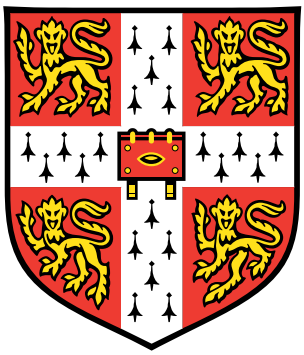
Alignment Models and Algorithms for Statistical Machine Translation

James Brunning

Cambridge University Engineering Department
and
Jesus College



August 2010

This degree is submitted to the University of Cambridge
for the degree of Doctor of Philosophy

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where stated. It has not been submitted in whole or in part for a degree at any other university.

The length of this thesis including appendices, references, footnotes, tables and equations is approximately 59,000 words.

Summary

Alignment models are used in statistical machine translation to determine translational correspondences between the words and phrases in a sentence in one language with the words and phrases in a sentence with the same meaning in a different language. They form an important part of the translation process, as they are used to produce word-aligned parallel text which is used to initialise machine translation systems. Improving the quality of alignment leads to systems which model translation more accurately and an improved quality of output. This dissertation investigates algorithms for the training and application of alignment models.

A machine translation system generates a lattice of translation hypotheses as an efficient, compact representation of multiple translation hypotheses. The alignment models are not traditionally used during translation, and we develop methods to do this using efficient lattice-to-string alignment procedures. The first part of this work deals with the use of alignment models for rescoring such lattices of hypotheses, and develops algorithms for simultaneous alignments of all hypotheses with the source sentence.

Discriminative training is a technique that has been used extensively for parameter estimation in automatic speech recognition with good results; however, it has not been used for machine translation other than to tune a small number of hyper-parameters for weighting of model features during translation. In the second part, we investigate a method of using manually-labelled word-to-word alignments of sentences for discriminative training of alignment models.

Finally, we introduce a method for the incorporation of context into alignment models. The surrounding words can influence the way in which as word translates, as can its part of speech and position in the sentence: we define an extension to existing alignment models that takes advantage of context information. Clustering algorithms are introduced to enable robust estimation of parameters. The context-dependent alignment models are evaluated by comparison with gold standard alignments, and by the translation quality of systems trained using their alignments. We find improvements in alignment quality and translation quality as a result of using our models.

Keywords: Statistical Machine Translation, Alignment Models, Hidden Markov Model, Machine Learning, Decision Tree, Context-dependence, MMIE, Expectation Maximisation

Acknowledgements

Firstly, I would like to thank my supervisor Dr Bill Byrne for his invaluable support, advice, ideas, encouragement and guidance. Thank you for always having the time to see me, helping to give direction to my work, reading drafts of the thesis, and everything else you've done for me over the course of my Ph.D.

This work would not have been possible without the support of the Schiff Foundation, whose studentship allowed me to pursue this research. Thanks also to Phil Woodland and the AGILE project for saving me from poverty in the fourth year.

I would like to thank Yonggang Deng, without whose work creating MTTK and help with initial problems I would have struggled to get started. It has been invaluable to have a well-written, extensible framework on which to build my experiments. I would like to thank Adrià de Gispert for all his help processing data and running experiments, particularly some of the translation experiments that allowed me to see if my models were working. Thanks to Gonzalo Iglesias for his work on the HiFST translation system. Yanjun Ma was a great help with part-of-speech tagging in Chinese, Lambert Mathias was very helpful with the lattice rescoring experiments and lately Juan Pino has helped with model training. Thanks to Graeme Blackwood for useful discussions throughout the course of the last four years. Thanks to Chris Longworth and Hank Liao for allowing me to use their LaTeX thesis template.

For the superb computing facilities, I am indebted to the AGILE project and the Division F computing stack. I calculated that one of my experiments, which ran overnight, would have taken 47 days to run on a single machine – without the stack of machines at my disposal, the would not have been possible. Anna Langley and Patrick Gosling, thank you for answering computer-related queries, upgrading machines and fixing things when I broke them.

I'm also very grateful to all the friends who've kept me sane over the past few years. Everyone in Jesus College Graduate Society, Jesus College Boat Club, Cambridge University Hare & Hounds, Cambridge University Modern Pentathlon Club, I've loved (almost) every minute. Alex M and Mike, Alex C and Juliet, I miss the days at #34 and I look forward to being able to spend more time with you. To Sam, Alec, Phil and Mel, you were the best housemates I could hope for when college threw us out. Emma and Zoë, your encouragement not to fail has been invaluable... massive thanks for always being there for a dose of silliness go to Chris (even though you weren't having the best time with your own work – it means a lot to me), Rich and The Horse. Thanks to Cat who's put up with me while I've been writing this up, I appreciate everything you do.

Finally, I'd like to thank my parents for putting up with my quest for a Ph.D., even though they thought I should "go out and get a real job" at times...

Acronyms

AER	Alignment Error Rate
AGILE	Autonomous Global Integrated Language Exploitation
ASR	Automatic Speech Recognition
ATM	Alignment Template Model
BLEU	BiLingual Evaluation Understudy
CFG	Context-Free Grammar
CMLE	Conditional Maximum Likelihood Estimation
CRF	Conditional Random Field
EM	Expectation Maximisation
FSA	Finite State Acceptor
FST	Finite State Transducer
HMM	Hidden Markov Model
MBR	Minimum Bayes-Risk
MADA	Morphological Analysis and Disambiguation for Arabic
MERT	Minimum Error Rate Training
MIRA	Margin Infused Relaxed Algorithm
MMIE	Maximum Mutual Information Estimation
MTTK	Machine Translation Tool Kit
SCFG	Synchronous Context-Free Grammar
SMT	Statistical Machine Translation
TER	Translation Edit Rate
TTM	Transducer Translation Model
WER	Word Error Rate
WFSA	Weighted Finite State Acceptor
WFST	Weighted Finite State Transducer

Notation

These are the terms and notation used throughout this work.

The *input language* is language from which we are translating and the *output language* is the language into which we are translating.

For generative models, we can view either language as being generated from the other; hence we define the *source language* to be the language that is viewed as the source for a generative model and the *target language* is generated from the source language. We describe sentences in each of these languages as the *source sentence* and *target sentence* respectively.

Variables, Symbols and Operations

$\mathcal{V}_{\text{src}}$	source language vocabulary
$\mathcal{V}_{\text{tgt}}$	target language vocabulary
$\mathbf{e}, e_1^I$	source sentence, i.e. a sequence of source language words
$\mathbf{f}, f_1^J$	target sentence, i.e. a sequence of target language words
$\mathbf{a}, a_1^J$	alignment sequence
$t(f\|e)$	word-to-word translation probability, i.e. probability f is generated from e
$t(f\|e,c)$	probability f is generated from e in context c
$a(j\|i,I,J)$	probability of emitting target word in position j from source word in position i under Models 1 and 2
$a(i\|i',I)$	probability of moving from state i' to state i under HMM model
$h_m(\mathbf{e},\mathbf{f})$	feature function for log-linear model
λ_m	feature weight
$\mathbf{h}$	vector of feature functions
$\mathbf{\Lambda}$	vector of feature weights

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Introduction to statistical machine translation

Statistical Machine Translation (SMT) is a technology for the automatic translation of text in one natural language into another. It translates the written form of the language but can be used in combination with speech recognition and text-to-speech synthesis to translate spoken language.

It has many possible applications and is beginning to have broad commercial and social impact. Translation is required in multilingual communities and is especially important in the European Union (E.U.), since documents and proceedings need to be translated into each of 23 (and increasing) official languages; automating translation would improve its speed and reduce expense. Translators are also used extensively in the United Nations (U.N.) and the Canadian Parliament. Machine translation can be used between any pair of languages, though there has been particular recent interest in Arabic to English and Chinese to English translation, since it also has intelligence and surveillance applications, including counter-terrorism monitoring. Automated translation can be used to provide multilingual access to the vast amount of information available on the internet, or for cross-language information retrieval. Google Translate[1] translates text or web pages between 51 different languages.

Even an imperfect translation is useful. Foreign language web sites can be translated and navigated, for example to buy tickets or reserve accommodation abroad. It can be used to get the gist of a document if a full translation is not needed, or used to determine whether it's worthy of further, more accurate translation. Machine translation can be used to help human translators, either by producing a rough translation that can be checked and then edited to

[1] `http://translate.google.com`

retain all the nuances of the original sentence or used in translation assistance software that suggests possible translations to make the translator's task easier.

Prior to statistical translation techniques, there were three approaches to machine translation (Jurafsky and Martin, 2000). In *direct* translation, the words are translated one by one using a large bilingual dictionary and simple reordering rules are applied. *Transfer* approaches rely on parsing a sentence before translation, translating the sentence structure and generating a sentence in another language. The third approach is to analyse the sentence information to form an abstract meaning representation known as an *interlingua*, before generating a sentence in another language.

Statistical approaches have a number of advantages over these non-statistical techniques. The primary advantage is that they have been shown to produce better translations. The relationships between words, phrases and grammatical structure are often ill-defined or vague, and probability distributions and statistical techniques allow us to capture this. The training procedure relies on example data and linguistic resources are not generally required. A statistical model can be trained on large amounts of data, and increasing the amount of training data will allow the model to capture more of the linguistic phenomena in the languages. Therefore, increased amounts of training data should lead to higher quality translations. The statistical framework also allows translation to be combined with other statistical techniques such as speech recognition.

A further benefit of the statistical techniques developed is that they need not rely on particular features of the languages involved, such as language-specific models of translation or grammar. Many features of the translation models are language-independent, and can be tuned for particular language pairs by the estimation of model parameters. This enables machine translation systems to be built for multiple language pairs with minimal modification to the technique. For increased quality of translation, specific knowledge of the languages involved is often needed: statistical models have been developed to incorporate additional language-specific information relatively easily, including morphological features, re-ordering and grammatical models. For these reasons, we concentrate on statistical techniques.

1.2 Problem definition

The fundamental aim of statistical machine translation is to take a fragment in one written language and translate it into another written language. A fragment can be anything from a single transcribed utterance to a book, document, newspaper or web page. We translate at the sentence level, viewing an input sentence as a sequence of words $\mathbf{f}$ and transforming it via the use of statistical models into a sequence of words in the output language $\mathbf{e}$ that represents an accurate translation of the original sentence. There is no unique solution, but we aim to produce a sentence that is fluent and grammatically correct in the output language that has the same meaning as the original sentence. We build probabilistic models and choose a translation $\hat{\mathbf{e}}$ that maximises $\mathrm{P}(\mathbf{e}|\mathbf{f})$, the probability of output sentence $\mathbf{e}$ given input sentence $\mathbf{f}$.

The desire to use computers to translate between human languages goes back decades, with Weaver (1955) expressing a desire to translate between written languages and suggesting that the problem of translation could be treated as a problem in cryptography. He writes:

> When I look at an article in Russian, I say: "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode."

