior probability of the alignment. We might also want to "second-guess" the extraction of phrase-pairs from the final LEAF Viterbi alignment in a fashion similar to the work of Deng and Byrne (2005). Given a new test set, they used their alignment model to try to determine probable translations for phrases which occurred in the training data but were aligned in such a way that extracting a translation rule was impossible. This revisiting of the alignment given a test set is a form of inexpensive transductive learning. As work in the area of estimating from more general

output distributions than the Viterbi alignment progresses, we envision the modification of LEAF to output a distribution over alignments which assigns non-zero probabilities to a large portion of the probable alignments. This will necessitate the modification of translation systems to estimate rules from this distribution.

A closely related advance would be to refine LEAF itself into a translation model. The success of the Hiero hierarchical translation model (Chiang, 2005) suggests that this would be possible. However this would be an ambitious research program as we would need to create a decoder integrating language modeling capability, and most likely we would have to create a very different search algorithm. We would also need to add new sub-models to the model to score translations. In particular it would be important to allow the model to memorize more of the context than is necessary in word alignment. A less ambitious project which could be used as a stepping stone towards this final goal would be to score the LEAF alignment model as a feature in a hierarchical decoder in a similar fashion to the "lexical smoothing" (scoring of the alignment links used to generate translation rules) already implemented in Hiero, or even as "lexical smoothing" in a phrase-based decoder (particularly if it were a more general phrase-based decoder which supported gaps in the phrases).

# Bibliography


Ahrenberg, L., Merkel, M., Sagvall, H. A., & Tiedemann, J. (2000). Evaluation of word alignment systems. *Second International Conference on Language Resources and Evaluation (LREC)* (pp. 1255–1261). Athens, Greece.

Al-Onaizan, Yaser, Curin, Jan, Jahr, Michael, Knight, Kevin, Lafferty, John D., Melamed, I. Dan, Purdy, David, Och, Franz J., Smith, Noah A., & Yarowsky, David (1999). Statistical machine translation, final report, JHU workshop.

Ayan, Necip Fazil, & Dorr, Bonnie J. (2006a). Going beyond AER: An extensive analysis of word alignments and their impact on MT. *Proceedings of COLING-ACL* (pp. 9–16). Sydney, Australia.

Ayan, Necip Fazil, & Dorr, Bonnie J. (2006b). A maximum entropy approach to combining word alignments. *Proceedings of HLT-NAACL* (pp. 96–103). New York.

Ayan, Necip Fazil, Dorr, Bonnie J., & Monz, Christof (2005). Neuralign: Combining word alignments using neural networks. *Proceedings of HLT-EMNLP* (pp. 65–72). Vancouver, Canada.

Banerjee, Satanjeev, & Lavie, Alon (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43th Annual Meeting of the Association of Computational Linguistics (ACL-2005)*. Ann Arbor, Michigan.

Bannard, Colin, & Callison-Burch, Chris (2005). Paraphrasing with bilingual parallel corpora. *Proceedings of ACL*. Ann Arbor, MI.

Basu, Sugato, Bilenko, Mikhail, & Mooney, Raymond J. (2004). A probabilistic framework for semi-supervised clustering. *KDD '04: ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 59–68). New York, NY: ACM Press.

Berger, Adam L., Della Pietra, Stephen A., & Della Pietra, Vincent J. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics, 22*, 39–72.

Blunsom, Phil, & Cohn, Trevor (2006). Discriminative word alignment with conditional random fields. *Proceedings of COLING-ACL*. Sydney, Australia.

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

Callison-Burch, Chris, Talbot, David, & Osborne, Miles (2004). Statistical machine translation with word- and sentence-aligned parallel corpora. *Proceedings of ACL*. Barcelona, Spain.

Cettolo, Mauro, & Federico, Marcello (2004). Minimum error training of log-linear translation models. *International Workshop on Spoken Language Translation* (pp. 103–106). Kyoto, Japan.

Cherry, Colin, & Lin, Dekang (2003). A probability model to improve word alignment. *Proceedings of ACL* (pp. 88–95). Sapporo, Japan.

Chiang, David (2005). A hierarchical phrase-based model for statistical machine translation. *Proceedings of ACL* (pp. 263–270). Ann Arbor, MI.

Collins, Michael (2000). Discriminative reranking for natural language parsing. *7th International Conf. on Machine Learning*.

Collins, Michael (2002). Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. *Proceedings of EMNLP*. Philadelphia, PA.

Collins, Michael, Koehn, Philipp, & Kucerova, Ivona (2005). Clause restructuring for statistical machine translation. *ACL05* (pp. 531–540). Ann Arbor, MI.

Corston-Oliver, S., & Gamon, M. (2004). Normalizing german and english inflectional morphology to improve statistical word alignment. *In R. E. Frederking and K. B. Taylor (eds.) Machine Translation: From Real Users to Research. Springer Verlag*.

Dagan, Ido, Church, Kenneth W., & Gale, William A. (1993). Robust bilingual word alignment for machine aided translation. *Workshop on Very Large Corpora* (pp. 1–8). Columbus, OH.

Daumé III, Hal, & Marcu, Daniel (2005). Induction of word and phrase alignments for automatic document summarization. *Computational Linguistics*, *31*, 505–530.

Davis, Paul C. (2002). *Stone soup translation: The linked automata model*. Doctoral dissertation, Ohio State University.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B*, *39*, 1–22.

Deng, Yonggang, & Byrne, William (2005). HMM word and phrase alignment for statistical machine translation. *Proceedings of HLT-EMNLP*. Vancouver, Canada.

Drabek, Elliott Franco, & Yarowsky, David (2004). Improving bitext word alignments via syntax-based reordering of english. *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics* (pp. 146–149). Barcelona, Spain.

Eisner, Jason, & Tromble, Roy W. (2006). Local search with very large-scale neighborhoods for optimal permutations in machine translation. *Proceedings of the HLT-NAACL Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*. New York.

Fraser, Alexander, & Marcu, Daniel (2005). ISI's participation in the romanian-english alignment task. *Proceedings of the ACL Workshop on Building and Using Parallel Texts* (pp. 91–94). Ann Arbor, MI.

Fraser, Alexander, & Marcu, Daniel (2006). Semi-supervised training for statistical word alignment. *Proceedings of COLING-ACL* (pp. 769–776). Sydney, Australia.

Fraser, Alexander, & Marcu, Daniel (2007a). Getting the structure right for word alignment: LEAF. *Proceedings of EMNLP-CONLL* (pp. 51–60). Prague, Czech Republic.

Fraser, Alexander, & Marcu, Daniel (2007b). Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, *33*, 293–303.

Fraser, Alexander, Xu, Jinxi, & Weischedel, Ralph (2002). TREC 2002 cross-lingual retrieval at BBN. *E.M. Voorhees and D.K. Harman (Eds) The Eleventh Text Retrieval Conference, TREC 2002. NIST Special Publication 500-251*. (pp. 102–106).

Fung, Pascale, & Cheung, Percy (2004). Mining very-non-parallel corpora: Parallel sentence and lexicon extraction via bootstrapping and em. *Proceedings of EMNLP*. Barcelona, Spain.

Galley, Michel, Graehl, Jonathan, Knight, Kevin, Marcu, Daniel, DeNeefe, Steve, Wang, Wei, & Thayer, Ignacio (2006). Scalable inference and training of context-rich syntactic translation models. *Proceedings of COLING-ACL* (pp. 961–968). Sydney, Australia: Association for Computational Linguistics.

Galley, Michel, Hopkins, Mark, Knight, Kevin, & Marcu, Daniel (2004). What's in a translation rule? *Proceedings of HLT-NAACL* (pp. 273–280).