Inspired by this, influential research was carried out by IBM (Brown et al., 1993) to formulate a source-channel model where it is assumed for modelling purposes that $\mathbf{f}$ is the result of passing $\mathbf{e}$ through the noisy channel defined by $\mathrm{P}(\mathbf{f}|\mathbf{e})$, and we wish to recover the original sentence $\mathbf{e}$ from the observed sentence $\mathbf{f}$. Using Bayes' Theorem, we can re-write the our aim as the finding of

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}}\, \mathrm{P}(\mathbf{e}|\mathbf{f}) = \underset{\mathbf{e}}{\operatorname{argmax}}\, \mathrm{P}(\mathbf{e})\mathrm{P}(\mathbf{f}|\mathbf{e}) \tag{1.1}$$

where $\mathrm{P}(\mathbf{e})$ is the *language model* and $\mathrm{P}(\mathbf{f}|\mathbf{e})$ is the *translation model*. This process is known as *decoding*. For a more detailed history of machine translation and the systems involved, see Koehn (2009).

We train the translation model on corpora of parallel text, i.e. sentences that have same meaning in each language, which are obtained from a number of sources of manually translated documents. We use parliamentary proceedings in multilingual communities such as the European Union, United Nations and Canadian Parliament. News articles that have been translated into other languages are used; in particular, the Xinhua news agency publishes 15,000 news articles per day in Chinese, English, Spanish, French, Russian and Arabic[1]. We also use bilingual data gathered from web pages. Generally, the more training data available for a given language pair, the better the quality of translation will be, as long as the data is of reasonable quality. We aim to be able to translate any text, so coverage of a wide variety of domains is essential. A large amount of monolingual data in the output language is needed to train the language model, which encourages the translation system to produce correct sentences in the output language.

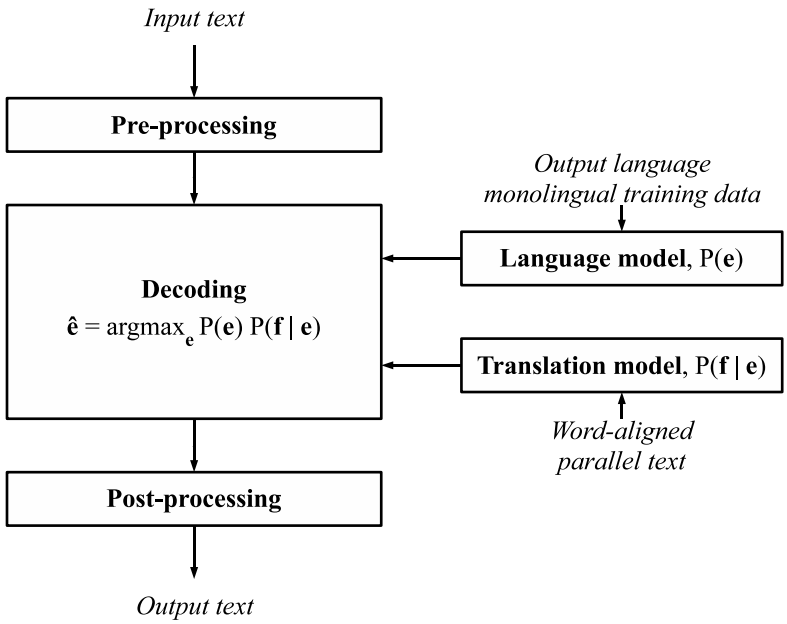


Figure 1.1: Graphical representation of noisy channel models and decoding

[1] http://news.xinhuanet.com/english/2007-08/31/content_6637522.htm

1.3 Alignment modelling for SMT

The usual approach to building a statistical machine translation system is to first build a model of alignment between the input and output languages. We use this to determine translational correspondences between the words and phrases of a sentence in one language with the words and phrases of a sentence in another language. We can view alignment as a hidden variable and build models to determine this alignment; we usually then use the models to output the best set of alignment links for each sentence in the training corpus. From these alignment links, we can extract phrases for use in translation and rules for translating those phrases.

Alignment models are currently used by almost all translation systems. They are used by phrase-based systems for extracting phrase pairs from training data and building tables of possible translations of a phrase, which are then used for generation and scoring of hypotheses. Syntax-based and hierarchical phrase-based systems also initialise their translation models from word-aligned parallel text. More accurate alignments between the parallel text should lead to better translation systems being built. Therefore, the development and training of alignment models is very important to the quality of translation.

This thesis focuses on alignment models for machine translation, and investigates algorithms used for training them. It looks at the discriminative training of alignment models to improve the quality of word-level alignments within sentences, and develops new methods of integrating word context into the alignment model. It also looks at methods of improving translation quality by including alignment models as a feature in the translation system.

1.4 Thesis structure

This chapter briefly introduces statistical machine translation and the problem of aligning sentences of parallel text. Chapter 2 presents an introduction to the models used for SMT. The particular translation systems used in this thesis are introduced, and methods of evaluating the output of a translation system are described. Chapter 3 discusses alignment models in detail, since the remainder of the work will involve these. We introduce the IBM series of models and hidden Markov models of alignment on which we perform experiments, as well as describing some of the other types of alignment model that appear in existing machine translation literature.

A feature of many existing machine translation systems is that the output from an alignment model is used to initialise the system, but information from these alignment models is not used further during translation. Many translation systems can produce a lattice of translation hypotheses, which is a compact and efficient method of representing many possible translations, and we wish to use alignment models to provide additional information (that is not captured by other aspects of the model) to improve translation quality. Chapter 4 describes a method by which a complete lattice of translation hypotheses can be aligned to the input sentence, and an algorithm for representing every alignment of every hypothesis using a different structure of lattice. We introduce an algorithm for using alignment models for rescoring lattices of translation hypotheses using certain alignment models. The use of alignment models for rescoring is investigated in Chapter 5.

Discriminative training has been extensively used for training the acoustic models used in automatic speech recognition. Prior to this work, discriminative training had only been used

for estimating small numbers of parameters used as weights for features in the translation system. We wish to use manually-produced alignments of parallel text to improve the quality of output from alignment models. We introduce algorithms for the discriminative training of a large number of alignment model parameters in Chapter 6. The discriminatively trained models are evaluated in Chapter 7, showing that discriminative training leads to improved alignment quality.

The area of this work that continues to provide the most consistent gains in translation quality is the addition of context-dependence to alignment models described in Chapter 8. Words translate differently in different contexts and existing alignment models do not adequately model this. We describe a novel method of adding context information to alignment models and introduce algorithms for clustering of word contexts to ensure robust parameter estimation. In Chapter 9, we carry out thorough investigation of context-dependent alignment models. We evaluate the models by comparison of the alignments they produce with manually-produced alignments; we then evaluate the translation systems built from the alignments using an automated metric for evaluation of translation quality.

Finally, Chapter 10 describes conclusions reached and suggests directions for future work.

Chapter 2
Statistical Machine Translation

2.1 Statistical machine translation models

The objective of machine translation is to translate from an input sentence $\mathbf{f}$ in one language to an output sentence $\mathbf{e}$ in another language that has the same meaning as $\mathbf{f}$. We do this by building statistical models to represent the translation process, and find

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \, \mathrm{P}(\mathbf{e}|\mathbf{f}). \tag{2.1}$$

Brown et al. (1993) introduced the source-channel model, where the output language sentence $\mathbf{e}$ is viewed as being generated by a source with probability $\mathrm{P}(\mathbf{e})$ defined by the *language model*; this is then passed though the translation channel to produce input language sentence $\mathbf{f}$, according to the *translation probability* $\mathrm{P}(\mathbf{f}|\mathbf{e})$. The task of a translation system is to determine $\mathbf{e}$ from the observed sentence $\mathbf{f}$. The best translation is found by computing

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \, \frac{\mathrm{P}(\mathbf{f}|\mathbf{e})\mathrm{P}(\mathbf{e})}{\mathrm{P}(\mathbf{f})} = \underset{\mathbf{e}}{\operatorname{argmax}} \, \mathrm{P}(\mathbf{f}|\mathbf{e})\mathrm{P}(\mathbf{e}). \tag{2.2}$$

A large amount of data is required to accurately train these models; however they can be trained separately. Estimation of the translation probability $\mathrm{P}(\mathbf{f}|\mathbf{e})$ requires training on bilingual parallel text. The amount of text in electronic parallel corpora that can be used for this purpose is rapidly increasing: Chinese news is available in a wide variety of languages and much E.U. and U.N. data is stored electronically. The language model $\mathrm{P}(\mathbf{e})$ is trained

over large monolingual corpora in the required language, since this allows us to represent the language as accurately as possible. Generally, the more training data we have, the more accurate our representation of the language will be.

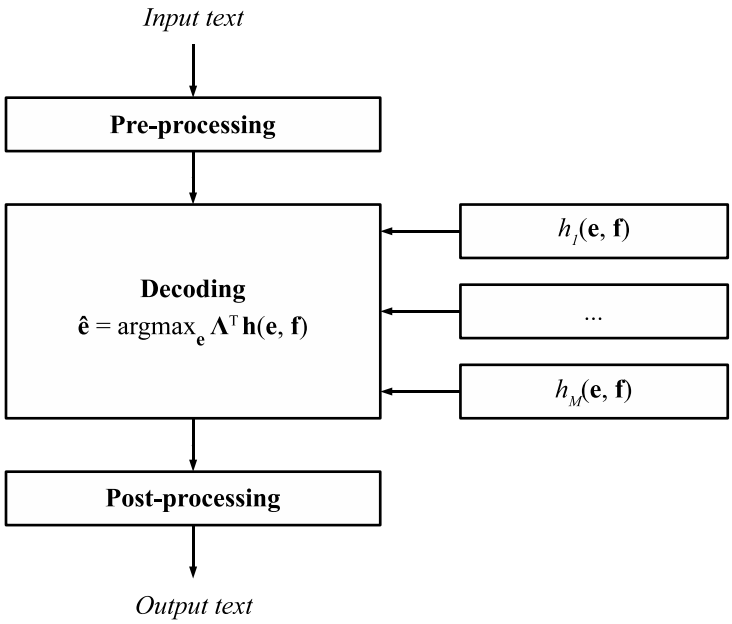


Figure 2.1: The decoding process, representing how the translation $\hat{\mathbf{e}}$ *is found using model features*

The current dominant approach is to use a log-linear combination of feature functions to model directly the posterior probability $\mathrm{P}(\mathbf{e}|\mathbf{f})$. We define M *feature functions* $h_m(\mathbf{e}, \mathbf{f}), m = 1, \ldots, M$, and the probability of generating $\mathbf{e}$ from $\mathbf{f}$ is given by

$$\begin{aligned} \mathrm{P}_{\boldsymbol{\Lambda}}(\mathbf{e}|\mathbf{f}) &= \frac{\exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}))}{\sum_{\mathbf{e}'} \exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}', \mathbf{f})} \\ &= \frac{\exp(\boldsymbol{\Lambda}^{\mathrm{T}} \mathbf{h}(\mathbf{e}, \mathbf{f}))}{\sum_{\mathbf{e}'} \exp(\boldsymbol{\Lambda}^{\mathrm{T}} \mathbf{h}(\mathbf{e}', \mathbf{f}))}, \end{aligned} \tag{2.3}$$

where each feature has a corresponding *feature weight* λ_m. We write $\boldsymbol{\Lambda} = (\lambda_1 \ldots \lambda_M)^{\mathrm{T}}$ for the vector of feature weights and $\mathbf{h}$ for the vector of features. The process of finding the best translation (i.e. the one with the highest probability) is known as *decoding*. We choose the translation $\hat{\mathbf{e}}$ that maximises $\mathrm{P}_{\boldsymbol{\Lambda}}(\mathbf{e}|\mathbf{f})$, i.e.

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}}\, \boldsymbol{\Lambda}^{\mathrm{T}} \mathbf{h}(\mathbf{e}, \mathbf{f}) \tag{2.4}$$

The decoding process is shown graphically in Figure 2.1. Note that this is equivalent to the source-channel model if we set $M = 2, \lambda_1 = \lambda_2 = 1$ and use feature functions

$$h_1(\mathbf{e}, \mathbf{f}) = \log \mathrm{P}(\mathbf{e}) \tag{2.5}$$

$$h_2(\mathbf{e}, \mathbf{f}) = \log \mathrm{P}(\mathbf{f}|\mathbf{e}) \tag{2.6}$$

Log-linear models were first used for machine translation by Berger et al. (1996), but were improved and made popular by Och and Ney (2002) and Och (2003).

2.1.1 Generation of hypotheses and decoding

Hypothesis sentences are generated from the input sentence by a generative translation model that depends on the particular translation system being used. While in theory we could hypothesise any sequence of words in the output language and allow the model to decide on the best translation, we consider only the sequences that could be reasonably expected to be translations based on what has been seen in the training data. Therefore, we learn rules or phrase tables from the parallel text. For a given input sentence, we only consider output words that have been seen in the parallel text as translations of the input words.

A large problem in the decoding process is the computational complexity. The *search space* is the set of possible hypotheses during translation, and can become very large for a number of reasons. We may have words in the output sentence that are not a direct translation of any words in the input sentence, so the models must allow for these to be inserted. Similarly, the models must allow the deletion of input words with no direct translation. Words and phrases in the input sentence may translate in many different ways; algorithms must be devised to find the best hypotheses in the search space efficiently. Decoding typically makes use of dynamic programming algorithms and efficient search procedures such as beam search and A* search.

2.1.2 Features for log-linear models

An important consideration is the derivation of features for the translation system. We require features that are complex enough that they model the translation process accurately and provide useful information, but also require that they can be computed efficiently and can be handled by the algorithms used for decoding.

One feature common to all translation systems is the output language model $\log \mathrm{P}(\mathbf{e})$, which is discussed further in Section 2.4. It is not uncommon to use more than one language model feature, with a relatively simple language model being used during a first pass of decoding and a more accurate, but a computationally more expensive, language model being used for rescoring a lattice of hypotheses to refine the output (Brants et al., 2007).

We can define features that depend on any property of the two sentences, and features can have varying complexity. Part-of-speech taggers can be run, grammatical dependencies can be analysed, parses of the sentences can be considered. Features from generative models $\mathrm{P}(\mathbf{e}|\mathbf{f})$, $\mathrm{P}(\mathbf{f}|\mathbf{e})$ in either translation direction can be used. Other, simpler features used include: sentence length distributions that condition the length of the output sentence on that of the input sentence; distortion models that consider the reordering of words and phrases; features which reward word pairs found in conventional bilingual dictionaries and lexical features which reward word pairs found in training data. A *word insertion penalty*, i.e. a cost that is added for every word in the sentence, is effective in controlling output length and makes a large difference to translation quality (Koehn, 2009). See Och et al. (2004) for a description of many different features.

2.2 Model training

Once we have defined our model $\mathrm{P}_{\mathbf{\Lambda}}(\mathbf{e}|\mathbf{f})$ by deciding on features, we need to estimate the model parameters from training data. This is typically done in two steps. The first step estimates parameters for the features that make up the model from a large corpus of training data. Most features are estimated on a corpus of parallel text $\{(\mathbf{e}_s, \mathbf{f}_s)\}_{s=1}^{S}$, though language models and other features for which $h(\mathbf{f}, \mathbf{e}) = h(\mathbf{e})$, are estimated on the output language only.

The second step is the estimation of the feature weights $\mathbf{\Lambda}$ of the log-linear model. Discriminative training is performed on a development set smaller than the parallel corpus, $\{(\mathbf{e}_s, \mathbf{f}_s)\}_{s=1}^{S'}$, and usually translates the source text to minimise the error (under some loss function) compared to a reference translation.

The models can be used to translate many different types of data, from newswire to web data, to the output from a speech recognition system run on television broadcasts, with each type having different properties desired from its output. Discriminative training can be used to help with domain adaptation, for example to give greater importance to a language model corresponding to the desired domain and modify the insertion penalty. The development set is matched to the text the model will be used to translate, and the parameters are tuned to optimise the system's performance on that type of data.

2.2.1 Discriminative training of log-linear models

These discriminative models are trained by optimising the parameters $\mathbf{\Lambda}$ to maximise some objective function. There are many possibilities for the objective function; the challenge is to find an objective function that is feasible to compute and improves translation quality.

First attempts at performing discriminative training optimised the feature weights to maximise the likelihood of the entire parallel text:

$$\hat{\mathbf{\Lambda}} = \underset{\mathbf{\Lambda}}{\operatorname{argmax}} \sum_{s=1}^{S} \mathrm{P}_{\mathbf{\Lambda}}(\mathbf{e}_s|\mathbf{f}_s) \tag{2.7}$$

This is known as maximum entropy training since the maximum likelihood estimate of $\mathbf{\Lambda}$ produces the model which has maximum entropy on the training data consistent with the constraints placed on the model. Berger et al. (1996) use an improved method of Generalised Iterative Scaling (GIS) (Darroch and Ratcliff, 1972). The number of features is limited since they add features one by one according to which one gives the largest increase in likelihood. However, calculation of the denominator of Equation (2.3) is computationally intensive, since it requires a sum over all possible translations $\mathbf{e}$, and many iterations of training are required. Och and Ney (2002) also use GIS but approximate the denominator by the sum over an N-best list from the previous output to improve performance. Since some of the sentences may have more than one reference translation, a modified training objective is introduced. This assigns an equal weight to each reference sentence where there are multiple references. Some reference sentences may not appear in the N-best list, so the reference for training purposes is the sentence from the N-best list which contains the minimal number of word errors relative to the reference translations.

Optimisation of the likelihood of training data does not necessarily guarantee improved translation quality on unseen text, which is our ultimate aim. There is a range of evaluation metrics for machine translation output (see Section 2.8 for details), which we can use for discriminative training; we pick the metric with which we want to evaluate the system, and optimise according to this metric during discriminative training. Och (2003) introduces Minimum Error Rate Training (MERT), which defines an error function $E(\mathbf{r}, \mathbf{e})$ measuring the error in sentence $\mathbf{e}$ relative to a reference sentence $\mathbf{r}$; we attempt to minimise the error function over the development corpus. The total error for a set $\mathbf{e}_1^{S'} = \{\mathbf{e}_1, \ldots, \mathbf{e}_{S'}\}$ of translation hypotheses with references $\mathbf{r}_1^{S'} = \{\mathbf{r}_1, \ldots, \mathbf{r}_{S'}\}$ is the sum of the errors of the individual sentences:

$$E(\mathbf{r}_1^{S'}, \mathbf{e}_1^{S'}) = \sum_{s=1}^{S'} E(\mathbf{r}_s, \mathbf{e}_s). \tag{2.8}$$

For a corpus $\mathbf{f}_1^{S'}$ of sentences representative of the data we wish to translate, assume that for each sentence $\mathbf{f}_s$ we have a set of candidate translations C_s. We find

$$\hat{\mathbf{\Lambda}} = \underset{\mathbf{\Lambda}}{\operatorname{argmin}}\, \mathcal{F}_{\mathrm{MERT}}(\mathbf{\Lambda}), \tag{2.9}$$

where the objective function $\mathcal{F}_{\mathrm{MERT}}(\mathbf{\Lambda})$ is

$$\mathcal{F}_{\mathrm{MERT}}(\mathbf{\Lambda}) = \sum_{s=1}^{S'} E(\mathbf{r}_s, \hat{\mathbf{e}}(\mathbf{f}_s; \mathbf{\Lambda})) \tag{2.10}$$

and $\hat{\mathbf{e}}(\mathbf{f}_s; \mathbf{\Lambda})$ is the candidate translation assigned highest probability by the model:

$$\hat{\mathbf{e}}(\mathbf{f}_s; \mathbf{\Lambda}) = \underset{\mathbf{e} \in C_s}{\operatorname{argmax}}\, \mathbf{\Lambda}^{\mathrm{T}} \mathbf{h}(\mathbf{e}, \mathbf{f}_s). \tag{2.11}$$

More generally, we may wish to minimise the expected error over all hypotheses in the set of candidate translations by performing MBR training of the feature weights, i.e. we use objective function

$$\mathcal{F}_{\mathrm{MBR}}(\mathbf{\Lambda}) = \sum_{s=1}^{S'} \sum_{\mathbf{e} \in C_s} E(\mathbf{r}_s, \mathbf{e}) \mathrm{P}_{\mathbf{\Lambda}}(\mathbf{e}|\mathbf{f}). \tag{2.12}$$

Liu et al. (2007) use this approach for discriminative adaptation of language models for ASR and SMT. However, the approach has not proved successful for optimisation of feature weights for machine translation (Mathias, 2007); we therefore use MERT as presented by Och (2003).

Optimisation is carried out using a line search algorithm that relies on the fact that the error is piecewise constant along a line through the parameter space, which simplifies computation. The set of candidate translations C_s is chosen to ensure the error does not increase on the training corpus, using an iterative procedure that incrementally adds new candidate hypotheses to C_s. At the first iteration, it contains the N-best translations according to the model; then the parameters Λ are re-estimated and the N-best translations according to the re-estimated model are added. This is repeated until C_s is unchanged by this process; at this point we know the model will not assign high probability to poor quality sentences not in C_s.

The principal advantages of MERT are that it does not require computation of the denominator in Equation (2.3), and that it optimises the translation quality metric directly. It has been used extensively for machine translation, and many efforts have been made to improve

the algorithm. Random restarts of the MERT algorithm, and the initialisation of parameters using random walks from the previous optimum, have been found to improve training time and the translation quality of the resulting model (Moore and Quirk, 2008).

Macherey et al. (2008) develop a lattice-based method of performing MERT rather than using an N-best list, reporting improved speed of convergence and modest gains in translation quality. Duh and Kirchhoff (2008) address under-fitting to development data by using boosting to combine multiple log-linear models generated by MERT to re-rank N-best lists. Cer et al. (2008) present a stochastic method for optimisation of $\mathbf{\Lambda}$ and use smoothing to improve generalisation performance (the intuition being that a local minimum in error rate may be surrounded by an area of high error rate so the system is not robust to small changes in parameters).

Other efforts have concentrated alternative discriminative training techniques to replace MERT (Blunsom et al., 2008; Turian et al., 2006). The error surface is piecewise constant, contains many local minima and is not differentiable with respect to the model parameters, which poses problems for parameter estimation. Smith and Eisner (2006) overcome this by using deterministic annealing to minimise the expected loss under a probability distribution over the hypotheses that is gradually sharpened to become the true error function as training progresses.