García-Varea, Ismael, Och, Franz J., Ney, Hermann, & Casacuberta, Francisco (2002). Improving alignment quality in statistical machine translation using context-dependent maximum entropy models. *Proceedings of COLING*. Taipei, Taiwan.

Germann, Ulrich (2003). Greedy decoding for machine translation in almost linear time. *Proceedings of HLT-NAACL*. Edmonton, Canada.

Germann, Ulrich, Jahr, Michael, Knight, Kevin, Marcu, Daniel, & Yamada, Kenji (2004). Fast decoding and optimal decoding for machine translation. *Artificial Intelligence, 154*, 127–143.

Gildea, Daniel (2003). Loosely tree-based alignment for machine translation. *Proceedings of ACL* (pp. 80–87). Sapporo, Japan.

Glover, Fred (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research, 13*, 533–549.

Goutte, Cyril, Yamada, Kenji, & Gaussier, Eric (2004). Aligning words using matrix factorisation. *Proceedings of ACL* (pp. 502–509). Barcelona, Spain.

Hiemstra, Djoerd, & de Jong, Franciska (1999). Disambiguation strategies for cross-language information retrieval. *Lecture Notes in Computer Science, Volume 1696, Jan 1999*.

Hwa, R., Resnik, P., Weinberg, A., & Kolak, O. (2002). Evaluating translational correspondence using annotation projection. *the Proceedings of the 40th Annual Meeting of the ACL*. Philadelphia, PA.

Ittycheriah, Abraham, & Roukos, Salim (2005). A maximum entropy word aligner for Arabic-English machine translation. *Proceedings of HLT-EMNLP* (pp. 89–96). Vancouver, Canada.

Ivanov, Yuri A., Blumberg, Bruce, & Pentland, Alex (2001). Expectation maximization for weakly labeled data. *ICML '01: Eighteenth International Conf. on Machine Learning* (pp. 218–225). Morgan Kaufmann Publishers Inc.

Knight, Kevin (1999). A statistical machine translation tutorial workbook. `http://www.isi.edu/natural-language/mt/wkbk.rtf`.

Koehn, Philipp, Hoang, Hieu, Birch, Alexandra, Callison-Burch, Chris, Federico, Marcello, Bertoldi, Nicola, Cowan, Brooke, Shen, Wade, Moran, Christine, Zens, Richard, Dyer, Chris, Bojar, Ondrej, Constantin, Alexandra, & Herbst, Evan (2007). Moses: Open source toolkit for statistical machine translation. *ACL 2007 Demonstrations*. Prague, Czech Republic.

Koehn, Philipp, & Knight, Kevin (2002). Learning a translation lexicon from monolingual corpora. *ACL02 Workshop on Unsupervised Lexical Acquisition*.

Koehn, Philipp, Och, Franz J., & Marcu, Daniel (2003). Statistical phrase-based translation. *Proceedings of HLT-NAACL* (pp. 127–133). Edmonton, Canada.

Kuhn, Jonas (2004). Experiments in parallel-text based grammar induction. *Proceedings of ACL*. Barcelona, Spain.

Kumar, S., & Byrne, W. (2002). Minimum bayes-risk word alignments of bilingual texts. *Proceedings of the Conf. on Empirical Methods in Natural Language Pr ocessing*. Philadelphia, PA.

Kumar, Shankar, Deng, Yonggang, & Byrne, William (2006). A weighted finite state transducer translation template model for statistical machine translation. *Journal of Natural Language Engineering, 12*.

Lacoste-Julien, Simon, Klein, Dan, Taskar, Ben, & Jordan, Michael (2006). Word alignment via quadratic assignment. *Proceedings of HLT-NAACL* (pp. 112–119). New York, NY.

Lee, Young-Suk (2004). Morphological analysis for statistical machine translation. *Proceedings of HLT-NAACL*. Boston, Massachusetts.

Liang, Percy, Taskar, Ben, & Klein, Dan (2006). Alignment by agreement. *Proceedings of HLT-NAACL*. New York.

Liu, Yang, Liu, Qun, & Lin, Shouxun (2005). Log-linear models for word alignment. *Proceedings of ACL* (pp. 459–466). Ann Arbor, MI.

Lopez, Adam, & Resnik, Philip (2005). Improved HMM alignment models for languages with scarce resources. *Proceedings of the ACL Workshop on Building and Using Parallel Texts* (pp. 83–86). Ann Arbor, MI.

Lopez, Adam, & Resnik, Philip (2006). Word-based alignment, phrase-based translation: what's the link? *Proceedings of AMTA*. Cambridge, MA.

Marcu, Daniel, & Wong, William (2002). A phrase-based, joint probability model for statistical machine translation. *Proceedings of EMNLP* (pp. 133–139). Philadelphia, PA.

Martin, Joel, Mihalcea, Rada, & Pedersen, Ted (2005). Word alignment for languages with scarce resources. *Proceedings of the ACL Workshop on Building and Using Parallel Texts* (pp. 65–74). Ann Arbor, MI.

Matusov, Evgeny, Zens, Richard, & Ney, Hermann (2004). Symmetric word alignments for statistical machine translation. *Proceedings of COLING*. Geneva, Switzerland.

May, Jonathan, & Knight, Kevin (2007). Syntactic re-alignment models for machine translation. *Proceedings of EMNLP-CONLL*. Prague, Czech Republic.

Melamed, I. Dan (1998). *Manual annotation of translational equivalence: the blinker project* (Technical Report 98-07). Institute for Research in Cognitive Science, Philadelpha, PA.

Melamed, I. Dan (2000). Models of translational equivalence among words. *Computational Linguistics, 26*, 221–249.

Melamed, I. Dan (2004). Statistical machine translation by parsing. *Proceedings of ACL*. Barcelona, Spain.

Mihalcea, Rada, & Pederson, Ted (2003). An evaluation exercise for word alignment. *Proceedings of the HLT-NAACL Workshop on Building and Using Parallel Texts* (pp. 1–10). Edmonton, Canada.

Miller, David J., & Browning, John (2003). A mixture model and EM-based algorithm for class discovery, robust classification, and outlier rejection in mixed labeled/unlabeled data sets. *IEEE Trans. Pattern Anal. Mach. Intell., 25*, 1468–1483.

Miller, David J., & Uyar, Hasan S. (1997). A mixture of experts classifier with learning based on both labelled and unlabelled data. *Advances in Neural Information Processing Systems, M.C. Mozer, M.I. Jordan, T. Petsche, eds. Cambridge, MA: MIT Press, vol. 9* (pp. 571–577).

Moore, Robert C. (2002). Fast and accurate sentence alignment of bilingual corpora. *Proceedings of AMTA*.

Moore, Robert C. (2004). Improving IBM word-alignment model 1. *Proceedings of ACL*. Barcelona, Spain.

Moore, Robert C. (2005). A discriminative framework for bilingual word alignment. *Proceedings of HLT-EMNLP*. Vancouver, Canada.

Moore, Robert C., Yih, Wen-Tau, & Bode, Andreas (2006). Improved discriminative bilingual word alignment. *Proceedings of COLING-ACL* (pp. 513–520). Sydney, Australia.

Munteanu, Dragos Stefan, Fraser, Alexander, & Marcu, Daniel (2004). Improved machine translation performance via parallel sentence extraction from comparable corpora. *Proceedings of HLT-NAACL* (pp. 265–272). Boston, Massachusetts.

Neal, Radford M., & Hinton, Geoffrey E. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. *Learning in Graphical Models*. Kluwer.