A further drawback of MERT is that it is limited by the number of feature weights it can optimise reliably (Chiang et al., 2008b). The Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006) has been used to perform discriminative training of larger numbers of parameters (Arun and Koehn, 2007; Watanabe et al., 2007), with Chiang et al. (2008b) using it to optimise weights for 34 features based on 'soft syntactic constraints' for hierarchical translation, which reward translations that respect syntactic structure of the sentence (Chiang, 2005; Marton and Resnik, 2008). Most recently, MIRA has been used for training systems that use thousands of features (Chiang et al., 2009).

2.3 Alignment modelling

In Automatic Speech Recognition (ASR), a transcription $\mathbf{e}$ is estimated from acoustic data $\mathbf{O}$ using the following formula (Jurafsky and Martin, 2000):

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \, \mathrm{P}(\mathbf{O}|\mathbf{e})\mathrm{P}(\mathbf{e}) \tag{2.13}$$

This can be viewed as a source-channel model, with the source word sequence $\mathbf{e}$, modelled by the language model $\mathrm{P}(\mathbf{e})$, being passed through a noisy channel to produce $\mathbf{O}$: our task is to determine $\mathbf{e}$ given only the information contained in $\mathbf{O}$; we use complex statistical models whose parameters are estimated from training data to ensure they accurately represent that data.

The translation model $\mathrm{P}(\mathbf{f}|\mathbf{e})$ can be viewed as an analogue of the acoustic model $\mathrm{P}(\mathbf{O}|\mathbf{e})$. In machine translation, however, there is an additional level of complexity that does not occur in ASR. Whereas the words in the ASR transcription $\mathbf{e}$ occur in the same order in which they are uttered, and therefore the same order in which their representations appear in $\mathbf{O}$, the ordering of words in the input and output languages need not be the same. It is necessary to find the alignment of words in order to find their translations. Alignment is considered at three levels: document (determining which input language document is a

translation of which output language document); sentence (determining a correspondence between the sentences within the document); and word/phrase level. Deng (2005a) considers the problem of determining alignment at the sentence level given equivalent documents, but we concentrate on determining word and phrase level alignments once sentence pairs have been matched.

We use a generative model of alignment and view $\mathbf{f}$ as being generated from $\mathbf{e}$ (Brown et al., 1993). For alignment, the source language is the language we generate from and the target language is the language that is generated. We can use alignment models in both translation directions, i.e. we can view the input language sentence as being generated from the output language sentence, and vice versa. Given a source language sentence $\mathbf{e}$ and target language sentence $\mathbf{f}$, we introduce a hidden alignment variable A which determines the correspondence between the words and phrases in the two sentences, and calculate the translation probability as

$$\mathrm{P}(\mathbf{f}|\mathbf{e}) = \sum_{A} \mathrm{P}(\mathbf{f}, A|\mathbf{e}), \tag{2.14}$$

where the sum is taken over all possible alignments A.

The number of possible alignments between $\mathbf{e}$ and $\mathbf{f}$ is $2^{|\mathbf{e}||\mathbf{f}|}$ and increases exponentially as the sentence lengths increase. This poses a problem for modelling due to the complexity of the models involved. The area of alignment modelling is concerned with developing techniques to overcome these problems and work out the translational correspondences between sentences. Alignment modelling is described in more detail in Chapter 3.

2.4 Language modelling

An important part of the translation system is the modelling of the output language, since it allows us to favour directly translation hypotheses that are grammatical sentences. We assume that the sentence is generated left to right and that each word depends on the previous words, i.e. the probability of sentence $\mathbf{e} = e_1, e_2, \ldots, e_I = e_1^I$ is given by

$$\mathrm{P}(e_1^I) = \prod_{i=1}^{I} \mathrm{P}(e_i|e_1, e_2, \ldots, e_{i-1}). \tag{2.15}$$

For computational reasons, we make the approximation that each word depends only on $n-1$ previous words; this is known as an *n-gram language model* (Chen and Goodman, 1999; Kneser and Ney, 1995). The sentence probability is approximated by

$$\mathrm{P}(e_1^I) \approx \prod_{i=1}^{I} \mathrm{P}(e_i|e_{i-n+1}, \ldots, e_{i-1}), \tag{2.16}$$

and the n-gram probabilities $\mathrm{P}(e_i|e_{i-n+1}^{i-1})$ are estimated from large amounts of monolingual data in the output language. The maximum likelihood estimate of $\mathrm{P}(e_i|e_{i-n+1}^{i-1})$ is

$$\mathrm{P}(e_i|e_{i-n+1}) = \frac{c(e_{i-n+1}^{i})}{c(e_{i-n+1}^{i-1})}, \tag{2.17}$$

where c are the counts of the word sequences in training data. These maximum likelihood estimates can suffer from data sparsity, i.e. they are inaccurate when there are few examples

of an n-gram in the training data. They also assign zero probability mass to n-grams or words that do not occur in training data, whereas we do not want to rule out the possibility of these occurring. Various smoothing methods are employed to adjust the maximum likelihood estimates to produce more accurate probability distributions (Chen and Goodman, 1999; Kneser and Ney, 1995). We can use interpolation of probabilities where a lower order distribution is interpolated with a higher order distribution; *backoff*, is the use of a lower order n-gram probability when a higher order n-gram is not available; *discounting* subtracts from non-zero counts in order that probability mass can be distributed among n-grams with zero count, usually according to a lower order distribution. The general form of a large number of language models is:

$$\mathrm{P}_{\mathrm{smooth}}(e_i|e_{i-n+1}^{i-1}) = \begin{cases} \rho(e_i|e_{i-n+1}^{i-1}), & \text{if } c(e_{i-n+1}^{i}) > 0 \\ \gamma(e_i|e_{i-n+1}^{i-1})\mathrm{P}_{\mathrm{smooth}}(e_i|e_{i-n+2}^{i-1}), & \text{otherwise} \end{cases}, \tag{2.18}$$

for some probability distribution ρ defined over n-grams that occur in the training data, where $\gamma(e_i|e_{i-n+1}^{i-1})$ is the *backoff weight* and is normalised to ensure a valid probability distribution. All such models require significant computation to ensure probabilities are normalised.

The largest language models (as of 2007) were built on 2 trillion (2×10^{12}) words of data and contain 200 billion distinct n-grams, using *stupid backoff*, a scheme that assigns a score to each word sequence but does not require normalisation of probabilities (Brants et al., 2007). The stupid backoff score is

$$S(e_i|e_{i-n+1}^{i-1}) = \begin{cases} \frac{c(e_{i-n+1}^{i})}{c(e_{i-n+1}^{i-1})}, & \text{if } c(e_{i-n+1}^{i}) > 0 \\ \alpha_n S(e_i|e_{i-n+2}^{i-1}), & \text{otherwise} \end{cases}, \tag{2.19}$$

where α_n is a backoff weight that can depend on the n-gram order. The main benefit of this scheme is that training is efficient due to the fact that normalisation of probabilities is not performed. Despite its simplicity, it is competitive with more sophisticated methods, especially as the amount of training data increases, and can be trained on large amounts of data where other models would be too complex.

2.5 Phrase-based translation

Initial statistical models of translation dealt only with correspondences between pairs of words in the two languages, i.e. the units of translation were single words. However, some sequences of words tend to translate as a unit, so it is desirable to think of the sentences in terms of *phrases*, i.e. sequences of consecutive words. *Phrase-based* translation makes use of these so a group of contiguous words in one language can translate as a contiguous sequence of words in the other language. This also enables the context of a word to have an effect, and local changes in word order between languages can be learned. In addition, the phrases are extracted from real text so words are likely to be grammatical within a phrase and the system can provide more fluent translations.

Och et al. (1999) introduce the Alignment Template Model of machine translation, which is a generative model that takes into account phrase structure during translation. Each word is assigned to a class given by an automatically trained method (Och, 1999) and an *alignment template* defines the alignment between a source class sequence and target class sequence. In translation, the alignment template is used to determine the ordering of a source phrase in

the target language, while the words within the phrase are translated using a word-to-word translation model.

Most phrase-based translation systems model the generation of hypotheses in a similar way (Koehn, 2009). Firstly, the input sentence is segmented into phrases, where a phrase is usually simply a sequence of words and is not syntactically motivated. The phrases are then translated independently into the output language, according to a phrase translation table estimated from word-aligned parallel text. Insertion of phrases is permitted to allow for phrases in the output language that are not direct translations of phrases in the input language; similarly, deletion of phrases is allowed for input phrases that have no translation in the output language.

A *distortion model* is applied to account for changes in phrase order between the input and output languages. The distortion models used during decoding are simple due to the computational expense of allowing all possible permutations of phrases: it has been shown that the search for the best permutation is NP-complete (Knight, 1999). Some phrase-based systems allow only monotonic alignments (Banchs et al., 2005; Zens and Ney, 2004), which leads to fast decoding but causes problems for languages with different word orders (Lopez, 2008).

The Pharaoh decoder (Koehn, 2004) is a phrase-based decoder that makes use of log-linear models. It performs efficient decoding using a beam search and can output lattices of translation hypotheses. Moses (Koehn et al., 2007) is an open source decoder with the functionality of Pharaoh, but also allows factored translation models where the text itself is augmented with additional information such as part of speech tags or other morphological, syntactic or semantic information. Confusion network decoding allows it to take ambiguous input, such as the output from a speech recogniser (Bertoldi et al., 2007). The distortion model used by Moses allows positioning of a phrase relative to the last word in the previous phrase, independent of the content of the phrases, and sets a *distortion limit* for the maximum distance a phrase is allowed to move. The cost of a reordering is defined to be the sum of the individual jump sizes; this cost is used as a feature in the log-linear model.

2.5.1 The Transducer Translation Model

The Transducer Translation Model (TTM), introduced by Kumar et al. (2006), is a phrase-based model for parallel text alignment and translation implemented using standard Weighted Finite State Transducer (WFST) operations. WFSTs are used extensively in speech recognition systems, and efficient algorithms have been developed for their manipulation (Mohri et al., 1997); hence the translation process can be implemented efficiently without the need for specialised decoders or dynamic programming techniques. Generation of N-best lists is also possible using standard WFST operations.

The entire translation process is formulated as a WFST (Blackwood et al., 2008a; Kumar and Byrne, 2003; Kumar et al., 2006). We view the input sentence as being generated from the output sentence by the model, and therefore refer to the input sentence as the *target* and the output language as the *source*. The generation proceeds as follows: first, the source sentence e_1^I is segmented into phrases u_1^K, before these phrases are translated into the target language to give phrases x_1^K and reordered, yielding y_1^K (Kumar and Byrne, 2005). We wish to allow some target phrases to be spontaneously generated rather than being direct translations of source phrases, so we allow the insertion of target phrases to give a sequence c_0^K of phrases with insertion markers appended; phrases are then inserted according to the markers to give

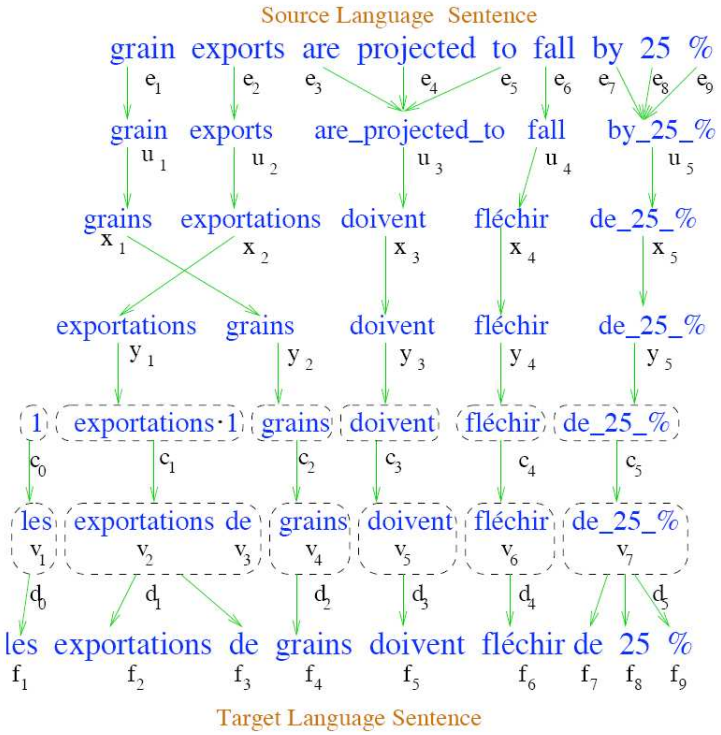


Figure 2.2: The TTM translation process (Kumar and Byrne, 2005)

phrases d_0^K. The phrases d_0^K are themselves made up of target language phrases v_1^R, which are then concatenated to form the target sentence f_1^J. Figure 2.2 shows how this occurs for an example sentence.

The translation is broken down as follows:

$$
\begin{aligned}
& \mathrm{P}(f_1^J, v_1^R, d_0^K, c_0^K, a_1^K, u_1^K, e_1^I) = \\
& \mathrm{P}(e_1^I) && \text{Source language model} \\
\times\ & \mathrm{P}(u_1^K, K | e_1^I) && \text{Source phrase segmentation} \\
\times\ & \mathrm{P}(y_1^K, x_1^K | u_1^K, K, e_1^I) && \text{Phrase translation and reordering} \\
\times\ & \mathrm{P}(v_1^R, d_0^K, c_0^K | y_1^K, x_1^K, u_1^K, K, e_1^I) && \text{Target phrase insertion} \\
\times\ & \mathrm{P}(f_1^J | v_1^R, d_0^K, c_0^K, y_1^K, x_1^K, u_1^K, K, e_1^I) && \text{Target phrase segmentation}
\end{aligned}
$$

The TTM model can be broken down into distinct component parts, allowing each one to be specified and worked on separately: a separate FST is built to model each of these processes. The *language model* G determines whether a given sequence of source language words is likely to be a correct, grammatical and coherent sentence in the source language. The n-gram language models described in Section 2.4 can be expressed as WFSTs (Allauzen et al., 2003), and are used here. The *source phrase segmentation* transducer W maps source word sequences to sequences of phrases depending on those phrases available for translation. A *phrase transduction model* maps source language phrases to target language phrases, and a *reordering model* acts on the phrases once they have been translated to put them in target language order; these two operations are carried out by transducer R. To allow for the possibility that there are words or phrases in the target sentence that are not a direct translation of words or phrases in the source sentence, we introduce a *phrase insertion transducer* Φ that

performs the spontaneous insertion of arbitrary target phrases in the sentence. The *target phrase segmentation* transducer Ω maps target phrase sequences to target sentences. Finally, we must ensure that the sequence of phrases agrees with the words in the target sentence, so we introduce the *target sentence acceptor* T.

The component parts of the model are as follows:

- **Source Phrase Segmentation Model:** This component finds all possible segmentations of the source sentence into phrases which appear in the inventory of source language phrases $\mathcal{P}_{\text{src}}$. We assume that the number of phrases depends only on the length of the sentence and write

$$\mathrm{P}(u_1^K, K|e_1^I) = \mathrm{P}(u_1^K|K, e_1^I)\mathrm{P}(K|I). \tag{2.20}$$

  The distribution for the number of phrases given the number of words is chosen to be uniform:

$$\mathrm{P}(K|I) = \frac{1}{I} \text{ for } K \in \{1, \ldots, I\}. \tag{2.21}$$

  Given the number of phrases K, we assume that every segmentation with that number of phrases that can be obtained using the inventory is equally likely, i.e.

$$\mathrm{P}(u_1^K|K, e_1^I) = \begin{cases} C, & u_1^K = e_1^I \text{ and } u_k \in \mathcal{P}_{\text{src}} \text{ for all } k \in \{1, \ldots, K\} \\ 0, & \text{otherwise} \end{cases} \tag{2.22}$$

  C is a constant chosen so that $\sum_{u_1^K} \mathrm{P}(u_1^K|K, e_1^I) = 1$, i.e. the distribution is normalised.

  An unweighted *phrase segmentation transducer* W maps source word sequences to source phrase sequences. It outputs all $u_1^K = e_1^I$ such that u_k appears in the source phrase inventory for all k. A weighted finite state transducer for the distribution $\mathrm{P}(u_1^K|K, e_1^I)$ is constructed, ensuring that $\sum_{u_1^K} \mathrm{P}(u_1^K|K, e_1^I) = 1$ for each K, e_1^I. For further details on the construction of this WFST, see (Kumar and Byrne, 2003).

- **Phrase Translation and Reordering:** The *Phrase Transduction Model* maps the source language phrases u_k to target language phrases x_k. We assume that the target language phrases are conditionally independent and depend only on the source language phrase that generated them, i.e.

$$\begin{aligned} \mathrm{P}(x_1^K|u_1^K, K, e_1^I) &= \mathrm{P}(x_1^K|u_1^K) \\ &= \prod_{k=1}^{K} \mathrm{P}(x_k|u_k). \end{aligned} \tag{2.23}$$

  The phrase translation probabilities $\mathrm{P}(x|u)$ are estimated from the relative frequencies of phrase translations found in parallel text alignments.

  Once the phrases have been translated, they are then reordered into target language order. (Kumar and Byrne, 2005) introduces Maximum Jump phrase reordering models that attempt to address the balance between accurate modelling of the translation process and computational complexity.

  The reordering models act on a target language phrase sequence x_1^K in source language phrase order, and map it to the target language phrase order y_1^K. The movement of the

phrases is encoded in a *jump sequence* b_1^K, where $y_{k+b_k} = x_k$ and b_k is the displacement of phrase x_k. The jump sequence is constructed to ensure the phrases are permuted, thus avoiding deficiency problems similar to those suffered by Model 3 and Model 4.

We model the jump sequence by assuming that b_k depends on the phrase pair (x_k, u_k) and the jumps b_1^{k-1} that have occurred in the sentence so far

$$\mathrm{P}(b_1^K|x_1^K, u_1^K) = \prod_{k=1}^{K} \mathrm{P}(b_k|x_k, u_k, b_1^{k-1}) \tag{2.24}$$

In order to model this using an FST, we introduce equivalence classes for the sequence b_1^{k-1}: an equivalence class corresponds to the state of the FST. Write ϕ_{k-1} for the equivalence class containing b_1^{k-1}. The probability then becomes

$$\mathrm{P}(b_1^K|x_1^K, u_1^K) = \prod_{k=1}^{K} \mathrm{P}(b_k|x_k, u_k, \phi_{k-1}). \tag{2.25}$$

The probability of the (reordered) translation y_1^K is then

$$\mathrm{P}(y_1^K|x_1^K, u_1^K) = \begin{cases} \mathrm{P}(b_1^k|x_1^K, u_1^K), & y_{k+b_k} = x_k \\ 0, & \text{otherwise} \end{cases} . \tag{2.26}$$

These two operations are carried out by transducer R.

Maximum Jump 1 (MJ-1) is the simplest such model and allows a maximum jump of distance 1, i.e. $b_k \in \{-1, 0, +1\}$. A jump of -1 cannot occur unless a jump of $+1$ has just taken place, and a jump of $+1$ must be immediately followed by a jump of -1. Therefore there are two equivalence classes: one where a jump of $+1$ is possible with a given probability and one where a jump of -1 must occur next. Then

$$\mathrm{P}(b_k|x_k, u_k, \phi_{k-1}) = \begin{cases} \beta_1(x_k, u_k) & b_k = +1, \phi_{k-1} = 0 \\ 1 - \beta_1(x_k, u_k) & b_k = 0, \phi_{k-1} = 0 \\ 1 & b_k = -1, \phi_{k-1} = 1 \end{cases}$$

where the equivalence class is given the label $\sum_{l=1}^{k-1} b_l$. $\beta_1(x, u) = \mathrm{P}(B = +1|x, u)$ depends on the phrase pair and is estimated by finding the maximum likelihood estimates from the test data corpus. The MJ-2 model allows a phrase to jump a distance of up to 2 places; however, this introduces many more possibilities for the reordering of the sentence and is too complex to be used during decoding.

- **Target Phrase Insertion Model:** This WFST allows the spontaneous insertion of arbitrary target phrases at any point in the sentence. The reordered source phrase sequence $y_1, \ldots, y_K$ is transformed to a new sequence $c_0, c_1, \ldots, c_K$ that encodes information about inserted phrases.

  The terms $c_1, \ldots, c_K$ are the original phrases $y_1, \ldots, y_K$ with additional information that determines how many target language phrases can be inserted after that phrase, and the lengths of those phrases. An element c_k has the form

$$c_k = y_k \cdot p_k,$$

where $p_k \in \{1, \ldots, M\}^*$ is a sequence of integers between 1 and the maximum phrase length M. This restriction on phrase length is necessary as the phrases are drawn from the phrase pair inventory. Write $|p_k|$ for the length of p_k, and write $p_k = p_k[1] \cdot p_k[2] \cdots p_k[|p_k|]$: this specifies that phrase y_k has $|p_k|$ phrases inserted after it, of length $p_k[1], p_k[2], \ldots p_k[|p_k|]$ respectively. The probability of c_k for $k \in \{1, \ldots, K\}$ is defined as

$$\mathrm{P}(c_k|y_k) = \begin{cases} \alpha_0, & c_k = y_k \cdot \epsilon \\ \alpha^{\sum_{i=1}^{|p_k|} p_k[i]}, & c_k = y_k \cdot p_k \\ 0, & \text{otherwise} \end{cases} \tag{2.27}$$

α is known as the *Phrase Exclusion Probability* (PEP) and α^n is the probability of inserting a phrase of length n; this means the likelihood of inserting a phrase is inversely proportional to the number of words in the phrase. α can be tuned to improve translation performance. α_0 is the probability that no target language phrase is inserted. The values of α and α_0 must be set to ensure that $\mathrm{P}(c_k|y_k)$ forms a probability distribution.