Niessen, Sonja, & Ney, Hermann (2004). Statistical machine translation with scarce resources using morpho-syntactic information. *Computational Linguistics, 30*, 181–204.

Nigam, Kamal, McCallum, Andrew K., Thrun, Sebastian, & Mitchell, Tom M. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning, 39*, 103–134.

Och, Franz J. (1999). An efficient method for determining bilingual word classes. *Processings of EACL* (pp. 71–76). Bergen, Norway.

Och, Franz J. (2003). Minimum error rate training in statistical machine translation. *Proceedings of ACL* (pp. 160–167). Sapporo, Japan.

Och, Franz J., Gildea, Daniel, Khudanpur, Sanjeev, Sarkar, Anoop, Yamada, Kenji, Fraser, Alex, Kumar, Shankar, Shen, Libin, Smith, David, Eng, Katherine, Jain, Viren, Jin, Zhen, & Radev, Dragomir (2003). *Syntax for statistical machine translation, final report* (Technical Report). JHU Workshop, Baltimore, MD.

Och, Franz J., Gildea, Daniel, Khudanpur, Sanjeev, Sarkar, Anoop, Yamada, Kenji, Fraser, Alex, Kumar, Shankar, Shen, Libin, Smith, David, Eng, Katherine, Jain, Viren, Jin, Zhen, & Radev, Dragomir (2004). A smorgasbord of features for statistical machine translation. *2004 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL)* (pp. 161–168). Boston.

Och, Franz J., & Ney, Hermann (2002). Discriminative training and maximum entropy models for statistical machine translation. *Proceedings of ACL* (pp. 295–302). Philadelphia, PA.

Och, Franz J., & Ney, Hermann (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, *29*, 19–51.

Och, Franz J., & Ney, Hermann (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, *30*, 417–449.

Pang, Bo, Knight, Kevin, & Marcu, Daniel (2003). Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. *Proceedings of the HLT-NAACL Workshop on Building and Using Parallel Texts*.

Papineni, Kishore A., Roukos, Salim, Ward, Todd, & Zhu, Wei-Jing (2001). *BLEU: a method for automatic evaluation of machine translation* (Technical Report RC22176 (W0109-022)). IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY.

Press, William H., Teukolsky, Saul A., Vetterling, William T., & Flannery, Brian P. (2002). *Numerical recipes in C++*. Cambridge, UK: Cambridge University Press.

Quirk, Chris, Brockett, Chris, & Dolan, William (2004). Monolingual machine translation for paraphrase generation. *Proceedings of EMNLP*.

Quirk, Chris, Menezes, Arul, & Cherry, Colin (2005). Dependency treelet translation: Syntactically informed phrasal SMT. *Proceedings of ACL* (pp. 271–279). Ann Arbor, MI.

Resnik, Philip, & Smith, Noah A. (2003). The web as a parallel corpus. *Computational Linguistics*, *29*, 349–380.

Riezler, Stefan, Vasserman, Alexander, Tsochantaridis, Ioannis, Mittaland, Vibhu, & Liu, Yi (2007). Statistical machine translation for query expansion in answer retrieval. *Proceedings of ACL*. Prague, Czech Republic.

Rijsbergen, Keith van (1979). *Information retrieval*. Newton, MA: Butterworth-Heinemann.

Seeger, Matthias (2000). Learning with labeled and unlabeled data. *Technical report, 2000. Available at http://www.dai.ed.ac.uk/ seeger/papers.html*.

Shen, Libin, & Joshi, Aravind K. (2005). Ranking and reranking with perceptron. *Machine Learning*, *60*, 73–96.

Taskar, Ben, Lacoste-Julien, Simon, & Klein, Dan (2005). A discriminative matching approach to word alignment. *Proceedings of HLT-EMNLP*. Vancouver, Canada.

Tiedemann, Joerg (2003). Combining clues for word alignment. *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Budapest, Hungary.

Toutanova, Kristina, Ilhan, H. Tolga, & Manning, Christopher D. (2002). Extensions to HMM-based statistical word alignment models. *Proceedings of EMNLP*. Philadelphia, PA.

Udupa, Raghavendra, & Maji, Hemanta Kumar (2005). Theory of alignment generators and applications to statistical machine translation. *Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*. Edinburgh.

Udupa, Raghavendra, & Maji, Hemanta Kumar (2006). Computational complexity of statistical machine translation. *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Venugopal, Ashish, Vogel, Stephan, & Waibel, Alex (2003). Effective phrase translation extraction from alignment models. *Proceedings of ACL* (pp. 319–326). Sapporo, Japan.

Vogel, Stephan, Ney, Hermann, & Tillmann, Christoph (1996). HMM-based word alignment in statistical translation. *Proceedings of COLING* (pp. 836–841). Copenhagen, Denmark.

Wang, Ye-Yi, & Waibel, Alex (1998). Modeling with structures in statistical machine translation. *Proceedings of COLING-ACL* (pp. 1357–1363). Montreal, Canada.

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

Wu, Dekai, & Xia, Xuanyin (1995). Large-scale automatic extraction of an english-chinese lexicon. *Machine Translation*, *9*, 285–313.

Xu, Jinxi, Fraser, Alexander, & Weischedel, Ralph (2002). Empirical studies in strategies for arabic retrieval. *Proceedings of the 25th Annual International Conference on Research and Development in Information Retrieval (SIGIR)* (pp. 269–274). Tampere, Finland.

Xu, Jinxi, Weischedel, Ralph M., & Nguyen, Chanh (2001). Evaluating a probabilistic model for cross-lingual information retrieval. *Proceedings of the 24th Annual International Conference on Research and Development in Information Retrieval (SIGIR)*.

Yamada, Kenji, & Knight, Kevin (2001). A syntax-based statistical translation model. *Proceedings of ACL* (pp. 523–530). Toulouse, France.

Yarowsky, David, Ngai, Grace, & Wicentowski, Richard (2001). Inducing multilingual text analysis tools via robust projection across aligned corpora. *Human Language Technology Conference* (pp. 109–116). San Diego, CA.

Zens, Richard, Matusov, Evgeny, & Ney, Hermann (2004). Improved word alignment using a symmetric lexicon model. *Proceedings of COLING*. Geneva, Switzerland.

Zens, Richard, & Ney, Hermann (2004). Improvements in phrase-based statistical machine translation. *Proceedings of HLT-NAACL* (pp. 257–264). Boston, MA.

Zollmann, Andreas, & Venugopal, Ashish (2006). Syntax augmented machine translation via chart parsing. *NAACL Workshop on Statistical Machine Translation*.

# Appendix A

# IBM Model 4

## A.1 Introduction

The definitive work on early generative models of word alignment for machine translation is by Brown et al. (1993), which describes a group of models called the IBM Models. We focus on IBM Model 4 in particular. An overview of other generative models for word alignment is given in Section 3.6.

## A.2 The IBM Models

Brown et al. (1993) developed five statistical models of translation (IBM Models 1 through 5) and parameter estimation techniques for them. The models were designed to be used in a pipeline, where each model is bootstrapped from the previous model.

For ease of exposition, the source language for the translation task is referred to as "French", and the target language is referred to as "English", although these can be any language pairs in practice. The translation problem is defined as given a French string $f$, find the English string $\hat{e}$ according to Equation A.1.

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, Pr(e|f) = \underset{e}{\operatorname{argmax}}\, Pr(e) * Pr(f|e) \tag{A.1}$$

where $e$ represents any potential English string made up of English words. $Pr(e)$ represents the true distribution over English strings. $Pr(f|e)$ represents the true distribution over French strings generated from English strings.

Consider $P_\theta(f|e)$ to be a model of $Pr(f|e)$. If we introduce a hidden variable $a$ representing word alignments, we can sum over these variables, see Equation A.2.

$$P_\theta(f|e) = \sum_a P_\theta(f, a|e) \tag{A.2}$$

For our task, which is word alignment annotation, we have fixed strings $f$ and $e$, and we wish to select the best alignment according to the model, $\hat{a}$, which we do in Equation A.3.

$$\hat{a} = \underset{a}{\operatorname{argmax}}\, P_\theta(a|e, f) = \underset{a}{\operatorname{argmax}}\, P_\theta(f, a|e) \tag{A.3}$$

The only alignments in the IBM models which can have non-zero probability involve links from one English word to zero or more French words. We call alignments which can have non-zero probability within a model "feasible" in that model. Not all French words must be aligned with an English word which appears in the sentence; those that aren't are considered to be spontaneously generated. For reasons of notational convenience we consider them to be aligned to a so-called NULL word which we will denote $e_0$.

### A.2.1 Introduction to Model 4

We concentrate on Model 4, presenting the generative story, the mathematical formulation, and the unsupervised training algorithm for the model using a variant of the Expectation Maximization (EM) algorithm. We also outline how Model 4 is used in practice, including the heuristic steps applied to the alignments predicted by the model in order to produce a final word alignment.

Brown et al. (1993) defined a model of $Pr(f|e)$ called Model 4. IBM Model 4 is a generative model, which is a model of how a French string $f$ is generated given an English string $e$. The steps followed determine a unique alignment $a$.

To generate $f$ from $e$ (using steps which determine $a$), the following generative story is used. We first pick for each English word a fertility value, which is the number of French words which will be generated from it. Then we choose a fertility value for

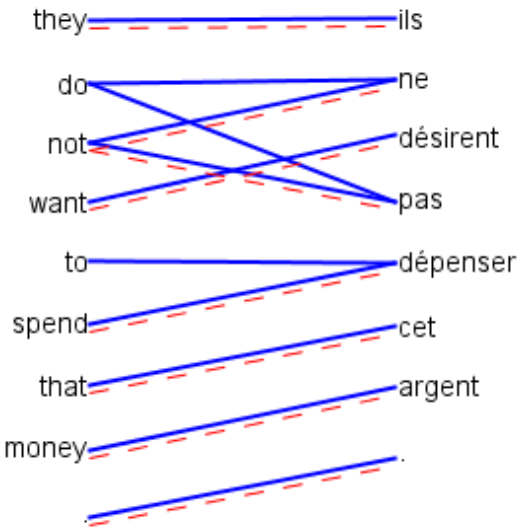


Figure A.1: French/English example, gold standard (solid lines) and best possible Model 4 decisions (dashed lines)

the NULL English word conditioned on the total number of French words generated from the non-NULL English words. For each English word including the NULL word we pick the identity of the French words that are generated from it. Finally, we choose the position of each French word in the French sentence.

### A.2.2 Example of Model 4 Generative Story

We start with an English sentence. We will use a shorter sentence similar to our example from the introduction which is shown in Figure A.1. The gold standard decisions are the solid lines, while the best alignment which is feasible in Model 4 is indicated with

dashed lines. Here are the Fertility and Translation decisions which we would like the model to make for our example:

- “They” is Fertility 1. It generates French “ils”.
- “do” is Fertility 0.
- “not” is Fertility 2. It generates French “ne” and French “pas”.
- “want” is Fertility 1. It generates French “désirent”.
- “to” is Fertility 0.
- “spend” is Fertility 1. It generates French “dépenser”.
- “that” is Fertility 1. It generates French “cet”.
- “money” is Fertility 1. It generates French “argent”.
- The English period is Fertility 1. It generates the French period.
- The English NULL word does not generate any spurious French words.

Because of the 1 to many assumption, we can not draw links from both English “do” and “not” to French: “ne” and “pas”. We also can not draw links from both “to” and “spend” to “dépenser”. This is a serious problem. We present a new model called LEAF in Chapter 3 which overcomes the 1-to-many assumption.

### A.2.3 Model 4 Generative Story

We present the full Model 4 generative story, following the exposition of Brown et al. (1993) very closely. We do make one assumption differently from Brown et al. (1993), which is that the placement position is only dependent on the previous placement position (in IBM Model 4 there is an additional conditioning on automatically derived word classes, but we omit this to simplify the presentation). Note that there is a non-zero probability of "failure", i.e. there is a non-zero probability that the generative story fails to generate anything. This means the model is deficient, wasting some probability mass.

The variable $l$ refers to the length of the English sentence $e$, and $m$ refers to the length of the generated French sentence $f$. $\phi_i$ is the number of French words generated by the English word at position i. The identity of these words is $\tau_{ik}$ ($k$ ranges from 1 to $\phi_i$), and their French position is $\pi_{ik}$. The term $\rho_i$ refers to the previous English word to the English word at position $i$ which has fertility greater than zero. $c_{\rho_i}$ is the "center" of the words placed by the previous English word of non-zero fertility to the English word at position i. The calculation of $c_z$ for a non-zero-fertility English word at position z is described in equation A.4, below.

The Model 4 generative story:

1. For each $i = 1, 2, ..., l$ choose a fertility value $\phi_i$ according to the distribution $n(\phi_i|e_i)$.

2. Choose a fertility value $\phi_0$ according to the distribution $n_0(\phi_0|\sum_{i=1}^{l}\phi_i)$.

3. Let m = $\phi_0 + \sum_{i=1}^{l}\phi_i$

4. For each $i = 0, 1, ..., l$ and each $k = 1, 2, ..., \phi_i$, choose a French word $\tau_{ik}$ according to the distribution $t(\tau_{ik}|e_i)$.

5. For each $i = 1, ..., l$ and each $k = 1, 2, ..., \phi_i$, choose a position $\pi_{ik}$ as follows:

   - If $k = 1$, choose $\pi_{i1}$ according to the distribution $d_1(\pi_{i1} - c_{\rho i})$
   - If $k > 1$, choose $\pi_{ik}$ according to the distribution $d_{>1}(\pi_{ik} - \pi_{ik-1})$ subject to the constraint that $\pi_{ik-1} < \pi_{ik}$

6. If any position has been chosen more than once, return "failure"

7. For each $k = 1, 2, ..., \phi_0$ choose a position $\pi_{0k}$ from $\phi_0 - k + 1$ remaining vacant positions in $1, 2, ..., m$ according to the uniform distribution.

8. Let $f$ be the string with $f\pi_{ik} = \tau_{ik}$

The calculation of the "center" of the French words generated from a non-zero fertility English word at position $i$ in the English sentence is shown in Equation A.4.

$$c_i = \text{ceiling}(\sum_{k=1}^{\phi_i}\pi_{ik}/\phi_i) \tag{A.4}$$

We call French words generated from English words (not including the special English NULL word) "non-spurious", as their generation is explained by the English words we observe. The number of non-spurious words is $m'$, which is the sum of the fertilities of the non-null English words, as shown in Equation A.5.

$$m' = \sum_{i=1}^{l} \phi_i \tag{A.5}$$

For notational reasons we annotate unexplained French words as being generated from the English NULL word, but this does not directly reflect the generative process. These French words are called "spurious", as they aren't being generated by the English words we observe. In the generative story, these words are generated as a part of the process of generating non-spurious French words. The parameter $p_1$ represents the probability that as we generate a non-spurious French word we also generate a single spurious French word, while $p_0$ is the probability that as we generate a non-spurious French word we don't generate any spurious French word ($p_0 + p_1 = 1$). The number of spurious words generated is modeled using a binomial distribution where the number of trials is $m'$ and the chance of trial success (generating a spurious word) is $p_1$ (the chance of trial failure is $1 - p_1 = p_0$). The equation is given in Equation A.6.

$$n_0(\phi_0|m') = \binom{m'}{\phi_0} p_0^{m'-\phi_0} p_1^{\phi_0} \tag{A.6}$$

The decisions made in a particular generative story can be mapped to a unique alignment $a$. When working with 1 to many alignments, a compact representation of an alignment which is sometimes used is a vector of length $m$ (the length of the French sentence), which indicates for each French word $f_j$ the position of the English word which generated it (i.e., the values in the vector range from $0, .., l$). The reader can verify that given the particular generative story outlined for our example (with the addition of distortion operations to specify the placement of the words) we generate the unique French string and unique alignment shown in A.1. Under the Model 4 generative story, given a starting English string $e$ and the decisions made (which did not result in "failure"), we generate a unique French string $f$ and a unique alignment $a$, and this is always the case[1].