The c_0 term allows the insertion of phrases at the beginning of the sentence and has the following probability distribution.

$$\mathrm{P}(c_0) = \begin{cases} \alpha_0, & c_0 = \epsilon \\ \alpha^{\sum_i p_0[i]}, & c_0 = p_0 \\ 0, & \text{otherwise} \end{cases} \tag{2.28}$$

The total probability of the sequence c_0 is given by

$$\mathrm{P}(c_0^K|u_1^K) = \mathrm{P}(c_0) \prod_{k=1}^{K} \mathrm{P}(c_k|u_k). \tag{2.29}$$

Given the sequence c_0^K, we now replace the markers with target language phrases to form d_0^K. Write $c_k = c_{k1}, c_{k2} \ldots$, and write $d_k = d_{k1}, d_{k2}, \ldots$. We assume that the inserted phrases are independent and depend only on the phrase length. The value of a phrase of length m is drawn uniformly from all phrases of length m in the PPI, i.e. the symbol m is mapped to v with probability

$$\mathrm{P}(v|m) = \frac{1}{\mathcal{P}_{\mathrm{tgt}}^{(m)}}, \tag{2.30}$$

where $\mathcal{P}_{\mathrm{tgt}}$ is the set of target phrases from the phrase pair inventory and $\mathcal{P}_{\mathrm{tgt}}^{(m)}$ is the subset of those target phrases that are of length m. Therefore

$$\begin{aligned} \mathrm{P}(d_0^K|c_0^K, y_1^K, x_1^K, u_1^K, K, e_1^I) &= \mathrm{P}(d_0^K|c_0^K) \\ &= \prod_{k=0}^{K} \mathrm{P}(d_k|c_k) \\ &= \prod_{l=1}^{|p_0|} \mathrm{P}(d_{0l}|c_{0l}) \prod_{k=1}^{K} \prod_{l=1}^{|p_k|+1} \mathrm{P}(d_{kl}|c_{kl}) \end{aligned} \tag{2.31}$$

We must ensure that the target phrase sequence v_1^R agrees with the phrase sequence produced by the model.

$$\mathrm{P}(v_1^R, d_0^K | c_0^K, y_1^K, x_1^K, u_1^K, K, e_1^I) = \mathrm{P}(d_0^K | c_0^K, a_1^K, u_1^K, K, e_1^I) 1\{d_0^K = v_1^R\} \tag{2.32}$$

Let Φ denote the target phrase insertion transducer, which performs all of this.

- **Target Phrase Segmentation Model:** We must now ensure that the phrase sequence v_1^R agrees with the words in the target sentence. Therefore we have

$$\mathrm{P}(f_1^J | v_1^R, d_0^K, c_0^K, a_1^K, u_1^K, K, e_1^I) = \mathrm{P}(v_1^R, d_0^K | c_0^K, a_1^K, u_1^K, K, e_1^I) 1\{f_1^J = v_1^R\} \tag{2.33}$$

For each sentence to be translated, a transducer Ω is created that maps the sequence of target phrases to the target sentence. A final acceptor T is used to ensure the output matches the target sentence.

Decoding uses a lattice of translation hypotheses $\mathcal{T}$ produced by the system as a composition (Mohri, 2009) of finite state transducers as follows, composing the FSTs right to left:

$$\mathcal{T} = G \circ (W \circ (R \circ (\Phi \circ (\Omega \circ T)))) \tag{2.34}$$

We then project onto the inputs of $\mathcal{T}$ to give a finite state acceptor that accepts any source sentence e_1^I that the translation process maps to the target sentence f_1^J. This takes the form of a lattice; each path through the lattice has an associated cost corresponding to a derivation $\{e_1^I, K, u_1^K, x_1^K, y_1^K, c_0^K, d_0^K, v_1^R\}$. The path of lowest cost through this lattice yields

$$\{\hat{e}_1^{\hat{I}}, \hat{K}, \hat{u}_1^{\hat{K}}, \hat{x}_1^{\hat{K}}, \hat{y}_1^{\hat{K}}, \hat{c}_0^{\hat{K}}, \hat{d}_0^{\hat{K}}, \hat{v}_1^{\hat{R}}\} = \underset{v_1^R, d_0^K, c_0^K, y_1^K, x_1^K, u_1^K, K, e_1^I}{\operatorname{argmax}} \mathrm{P}(v_1^R, d_0^K, c_0^K, y_1^K, x_1^K, u_1^K, e_1^I | f_1^J). \tag{2.35}$$

Taking the sequence of words e_1^I that corresponds to this path gives us the favoured translation hypothesis. The probabilities associated with the model components can be used as features in a log-linear model and training carried out using MERT (Och, 2003). MERT is applied to adjust the language model scale factor, word and phrase insertion penalties, phrase reordering scale factor, phrase insertion scale factor, source-to-target phrase translation model scale factor, target-to-source phrase translation model scale factor, and three phrase pair count features (Blackwood et al., 2009). The phrase pair count features track whether each phrase-pair occurred once, twice, or more than twice in the parallel text (Bender et al., 2007).

The experiments of Chapter 5 and Chapter 7 make use of the TTM as the underlying translation system.

2.6 Syntax-based translation

One drawback of phrase-based models is that the phrases are simply sequences of words rather than being syntactically motivated. Changes in word order usually occur for syntactic reasons, and the lack of syntactically motivated translation units makes it difficult to model changes in word order effectively. We may wish to use information about the syntax and structure of the sentence, particularly when translating between languages with different sentence structures and word order. For example, re-ordering is needed when translating

between languages with subject-verb-object sentence structure such as English and Chinese and languages with subject-object-verb structure such as Arabic and Japanese. We can make use of syntax information in the input language, output language or in both languages by constructing trees to model the sentence structure. We can view these as tree-to-string, string-to-tree and tree-to-tree models respectively.

Yamada and Knight (2001) introduce a string-to-tree syntax-based translation model that views the input language string as being generated by an output language parse tree passed through a noisy channel; the translation system can assign $\mathrm{P}(\mathbf{f}|\mathcal{E})$ for an English parse tree $\mathcal{E}$ and input sentence $\mathbf{f}$. An English string is assigned a parse tree by a syntactic parser, then the process by which translation occurs is modelled as a reordering of the parse tree, followed by insertion of words at each node and translation of leaf words. They report gains over IBM Model 5 for alignment of parallel text sentences. A decoder for this syntactic model, with the addition of phrasal translation, is presented by Yamada and Knight (2002).

Knight and Graehl (2005) develop tree transducers as a generalisation of the finite state transducer able to compute transformations of trees. Other work builds on this using increasingly complex rules extracted from parallel text to build models of string-to-tree alignment (Chiang et al., 2009; Galley et al., 2006, 2004). Joshua (Li et al., 2009) is an open source toolkit for parsing-based machine translation. Other syntax-based translation systems use tree-to-tree translation to make use of syntax in both languages (Alshawi et al., 2000; Melamed, 2004; Wu, 1997).

2.7 Hierarchical phrase-based translation and the HiFST decoder

Hierarchical phrase-based translation was introduced by Chiang (2005) and improved by Chiang (2007); the intention is to incorporate some of the advantages of both syntax-based translation and phrase-based translation. A *hierarchical phrase* is defined as a phrase that can also contain other phrases by using placeholders for those other phrases. The translation process is modelled using a synchronous context-free grammar (SCFG) to simultaneously build a tree in each language. Rules for the grammar are extracted from aligned parallel text data. The grammar in this case does not enforce syntactic rules (in the sense of traditional linguistic syntax) in either language and is referred to as *formally syntactic* rather than *linguistically syntactic*. Each rule in the grammar takes the form

$$X \rightarrow \langle \gamma, \alpha, \sim \rangle \tag{2.36}$$

with γ a string of terminals and non-terminals in the input language, α a string of non-terminals and terminals in the output language, and $\sim$ a one-to-one correspondence between non-terminals in γ and non-terminals in α. These non-terminals can then be expanded further during translation using other rules. As an example rule, the following rule encodes the fact that Chinese relative clauses modify the noun to their right and English relative clauses modify the noun to their left:

$$X \rightarrow \langle X_{\boxed{1}} \text{ de } X_{\boxed{2}}, \text{the } X_{\boxed{2}} \text{ that } X_{\boxed{1}} \rangle. \tag{2.37}$$

The correspondence between non-terminals is shown by boxed indices, e.g. $\boxed{1}$. Rules are learnt from a corpus of word-aligned parallel text. In addition to the rules learnt from the

parallel text, *glue rules* are defined, which allow a sentence to be split into a sequence of hierarchical phrases which are translated separately:

$$S \rightarrow \langle S_{\boxed{1}} X_{\boxed{2}}, S_{\boxed{1}} X_{\boxed{2}} \rangle \quad (2.38)$$

$$S \rightarrow \langle X_{\boxed{1}}, X_{\boxed{1}} \rangle \quad (2.39)$$

The start symbol S can then be expanded as a sequence of non-terminals which are translated without reordering. This allows increased robustness as it allows translation of sentences that cannot be derived from a single non-terminal X. The grammar is used to produce a derivation constraining the left hand side to be the input sentence. The expansion of a sentence halts when there are no further non-terminals: for each derivation, the right hand side is a hypothesis sentence e_1^I that the grammar has produced as a translation of f_1^J. We can relate hierarchical phrase-based translation to phrase-based translation as follows: restricting the grammar to conventional phrase pairs (i.e. rules with no non-terminals on the right hand side) and glue rules gives a phrase-based model with monotone phrase reordering.

The weight of each rule is determined by a number of features defined on rules such as:

- $\mathrm{P}(\alpha|\gamma)$ and $\mathrm{P}(\gamma|\alpha)$;
- lexical weights $\mathrm{P}_w(\alpha|\gamma)$ and $\mathrm{P}_w(\gamma|\alpha)$, calculated using alignment models to estimate how well the words of γ translate to the words of α and vice versa;
- a phrase penalty to allow the model to learn a preference for shorter or longer derivations.

The cost assigned to a hypothesis sentence is given by combination of the costs of the rules used to derive it, and by the application of a language model trained on large amounts of data in the output language. Cube pruning, a method of pruning that aims to make a minimal number of applications of the language model to the tree while maintaining translation quality, is introduced to control the size of the hypothesis lattices during translation.

The HiFST decoder (Iglesias et al., 2009a) is a lattice based decoder for hierarchical phrase-based translation, implemented using WFSTs. The OpenFst libraries (Allauzen et al., 2007) and standard FST operations such as composition, determinisation and minimisation are used to efficiently implement a decoder. They find that the use of WFSTs requires less pruning, resulting in fewer search errors and improved translation performance. The use of complete lattices rather than N-best lists improves parameter optimisation.

For investigation of the context-dependent alignment models described in Chapter 8, HiFST is used to perform translation.

2.8 Evaluation of translation quality

The quality of output from a translation system is generally judged relative to a reference translation (or reference translations) of the sentence in question. The criteria commonly used for evaluation of machine translation by human evaluators are *fluency* and *adequacy* (LDC, 2005; White et al., 1993). Fluency is a measure of the quality of language of the hypothesis translation, ranging from a grammatically flawless sentence to incomprehensible. Adequacy is a measure of how much of the meaning of the reference translation is expressed in the hypothesis translation.