### A.2.4 Model 4 Mathematical Formulation

Given an English string $e$, a French string $f$ and a candidate alignment $a$, we would like to look up $p(f, a|e)$. The formula for Model 4 is in Equation A.7. See Equation A.8 for the expansion of the simplified distortion calculation which we abbreviate $D_{ik}(j)$.

$$\left[\prod_{i=1}^{l} n(\phi_i|e_i)\right] n_0(\phi_0|\sum_{i=1}^{l}\phi_i)\prod_{i=0}^{l}\prod_{k=1}^{\phi_i} t(\tau_{ik}|e_i)\prod_{i=1}^{l}\prod_{k=1}^{\phi_i} D_{ik(\pi_{ik})} \tag{A.7}$$

[1]The inverse is not generally true; given an English string $e$, a French string $f$, and an alignment $a$, there is not only one particular generative story that would have generated $f$ and $a$ from $e$ unless $\phi_0 = 0$ (such as in our example).

$$D_{ik}(j) = \begin{cases} d_1(j - c_{\rho_i}) & \text{if } k = 1 \\ d_{>1}(j - \pi_{ik-1}) & \text{if } k > 1 \end{cases} \tag{A.8}$$

### A.2.5 Training Model 4 Using Expectation-Maximization

#### A.2.5.1 Introduction

In this section we present the training of Model 4 using the Expectation-Maximization (EM) algorithm. EM is an algorithm for finding parameter settings of a model which maximize the expected likelihood of the observed and the unobserved data (this is called the complete data likelihood; the incomplete data likelihood is the likelihood of only the observed data). Intuitively, in statistical word alignment, the E-step corresponds to calculating the probability of all alignments according to the current model estimate, and the M-step is the creation of a new model estimate given a probability distribution over alignments (which was calculated in the E-step).

Model 4 is a generative model with carefully controlled complexity. In Model 4, given strings $e$ and $f$, every particular generative story which explains how $f$ was generated from $e$ represents $l + 2m$ decisions. There are $l$ fertility decisions over the English string and there is a generation decision and a placement decision for each of the $m$ French words. It is important in EM to control complexity. If complexity is not carefully controlled, there can be a bias towards simpler structure, by which we mean solutions where less decisions are made. If this is the case then heuristics must be used

to compensate. It is difficult to craft an effective generative model of word alignment which has a constant number of decisions for use with EM.

#### A.2.5.2 E-step

In the E-step we would ideally like to enumerate all possible alignments and label them with $p(f, a|e)$. However, this is not possible when using an alignment model as complex as Model 4. As we will see below in the discussion of the M-step, we would at least like to find the most likely alignment given the model. This is referred to as the Viterbi alignment, $\hat{a}$ in this formula:

$$\hat{a} = \underset{a}{\operatorname{argmax}}\, P_\theta(a|e, f) = \underset{a}{\operatorname{argmax}}\, P_\theta(f, a|e) \tag{A.9}$$

This is a repeat of equation A.3 which represents the task of finding an approximate Viterbi alignment to output as the final alignment output from the alignment process. Here, in Equation A.9 we are referring to the search for an alignment during training. We can vary this to be, for instance, the search for the 10 most probable alignments (where a probability distribution over the 10 alignments would be used for the M-step).

The calculation of the Viterbi alignment for IBM Model 4 was proven to be NP-hard by Udupa and Maji (2006). So we take the most probable alignment we can find, and assume it is the Viterbi alignment.

A local hillclimbing search algorithm is used (Brown et al., 1993). The search starts from the presumed Viterbi alignment found during the previous iteration of training. Brown et al. (1993) recommends instead starting the search from the Viterbi alignment of IBM Model 2, but we do not believe this is more effective. All alignments which are reachable by two search operations from the current best alignment are considered. One search operation is to change the generation decision for a French word to a different English word, and the other search operation is to swap the generation decision for two French words. The two search operations are applied exhaustively, and the best resulting alignment is chosen; this is iterated. The search is terminated when no improved alignment can be found.

#### A.2.5.3 M-step

For the M-step, we would like to take a sum over all possible alignments for each sentence pair, weighted by $p(a|e, f)$ which we calculated in the E-step (note that the labels labeled in the E-step must be renormalized to sum to 1 for each $e, f$ pair, as they are estimates of $p(f, a|e)$, and we would like estimates of $p(a|e, f)$). As we mentioned, this is not tractable.

We make the assumption that the single assumed Viterbi can be used to update our estimate in the M-step (which we call $p_M(a|e, f)$, the probability of the alignment given the sentence $e$ and the sentence $f$):

$$p_M(a|e,f) = \begin{cases} 1 & \text{if } a = \hat{a} \\ 0 & \text{if } a \neq \hat{a} \end{cases} \tag{A.10}$$

Note that when discussing "Viterbi training", we are abusing the term "Viterbi alignment" to mean the best alignment according to the model that we can find, not the best alignment according to the model that exists.

We estimate new parameters from the assumed Viterbi alignments found during the E-step by simply counting events in the assumed Viterbi alignments, since they are assumed in equation A.10 to be the only alignments of non-zero probability. We collect the counts listed in Figure A.2. After collecting the counts, for each condition, we normalize these counts so that they sum to one, which provides us with the model estimate for the next E-step, listed in the following equations:

$$t(\bar{f}|\bar{e}) = c_t(\bar{f}|\bar{e}) / \sum_{\bar{f}'} c_t(\bar{f}'|\bar{e}) \tag{A.11}$$

$$n(\bar{\phi}|\bar{e}) = c_n(\bar{\phi}|\bar{e}) / \sum_{\bar{\phi}'} c_n(\bar{\phi}'|\bar{e}) \tag{A.12}$$

$$d_1(\triangle j) = c_{d1}(\triangle j) / \sum_{\triangle j'} c_{d1}(\triangle j') \tag{A.13}$$

| | |
|---|---|
| $c_t(\bar{f}\|\bar{e})$ | translation counts, $\bar{f}$ is a French word and $\bar{e}$ is an English word |
| $c_n(\bar{\phi}\|\bar{e})$ | fertility counts, $\bar{\phi}$ is the number of words generated by the English word $\bar{e}$ |
| $c_{d1}(\triangle j)$ | distortion (movement) counts of the first French word translated from a single English word (looking from left to right in French sentence) |
| $c_{d>1}(\triangle j)$ | distortion (movement) counts of other French words translated from a single English word |

Figure A.2: Counts collected in unsupervised Model 4 training

$$d_{>1}(\triangle j) = c_{d>1}(\triangle j) / \sum_{\triangle j'} c_{d>1}(\triangle j') \tag{A.14}$$

Clearly the Viterbi training approximation is related to EM training, which tries to maximize the complete data log likelihood. Neal and Hinton (1998) analyzed approximate EM training and motivates this general variant. We would like to eventually try using a probability estimate over a larger set of hypothesized alignments to reestimate the model, but finding a set to use which will help the performance of the estimated models is an open research problem.