Human evaluation of machine translation output is expensive in terms of both the time taken and the financial expense of paying bilingual and monolingual evaluators. There are also issues with inter-evaluator agreement (different evaluators may give different results) and intra-evaluator consistency (the same evaluator may produce different results at different times). Therefore it may not be possible to measure small changes in the quality of the output. When developing machine translation systems, we require feedback as rapidly as possible after a change has been made to the system, in order to determine quickly the success of the approaches applied. We therefore require an automated method of evaluating the output of a system.

For a corpus $\mathcal{C}$ of translation hypotheses, we use a corpus $\mathcal{R}$ of human-generated reference translations. Each sentence $\mathbf{e} \in \mathcal{C}$ may have one or more reference translations; write $R(\mathbf{e})$ for the reference sentences corresponding to $\mathbf{e}$. We now present a number of metrics used for evaluation of the hypotheses.

2.8.1 BLEU score

The BiLingual Evaluation Understudy (BLEU) score (Papineni et al., 2002) is a metric widely used for automatic evaluation of machine translation output. The basic premise is that a translation of a piece of text is better if it is close to a high-quality translation produced by a professional translator. The translation hypothesis is compared to the reference translation, or multiple reference translations, by counting how many of the n-grams in the hypothesis appear in the reference sentence(s); better translations will have a larger number of matches. The ranking of sentences using BLEU score has been found to closely approximate human judgement in assessing translation quality.

2.8.1.1 Computing modified n-gram precision for a sentence

Define $c(\mathbf{e}, w)$ to be the number of times a word w appears in sentence $\mathbf{e}$. For a candidate sentence $\mathbf{e}$, *unigram precision* is calculated by finding the number of words in the candidate sentence that occur in any reference transcription and dividing by the total number of words in the candidate sentence.

$$\text{unigram precision}(\mathbf{e}) = \frac{\sum_{w \in \mathbf{e}} c(\mathbf{e}, w) 1\{w \text{ occurs in some reference } \mathbf{e}_r\}}{\sum_{w \in \mathbf{e}} c(\mathbf{e}, w)}. \tag{2.40}$$

However, this is not an accurate measure of translation quality as the system can generate many words that occur in the references but not output grammatical or meaningful sentences. To overcome this, we clip the count of the number of times a word occurs so that it is not counted if it occurs more than the maximum number of times it occurs in any reference sentence. For words w in the candidate sentence $\mathbf{e}$, define

$$c_{\max}(w) = \max_{\mathbf{e}_r \in R(\mathbf{e})} c(\mathbf{e}_r, w) \tag{2.41}$$

$$c_{\text{clip}}(\mathbf{e}, w) = \min(c_{\max}(w), c(\mathbf{e}, w)) \tag{2.42}$$

The *modified unigram precision* for a sentence $\mathbf{e}$ is given by

$$p_1(\mathbf{e}) = \frac{\sum_{w \in \mathbf{e}} c_{\text{clip}}(\mathbf{e}, w)}{\sum_{w \in \mathbf{e}} c(\mathbf{e}, w)} \tag{2.43}$$

Similarly, for an n-gram $w_1^n = w_1, \ldots, w_n$, define $c(\mathbf{e}, w_1^n)$ to be the number of times the n-gram appears in a sentence and define

$$\begin{aligned} c_{\max}(w_1^n) &= \max_{\mathbf{e}_r \in R(\mathbf{e})} c(\mathbf{e}_r, w_1^n) \\ c_{\text{clip}}(\mathbf{e}, w_1^n) &= \min(c_{\max}(w_1^n), c(\mathbf{e}, w_1^n)) \end{aligned}$$

The *modified n-gram precision* is

$$p_n(\mathbf{e}) = \frac{\sum_{n\text{-grams } w_1^n \in \mathbf{e}} c_{\text{clip}}(\mathbf{e}, w_1^n)}{\sum_{n\text{-grams } w_1^n \in \mathbf{e}} c(\mathbf{e}, w_1^n)} \tag{2.44}$$

We wish to compute the BLEU score over the entire test corpus rather than at a per-sentence level. This is done by finding the n-gram counts and clipped n-gram counts sentence by sentence, adding the clipped n-gram counts and dividing by the total sum of the counts. The modified n-gram precision for the whole corpus $\mathcal{C}$ is given by

$$p_n(\mathcal{C}) = \frac{\sum_{\mathbf{e} \in \mathcal{C}} \sum_{n\text{-grams } w_1^n \in \mathbf{e}} c_{\text{clip}}(\mathbf{e}, w_1^n)}{\sum_{\mathbf{e} \in \mathcal{C}} \sum_{n\text{-grams } w_1^n \in \mathbf{e}} c(\mathbf{e}, w_1^n)} \tag{2.45}$$

2.8.1.2 Applying a brevity penalty to penalise short sentences

A candidate translation that is longer than the reference sentences will have a lower modified n-gram precision since the denominator of Equation (2.44) is larger. However, we also wish to penalise sentences that are shorter than the reference transcriptions. A *brevity penalty* is calculated over the entire corpus and penalises the model if the candidate translation sentences are too short compared to the reference translations. This brevity penalty is given by

$$b(c, r) = \begin{cases} 1 & \text{if } c > r \\ e^{1-\frac{r}{c}} & \text{if } c \leq r \end{cases}, \tag{2.46}$$

where $c = \sum_{\mathbf{e} \in \mathcal{C}} |\mathbf{e}|$ is the length of the candidate translation corpus and r is the *effective reference corpus length*. The effective corpus length can be calculated in a number of ways. The approach proposed by Papineni et al. (2002) is to take the reference sentence that is closest in length to the candidate sentence, the *best match length* and add these lengths to find the effective corpus length. However, the standard approach is to add the lengths of the shortest reference sentences, i.e.

$$r = \sum_{\mathbf{e} \in \mathcal{C}} \min_{\mathbf{e}_r \in R(\mathbf{e})} |\mathbf{e}_r|. \tag{2.47}$$

2.8.1.3 The BLEU score

The BLEU score of a corpus $\mathcal{C}$ is

$$\text{BLEU} = b(c, r) \exp\left(\sum_{n=1}^{N} \lambda_n \log p_n(\mathcal{C})\right) \tag{2.48}$$

where N is the maximum length of n-gram considered and λ_n are positive weights such that $\sum_{n=1}^{N} \lambda_n = 1$. The most commonly used values for these parameters are $N = 4, \lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \frac{1}{4}$.

This score ranges from 0 to 1, with a higher score indicating a better translation, but results are often quoted by scaling to a number between 0 and 100. Very few translations will achieve a score of 100, unless they are identical to one of the reference translations.

BLEU is widely used and has been found to correlate closely with human judgement of translation quality (Papineni et al., 2002), in that it assigns the same ranking to a set of translations as human judges. It can be used with one or multiple references; however, the score increases when more references are used (there are more n-grams in the references, so the numerator of Equation (2.44) increases while the denominator remains constant), and it is important to compare models using the same number of references on a particular data set.

2.8.2 Translation Edit Rate (TER)

Snover et al. (2005) introduce *Translation Edit Rate* (TER) as a more intuitive measure for evaluating machine translation quality. TER is defined as the minimum number of edits required to change a hypothesis so that it exactly matches the closest of the reference translations, normalised by the average length of the references.

$$\text{TER} = \frac{\#\text{ of edits}}{\text{average }\#\text{ of reference words}}, \tag{2.49}$$

where an edit can be insertion, deletion or substitution of a single word, or a shift of a word sequence. This is found to have a better correlation with human judgement than the BLEU score. *Human-targeted Translation Edit Rate* (HTER) is a modification that uses a human, who needs to be fluent only in the target language, to produce a fluent target reference that is as close as possible to the hypothesis and has the same meaning as the references (Snover et al., 2006). HTER has been found to be more consistent and fine-grained than human annotations of fluency and adequacy (Snover et al., 2009); it is used by the AGILE project for these reasons. However, the cost of HTER makes it impractical for frequent evaluation, and research has been carried out to find automated metrics to replace it (Przybocki et al., 2008).

2.8.3 Other evaluation criteria

Other methods of evaluating the quality of translation are available, though we use BLEU throughout.

Word Error Rate (WER) is found by computing the edit distance $d(\mathbf{e}, \mathbf{e}_r)$, i.e. the number of insertions, deletions and substitutions between the candidate sentence $\mathbf{e}$ and the reference sentence $\mathbf{e}_r$. **Multireference Word Error Rate** (mWER) is found by taking, for each candidate sentence $\mathbf{e}$, the edit distance of the closest reference sentence (Nießen et al., 2000).

A major disadvantage of WER is that it depends too much on the choice of reference sentence and requires the word order to be the same. **Position-independent WER** (PER) attempts to overcome this problem by comparing the words in the candidate and reference sentences, ignoring word order (Och and Ney, 2004).

Chan and Ng (2008) introduce **MaxSim**, an evaluation metric designed to replace BLEU that takes into account part-of-speech tags, grammatical fluency, lemmatised (i.e. grouped together so they can be analysed as a single item) word forms and synonyms, as well as n-gram similarity between a pair of sentences. Chiang et al. (2008a) discuss the limitations of

BLEU and propose a modification that improves correlation with human judgements. They find that deletion of entire sentences does not lead to a statistically significant drop in BLEU score relative to a baseline in which no words can be deleted during translation, and that models can defeat the intended purpose of the brevity penalty by deleting entire sentences to compensate for other sentences being too long. The solution is to clip the reference sentence length to encourage all sentences to be similar in length to their reference translations rather than making the corpus the same length overall; then the BLEU *with strict brevity penalty score* (BLEU-SBP) is identical to BLEU but uses corpus length

$$c = \sum_{\mathbf{e} \in \mathcal{C}} \min \left\{ |\mathbf{e}|, \min_{\mathbf{e}_r \in R(\mathbf{e})} |\mathbf{e}_r| \right\}. \tag{2.50}$$

Chiang et al. (2008a) also propose a cross between BLEU and word error rate. Many further metrics for automatic evaluation of machine translation output are studied by the NIST MetricsMATR08 Challenge (Przybocki et al., 2008).

2.9 Pre-processing of data

In this thesis, experiments are primarily carried out for Arabic to English and Chinese to English translation, though some supplementary experiments are run for French to English. Data are obtained from a variety of sources and are pre-processed to improve translation performance. The most significant changes occur in Arabic, due to the complexity of its morphology.

2.9.1 Tokenisation and morphological analysis of Arabic text

Arabic is a morphologically complex language, with the morphological analysis of each word depending on a large number of features such as part-of-speech, voice, gender, person and number. Prefixes and suffixes are added to words to indicate each of these features, which results in an increased vocabulary size compared to English for a given collection of parallel text, and fewer examples of words in the vocabulary. For example, 2,200 distinct morphological analyses appear in the first 280,000 words of the Penn Arabic Treebank, whereas English morphological tagsets generally have about 50 tags (Habash and Rambow, 2005).