### A.2.6 How Model 4 is Used in Practice

#### A.2.6.1 Open Parameters Used with Model 4

In practice, $p_0$ is not usually trained using likelihood (see (Brown et al., 1993) for details of count collection). Instead $p_0$ is set to a fixed value which produces good quality alignments.

The GIZA++ Model 4 implementation used in our experiments has two smoothing parameters to smooth the fertility distribution which are not part of the original Model 4 formulation.

We set these three open parameters based on final translation quality, in an expensive grid-search process which involves building a full SMT system for each parameter setting we would like to try. In our work on semi-supervised training presented in Chapter 4 we overcome this difficulty and show how to efficiently train such parameters using a small amount of hand annotated word alignment data.

#### A.2.6.2 Heuristic Symmetrization for the IBM Models

All of the IBM Models assign zero probability to alignments in which more than one English word is aligned to a single French word. This is a poor assumption. Ideally, we would like a model to be able to assign non-zero probabilities to all of the possible alignments, which includes alignments that violate the one to many assumption.

In practice, in current state of the art machine translation systems, heuristic techniques are used to obtain M-to-N discontinuous alignments. For 1-to-N models like the IBM Models, the following approach is used:

- We are supplied with a bitext to be aligned, a 1-to-N alignment system, and a symmetrization heuristic.
- Generate the predicted 1 to many alignment in the direction English to French. In this alignment one English word aligns to zero or more French words. Call the resulting alignment A1.
- Generate the predicted 1 to many alignment in the direction French to English. In this alignment one French word aligns to zero or more English words. Call the resulting alignment A2.
- Combine A1 and A2 into a many to many alignment using a symmetrization heuristic. Call this many to many discontinuous alignment A3
- Return A3

We briefly discuss the three symmetrization heuristics defined by Och and Ney (2003). For discussion of other heuristics the reader is referred to Koehn et al. (2003).

- The “Union” symmetrization heuristic involves taking the union of the links in the A1 and A2 alignments. This results in an alignment having M-to-N discontinuous structure.

- The “Intersection” symmetrization heuristic involves taking the intersection of the links in the A1 and A2 alignments. This will result in a 1-to-1 alignment structure.

- The “Refined” symmetrization heuristic starts from the “Intersection” 1-to-1 alignment, and adds some of the links present in the “Union” M-to-N discontinuous alignment following the algorithm defined by Och and Ney (2003). This results in an alignment containing 1-to-N and M-to-1 correspondences, but importantly the words in the minimal translational correspondences must be consecutive, so this is not as general as the “Union” heuristic. This heuristic is described in further detail in Section 2.2.3.

Use of these heuristics is undesirable. We would ideally use a model which is able to assign non-zero probability to many to many discontinuous alignments directly, without requiring the use of heuristics. We present the LEAF model in Chapter 3 which is able to do this.

### A.2.7 Discussion

We have presented the important issues behind the work of Brown et al. (1993). We have shown how Model 4 works in detail, and have discussed the structural assumptions that were used in all of the IBM models. In addition, we have discussed how Model 4 is used in practice. We hope that the reader now has an understanding of the previous state of the art unsupervised solution for word alignment and some idea of its strengths and weaknesses.

For the baselines in this thesis, we directly compare results with the freely available GIZA++ software package, which is used to generate the alignments for many MT systems.

However, we have also reimplemented the Model 4 alignment model. We have implemented our code so that we can calculate presumed Viterbi alignments for Model 4 on many servers using a small memory footprint, which is a large advantage over GIZA++ which has a large memory footprint and can only use one server.

# Appendix B

# Details of Introductory Experiments

## B.1 Building Translation Systems with Word Alignments

SMT systems are usually broken down into two types of model, the translation model, which is a model of translational correspondence between the source and target languages, and the language model, which is a model of well-formed sentences in the target language.

The translation model is estimated using a bitext of parallel source language sentences and target language sentences and an alignment of that bitext. The model estimated from the bitext is called the translation model because it models the mapping of a source phrase to a target phrase. The language model is estimated only from the target language text, this is a model of well-formed target language sentences. We can

use additional target language text which is not from the bitext to help us build a better language model.

In the experiments presented in this section, we use the ISI implementation of the alignment templates system (Och & Ney, 2004), which is a phrase-based SMT translation system (Koehn et al., 2003). This is a log-linear translation model (Och & Ney, 2002). The log-linear model is trained to maximize an automatic translation quality metric called BLEU (Papineni et al., 2001). BLEU is an automatic evaluation metric which measures translation quality. BLEU has been shown to correlate well with human judgments of quality. To maximize BLEU we use the Maximum BLEU training algorithm (Och, 2003). This algorithm uses the translation "dev" set as training data to train the weights of the log linear model so as to maximize BLEU.

In phrase-based SMT, we estimate the phrase lexicon (the most important part of the translation model) using a word alignment of the training bitext. We will vary how we construct this word alignment. This is the only factor varied in all experiments in this thesis[1]. We will always compare two or more systems using the same language models and the same bitext, but the two alignments of the bitext will be different.

For all of our experiments, we use a language model built on the target language training data and a large language model built on news data.

[1]Note that because we only allow the final alignment to vary, features based on IBM Model 1 (a lower order alignment model) are also held constant.

We evaluate an alignment by building a machine translation system, translating a machine translation test set and evaluating it using BLEU. For ease of reading we multiply the BLEU score by 100, and for this reason we report "BLEU %" in our results.

We present our own word alignment systems in Chapters 3 and 4. In this section we present results based on our baseline, a widely used unsupervised alignment procedure, which is used as the baseline in most papers on word alignment. This approach uses a freely available software package called GIZA++ (Och & Ney, 2003), which implements several alignment models. GIZA++ implements both the IBM Models (Brown et al., 1993) and the HMM word alignment model (Vogel et al., 1996). In our baseline, we use heuristic post-processing of the output of GIZA++, as is standardly done.

GIZA++ implements both Model 4 and the HMM using a few extensions which were not in the original formulations. We use IBM Model 1, the Aachen HMM, and IBM Model 4 in that order (these models "bootstrap" from one another, see Appendix A for more details). The output of these models is an alignment of the bitext which projects one language to another. GIZA++ is run end-to-end twice. In one case we project the source language to the target language, and in the other we project the target language to the source language. The output of GIZA++ is then post-processed using so-called "symmetrization heuristics" to produce a single alignment by combining the source to target alignment and the target to source word alignment output by the models.

We describe Model 4 and the heuristic symmetrization algorithms in more detail in Appendix A.

## B.2 Experimental Details for the Romanian/English Weak Oracle Experiment

We would like to substantiate the claim that improved alignments will lead to improved MT systems. We show that there exist alignments of a fixed bitext which are significantly better for translation than the alignments generated by our baseline system. We generate the improved alignments by using an "oracle", a system which (in an unfair fashion) tells our alignment system how to improve the alignments. We measure phrase-based statistical machine translation performance both when using our baseline alignment system, and using the "oracle". We show that alignments can be improved by showing that the "oracle" alignments lead to higher performance than the baseline.

**Experiment overview:** We report on a "weak oracle" experiment. We select a training bitext (parallel sentences in Romanian and English) to be aligned under three different experimental conditions. For the baseline, we use the current state of the art alignment system to align the training bitext and then build a machine translation system and translate a held out test set. The second experimental condition is to show that our reimplementation of the baseline has identical performance (this is only necessary

because we need to use our reimplementation for the weak oracle). For the "weak oracle", we allow the word alignment system access to gold standard alignments of the test data to force it to make better alignment decisions on the training bitext. The difference with the baseline is that a "weak oracle" told the alignment system how to align the training bitext well (for this test set). We show that the translations of the test data generated by an MT system using this alignment is of higher quality than the translation which was generated by the baseline system. This shows the existence of better alignments than those generated by our baseline system.

**Experiment details:** We build SMT systems for three distinct experimental conditions which we list below. See Table B.1 for statistics of the data.

We use the training data originally supplied for the WPT05 shared task (Martin et al., 2005) on word alignment. For the machine translation "dev" set, which is used for Maximum BLEU training, we use the WPT05 alignment test set, and for the machine translation "test" set, we use the WPT03 alignment test set.

The first system, "Symmetrized GIZA++", is the result of running 5 iterations of running GIZA++ IBM Model 1, 5 iterations of GIZA++ HMM Model, and 4 iterations of GIZA++ Model 4 where one alignment was generated in the Romanian to English direction and one alignment was generated in the English to Romanian direction. The second system, "Symmetrized Model 4", is the result of bootstrapping our

implementation of Model 4 using the GIZA++ HMM Model outputs, running 4 iterations of Model 4 in both directions using our implementation and is otherwise identical to “Symmetrized GIZA++”.

The first system, “Symmetrized GIZA++”, is the result of running GIZA++ Model 4 and post-processing the output with heuristics. “Symmetrized Model 4” is our implementation of Model 4, also post-processed with the same heuristics. The third system, “Weak Oracle” is generated by concatenating the training data together with 1000 copies of the manually annotated gold standard word alignments for both the machine translation “dev” set and the machine translation “test” set each time parameters are estimated for use in our implementation of Model 4. These gold standard alignments are removed before the alignments are used to build the machine translation system. The effect these gold standard alignments have on the machine translation system is indirect; they force the decisions made in the alignment of the training data to be good decisions for the translation of the development and test sets (which is why this is an oracle experiment).

Using gold standard word alignments for a fraction of the parallel sentences in our augmented training+dev+test corpus is easy to do in our reimplementation of Model 4 but not implemented in GIZA++, which is why we use our implementation to implement the “Weak Oracle”. A preliminary comparison is necessary to show that our alignment package is equivalent in performance to GIZA++. The BLEU scores in line 1

(GIZA++) and line 2 (our implementation) of Table B.2 show that our implementation has equivalent performance.

The main comparison directly addresses the existence of better alignments. We compare "Symmetrized Model 4" (line 2 of Table B.2) with "Weak Oracle" (line 3 of Table B.2). The "Weak Oracle" is 3.30 BLEU points better than "Symmetrized Model 4". This shows the existence of alignments which give us better translation performance than the best we can obtain with our baseline.

Note that this is only a weak oracle experiment because it is possible to find even better alignments. For instance, if a word is translated as two words in the gold standard in one context, it will translate as two words in every context. This will damage the quality of other alignments of that word in other contexts which could affect translation decisions and adversely affect translation quality. In addition, the oracle is weak because it is constrained to the alignment structure which is modeled by the IBM Models which is not the correct alignment structure (see Section 1.2.4). If we were given infinite resources to search all alignments exhaustively by evaluating them in a translation system directly, it would be possible to find better alignments with even larger BLEU improvements (which would be a strong oracle).

**Experiment Results Summary:** Table B.2 shows that the current state of the art (line 1) and our reimplementation (line 2) have the same performance. Line 2 is the baseline for the main experiment, the BLEU score is 23.06. Line 3 shows the existence

Table B.1: Romanian/English Weak Oracle Data

| | | ROMANIAN | ENGLISH |
|---|---|---|---|
| TRAINING | SENTENCES | 48222 | |
| | WORDS | 971525 | 1024321 |
| | VOCABULARY | 45782 | 25507 |
| | SINGLETONS | 19328 | 8567 |
| TRANSLATION DEV | SENTENCES | 200 | |
| | WORDS | 4365 | 4562 |
| TRANSLATION TEST | SENTENCES | 248 | |
| | WORDS | 5495 | 5639 |

Table B.2: Romanian/English Weak Oracle Results

| SYSTEM | BLEU % |
|---|---|
| SYMMETRIZED GIZA++ | 22.85 |
| SYMMETRIZED MODEL 4 | 23.06 |
| WEAK ORACLE | 26.36 |

of alignments which give us better translation performance than the best we can currently obtain with our baseline. These improved alignments result in a BLEU score of 26.36; this is 3.30 points better than the baseline which is a large improvement. This experiment is evidence that MT quality can be improved by producing improved word alignments. We will show how to obtain such improved word alignments (without using an oracle) in the main part of the dissertation.

# Appendix C

# Search Implementation Details

## C.1 Comparing the Current LEAF Search Implementation with Model 4

Our current implementation of the LEAF search (used in both the D-step and the E-step) is unoptimized. We compare it with an unoptimized version of Model 4 (our implementation) and a highly optimized implementation of Model 4 (GIZA++, Och and Ney (2003)). We will discuss how the search for the LEAF Viterbi alignment can be improved (using the same techniques implemented in GIZA++) to be about 12 times slower than the time required by the GIZA++ Model 4 implementation. GIZA++ is implemented such that only a single processor can be used. Both of our current LEAF and Model 4 search implementations are fully parallelized and can be run on

Table C.1: Average milliseconds per sentence pair in E-Step

| System | E to F | F to E | Final M-to-N |
|---|---|---|---|
| Unoptimized Model 4 (unsupervised) | 336 | 493 | 829 |
| GIZA++ Model 4 (unsupervised) | 8 | 10 | 18 |
| Unoptimized LEAF (unsupervised) | NA | NA | 10151 |
| Unoptimized LEAF (semi-supervised) | NA | NA | 11810 |

any number of processors; this is what has enabled us to carry out experimentation without implementing the optimizations.

The number of milliseconds used per sentence pair in the E-Step is presented in Table C.1. We calculated this on the French data set which is 2,842,184 sentences, 67,366,819 English words (see Table 3.3 on Page 74 for the full statistics). This data set contains sentences of length up to 254 words, which increases the average search time required, versus other data sets where the sentence length cut-off is significantly shorter.

We have already shown that our implementation of Model 4 and GIZA++ have the same performance (as measured by BLEU) in Appendix B.2. In our discussion of Model 4 alignment search implementations we restrict ourselves to the "baseline" search algorithm, as described by Brown et al. (1993), which uses a hillclimbing search from only one starting point to converge to a local probability maxima; no restarts are used, see Section 3.4.2.1.

The first line of Table C.1 shows that we spend an average of 829 milliseconds per sentence pair (column 3) for our unoptimized Model 4 implementation (we sum the

two directions in columns 1 and 2 to determine this number, and assume that applying the "Union" or "Refined" symmetrization heuristic to these two alignments to obtain the final alignment takes a negligible amount of time).

We consider Model 4 in the English to French alignment direction. Our unoptimized implementation of Model 4 uses a representation of the alignment as a vector $v$ of length $m$ (the number of French words) where $v_j$ is the position of the English word which generated the French word at position $j$. The two search operations, "move" and "swap" (described in Appendix A.2.5.2), copy this alignment vector, and change one position (for "move"; two positions are changed for "swap"), and then score the new alignment created by calling a function which returns a probability for the new alignment.

The second line of Table C.1 shows that the Model 4 implementation in GIZA++ is much faster, an average of 18 milliseconds is used per sentence pair, which is 46 times faster than our unoptimized Model 4 implementation. The reason for this is that GIZA++ has two optimizations which are not yet implemented in our implementation of Model 4.

The first optimization is described by Brown et al. (1993), we will call it the "Incremental Probability Calculation" optimization. Given an alignment $a$, from which we obtain the alignment $a'$ by applying a particular search operation, we can obtain $p(a', f|e)$ by a constant small number of steps. This involves starting from $p(a, f|e)$,

dividing out just the probabilities of the generative actions made to arrive at $a$ which were not made in arriving at $a'$, and multiplying in the probabilities of the generative actions made to arrive at $a'$ but not made in arriving at $a$. This is much faster than calculating $a'$ from scratch by looking up the probabilities of all of the generative actions used to obtain $a'$ (including particularly those which were the same as those used to arrive at $a$). The cost of looking up all of the probabilities is $O(l + m)$ (where $l$ is the length of the English sentence and $m$ is the length of the French sentence).

In LEAF, such procedures for updating in a constant number of steps can also be defined. We will present a very simple example in which we assume we are calculating LEAF in just the English to French direction (for ease of exposition). Suppose we have an alignment $a$ in which an English non-head word at position $i$ is in a three word English cept headed by the English head word at position $y$. The "move English non-head word to new head" search operation is used to change $e_i$ to be of word type "deleted", resulting in a new alignment $a'$. The probability $z$ of $a'$ can be quickly determined given the probability of $a$. This is done by performing the following calculations:

- $z = \text{p}(a, f|e)$
- // divide the probability of the non-head word to head word association

  $z = z/w_{-1}(y - i|\text{class}_e(e_i)$

- // divide the probability of $e_i$ being type $-1$ (non-head word)

  $z = z/g(-1|e_i)$

- // multiply the probability of $e_i$ being type $0$ (deleted)

  $z = z * g(0|e_i)$

- // $z$ is the probability of $a'$

  return $z$

For LEAF, as in the case of Model 4, this allows us to calculate the probability of $a'$ from $a$ in a small constant number of steps, rather using an $O(l + m)$ lookup of the probabilities for all of the actions. We expect that the speed up from using this optimization with LEAF is analogous to the speed up obtained when using this optimization with Model 4.

The second optimization is from the appendix of the work by Och and Ney (2003). This optimization is called "Fast Hill Climbing". If we start from an alignment $a$, we can keep a single matrix for each search operation, which will cache $p(a', f|e)/p(a, f|e)$ for alignments $a'$ reachable by applying the search operation to $a$. For instance, if we have a search operation with two arguments $i$ and $j$, a matrix $M$ indexed by the possible $i$ and $j$ values is defined. The probability of an alignment $a'$ generated by applying this search operation using arguments $i$ and $j$ is $M_{[i,j]}$ times the probability of the original

alignment $a$. Initially, all of the cells of this matrix must be calculated explicitly by calculating the costs of the alignments (using the first optimization). However, the speed up of the "Fast Hill Climbing" optimization is obtained because updating $M$ when the starting alignment $a$ is changed does not require revisiting all of the cells. Only those columns and rows for which the ratio changes need to be updated, and this is a small number of rows and columns. This means that after the matrices are initially created, search simply consists of scanning these matrices for the cell with the best probability multiplier, updating the alignment using that search operation, and updating a few of the columns and rows of the matrices. Och and Ney report a 10 to 20 times speed up in local search using this optimization.

This "Fast Hill Climbing" optimization can also be applied to LEAF. Six of the seven search operations in LEAF also have two arguments and require matrices of similar size to those required for Model 4. The seventh operation, "unlink the link between an English head word and a French head word" (operation 7 in Table 3.4 on Page 69) has three parameters, but two of these parameters are restricted to three values each, so this will be a small matrix which can be rapidly updated. The matrices required for the first six operations are $l * m$, $m^2$ or $l^2$ in size, and it is easy to see that the cost to update them will be similar to the cost to update the matrices used with Model 4. We believe search using the "Fast Hill Climbing" optimization is dominated by the time to calculate the initial matrices, where each cell must be visited. LEAF will require the

calculation of six matrices, while each Model 4 direction requires the calculation of two matrices, for a total of four matrices. Therefore we believe that the speed up obtained by using this optimization with LEAF will be about 1.5 times less than that obtained for Model 4.

By implementing these two optimizations it is clearly possible to speed up our implementation of Model 4 to match the speed of the GIZA++ implementation of Model 4. According to our empirical measurements comparing our unoptimized Model 4 implementation with GIZA++ it will be at least a 46 times speed up[1], which is close to the estimate of Och and Ney.

The third line of Table C.1 shows that the unoptimized unsupervised LEAF implementation is very slow. It is about 12 times slower than the unoptimized Model 4 implementation. The fourth line of Table C.1 shows that the unoptimized semi-supervised LEAF implementation is about 14 times slower than the unoptimized Model 4 implementation.

[1]In fact, it is likely that this speed up would be more than a 46 times speed up as long as we continue to use the Viterbi approximation in training. GIZA++ uses the "neighborhood" training approximation (Al-Onaizan et al., 1999; Och & Ney, 2003) by default (we used "neighborhood" training in all of our experiments using GIZA++). Using the neighborhood approximation requires incurring additional computational costs to those incurred in Viterbi training, see the appendix of the work by Och and Ney (2003) for the details.

As we have already discussed the optimizations required for LEAF are very similar to those used with Model 4. For LEAF, the results use the new search algorithm of Section 3.4.2.2, because the baseline search algorithm is unacceptably slow to converge. The speed ups gained by implementing the two optimizations discussed in this section apply equally to both the baseline search algorithm and the new search algorithm as the optimizations make the search operations faster and the same search operations are used in both algorithms.

In summary, we expect that the first optimization would result in an analogous speed up for LEAF search to the speed up obtained for Model 4. The speed up from applying the second optimization to LEAF would be 1/1.5 times the speed up gained for Model 4. The unoptimized Model 4 search can be sped up by at least 46 times. This implies that we can obtain at least a 30 times speed up for the LEAF search process by implementing these optimizations. We plan to implement these optimizations in future work.

## C.2 LEAF Search and Dynamic Programming

In this section we briefly consider other search algorithms reported on in the literature which we consider directly relevant to the search for the LEAF Viterbi alignment. They share the commonality that they are all based on dynamic programming.

Germann (2003) produced an impressive speed-up in local hillclimbing search for machine translation by segmenting the starting hypothesis into overlapping local areas (called tiles) which can be independently searched, and then reintegrating these partial solutions into a complete solution using dynamic programming. Such a decomposition appears to be possible for the LEAF model (though it might be more complicated in the semi-supervised case if global features such as the name transliteration sub-model are used). If such decomposition is possible this would lead to a much higher performance in search, particularly when applied in combination with our search advances and the optimizations discussed in the previous section.

We can also consider search algorithms which are quite different from the local hillclimbing search algorithms we currently use. Udupa and Maji (2005) defined a search algorithm for Model 4 which considers an exponential number of alignments in polynomial time. Eisner and Tromble (2006) presented a search algorithm for "very large neighborhood" search in machine translation which can be used to consider an exponential number of reorderings for translation in polynomial time. Both of these approaches use dynamic programming to examine a much larger space of alignments than our current search algorithms can examine. We speculate that it is possible to produce a dramatically improved search algorithm for finding the LEAF Viterbi alignment by inventing a similar approach based on dynamic programming which allows the consideration of exponentially many LEAF alignments in polynomial time.