Determination of the morphological analysis of a word is difficult for two main reasons. Firstly, Arabic words are often ambiguous in their analysis due to the omission of disambiguating short vowels and other orthographic diacritics; a word form in the Penn Arabic treebank has an average of 2 morphological analyses. The second reason is that, due to the number of possible morphological analyses for a word, computational methods of morphological analysis developed for English suffer from data sparsity. Habash and Rambow (2005) introduce the Morphological Analysis and Disambiguation for Arabic (MADA) tool, which performs simultaneous tokenisation, part-of-speech tagging and morphological analysis of Arabic. MADA selects from analyses of Arabic sentences produced by the Buckwalter Arabic Morphological Analyzer (Buckwalter, 2002) using a combination of classifiers including those for part-of-speech, number and gender.

MADA was used to pre-process the Arabic data for all experiments described in the following chapters. The 'D2' decliticisation scheme described by Habash and Sadat (2006) was used. This causes a reduction in vocabulary size of around 30% from the original Arabic

[1]. This processing is carried out on the parallel text used for model training and on the Arabic test data prior to its translation into English. This kind of processing has also been shown to produce a significant increase in BLEU score if there is a change in genre between the training data and test data (Habash and Sadat, 2006).

2.9.2 Processing of Chinese text

Word-segmented Chinese text was provided by the DARPA AGILE project. Chinese text segmentation is a well established technology (Ng and Low, 2004; Shi and Wang, 2007; Zhang and Clark, 2008). The segmenters used within the AGILE project were trained so as to be consistent with the Penn Chinese Treebank (Xue et al., 2005).

2.10 Post-processing of translation hypotheses

Once a lattice or N-best list of translation hypotheses has been built and scored by the model during decoding, additional algorithms can be run to further improve the quality of translation. There are two broad approaches to post-processing. Some algorithms can be used to rescore the output of one decoder, while others combine the outputs of two or more decoders. They do not depend on the choice of decoder so they can be run on the output of both phrase-based and syntax-based systems.

Minimum Bayes Risk (MBR) decoding is a rescoring method that tries to reduce the probability of choosing the wrong hypothesis by selecting the translation that is closest on average to the likely translations from an N-best list (Kumar and Byrne, 2004). The 'closeness' is measured by a loss function that we define, usually a quantity related to the translation quality metric we use for evaluation. Ehling et al. (2007) find MBR decoding of N-best lists gives gains in BLEU score for a variety of languages. Lattices generally contain many more hypotheses and lattice-based MBR is known to improve on N-best MBR for speech recognition (Goel and Byrne, 2000); Tromble et al. (2008) develop lattice-based MBR for machine translation output. They find a linear approximation to the BLEU score using a first order Taylor expansion, which can be computed over the hypotheses of the entire lattice and obtain statistically significant gains in BLEU score relative to N-best MBR.

2.10.1 System combination

System combination is another area in which machine translation has drawn inspiration from speech recognition. Most system combination approaches are similar to ROVER (Fiscus, 1997), used in speech recognition for allowing systems to 'vote' on a confusion network of hypotheses. The outputs from multiple systems are combined to give a single hypothesis, in the hope that the combined information from a number of systems will give a more accurate translation.

The approach works by selecting a *skeleton* or *primary hypothesis* and aligning other hypotheses with it. We can then build a confusion network, which encodes all hypotheses

[1]Words that MADA cannot process correctly are passed directly to the output: our objective in using MADA is to reduce the Arabic vocabulary whenever possible so that the statistical translation models do not suffer from data sparsity. We do not seek any linguistic or phonetic processing of the Arabic word, and it is sufficient that words are processed consistently across the data sets.

from an N-best list and the differences between them; the component systems are then used to select the best path through the confusion network, which is known as the *consensus hypothesis*. The consensus hypothesis may be different to any of the input hypotheses; we can view this as using the models to select the best parts from each hypothesis and combining them to produce a sentence that is a more accurate translation than any of the hypotheses individually.

Bangalore et al. (2001) introduce this approach and use the edit distance metric (which is the number of insertions, deletions and substitutions taken to get from one sequence of words to another) to align sentences. Matusov et al. (2006) introduce models for word-to-word alignment based on IBM Model 1 and Hidden Markov Model alignment (see Chapter 3 for more details of these models). These models are estimated on all sentences of the test corpus, and hypothesis sentences are re-ordered to match the ordering of the primary hypothesis prior to their inclusion in the confusion network.

Phrases are a more natural translation unit than words and reordering at the word level can introduce disfluencies in the output. Sim et al. (2007) overcome this problem by using consensus network decoding to find a consensus hypothesis $\mathbf{e}_{\text{con}}$, then applying MBR decoding to find the original hypothesis that has minimal loss with respect to $\mathbf{e}_{\text{con}}$. Feng et al. (2009) introduce a method of system combination that uses lattices rather than confusion networks and deals with phrase alignments in addition to word alignments.

De Gispert et al. (2009) combine N-best lists from two machine translation systems trained on alternative morphological decompositions of the input language. The posterior distributions over the individual lists are interpolated to form a distribution over the entire list, and MBR rescoring of the new list is carried out; this produces a higher quality of translation than either system individually.

The approach of de Gispert et al. (2009) can be extended using methods presented by Tromble et al. (2008) so that it can perform MBR system combination with complete lattices (de Gispert et al., submitted to Computational Linguistics). We begin with M lattices $\mathcal{E}_1, \ldots, \mathcal{E}_M$ and wish to find the best hypothesis sentence from their union $\mathcal{E} = \mathcal{E}_1 \cup \ldots \cup \mathcal{E}_M$. For each n-gram w occurring in a lattice, the posterior probability of w over that lattice is calculated; the posterior probabilities for each n-gram are then linearly interpolated (using weights determined on a tuning set) to give posterior probability distribution $p(w|\mathcal{E})$. These interpolated probabilities are used to compute a linear approximation to BLEU (Tromble et al., 2008), and the MBR decision rule can then be written

$$\hat{\mathbf{e}} = \underset{\mathbf{e} \in \mathcal{E}}{\operatorname{argmax}} \left\{ \theta_0 |\mathbf{e}| + \sum_{w \in \mathcal{N}} \theta_w \#_w(\mathbf{e}) p(w|\mathcal{E}) \right\} \tag{2.51}$$

where $\mathcal{N}$ is the set of all n-grams occurring in the lattices, $\#_w(\mathbf{e})$ is the number of times w occurs in the sentence $\mathbf{e}$ and θ_w is an n-gram specific weight. This can be implemented efficiently using WFST operations, and is used in Chapter 9 for system combination.

The choice of systems that are combined is also important. Macherey and Och (2007) discuss desirable properties for the systems being combined. The systems should be of similar quality, to avoid systems of inferior quality reducing the overall effectiveness. They should also be dissimilar, so that each system provides different information about the overall translation: we can view this as one system correcting the mistakes made by others. Equally, if all systems 'agree' that a translation scores highly, then it is likely to be a good translation.

2.11 Summary

This chapter has introduced the framework used for statistical machine translation and examined some of the models used. The steps in the process have been described, including sources of data and its pre-processing, building of alignment models using parallel text, log-linear models of translation, features for log-linear models, discriminative training of models, the decoding process itself and subsequent post-processing and system combination. The translation systems we use for our work were introduced, as well as automated metrics for evaluation of translation quality.

CHAPTER 3
Alignment Models

3.1 Introduction

The fundamental motivation of alignment modelling for machine translation is that, given a source sentence $\mathbf{e} = e_1^I$ and target sentence $\mathbf{f} = f_1^J$, we wish to define a correspondence between the words of $\mathbf{e}$ and $\mathbf{f}$. A link between two words can be viewed as an indication they are translations of each other, and each link can be represented as a pair (i, j) with $1 \leq i \leq I, 1 \leq j \leq J$. We write A for the set of such links; each element of A can be viewed as an undirected arc in a graph with $I + J$ nodes, one node for each word of each sentence. We can also view the alignment links as an $I \times J$ matrix B, where

$$b_{ij} = \begin{cases} 1 & \text{if } e_i \text{ is linked to } f_j \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Figure 3.1 shows the graphical representation of the alignment between the English sentence *and the program has been implemented* with an equivalent French sentence *le programme a été mis en application*, and Figure 3.2 shows the same alignment represented as a matrix. We develop models to estimate the probability of alignments between the sentences, i.e. we wish to know

$$\mathrm{P}(A|\mathbf{e}, \mathbf{f}). \tag{3.2}$$

We should note that it is difficult to judge which words in a given target sentence correspond to which words in a source sentence. There is significant disagreement even among human translators as to the correspondence between words (Melamed, 1998), so the solution is subjective and there is no "best" solution: our main objective is to produce alignment models that lead to the best translations. However, the quality of alignment is crucial to the

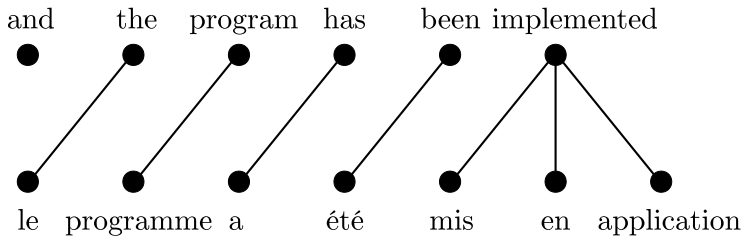


Figure 3.1: Representation of alignment as a graph.

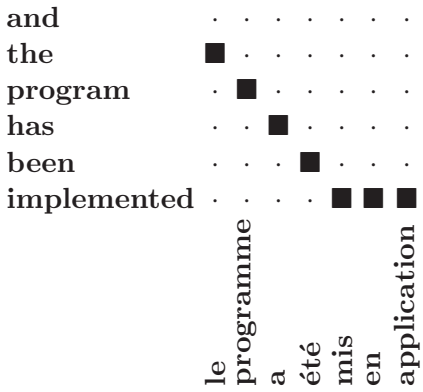


Figure 3.2: Representation of alignment as a matrix.

quality of the phrases extracted from that alignment and therefore the overall quality of the translation system. For this reason, we find it useful to investigate the quality of alignments produced by our models and compare them to manually obtained alignments.

A common approach to producing alignments, and the one that is taken in this thesis, is to use parallel text to estimate two alignment models: one aligning $\mathbf{f}$ to $\mathbf{e}$ and another aligning $\mathbf{e}$ to $\mathbf{f}$. The alignments in the two directions are then combined to produce a single alignment for each sentence in the parallel text. The translation model is then initialised from this word-aligned parallel text. The process is represented graphically in Figure 3.3.

3.2 Many-to-one alignment models

Unconstrained alignment allows each source word to be aligned to any number of target words and each target word to be aligned to any number of source words. This is complicated to model, so we constrain the models to reduce computational complexity. We allow each target word f_j to be aligned to just one source word e_i; we can then represent the alignment between the sentences as a sequence $\mathbf{a} = a_1^J$ and define $a_j = i$ to be the position of the word in the source sentence to which f_j is aligned. This is many-to-one, in that many target words can be aligned to each source word, but each target word is aligned to at most one source word. We can view the the alignment sequence a_1^J as a hidden variable that is not specified in the parallel text, but which must be determined in order to build accurate models

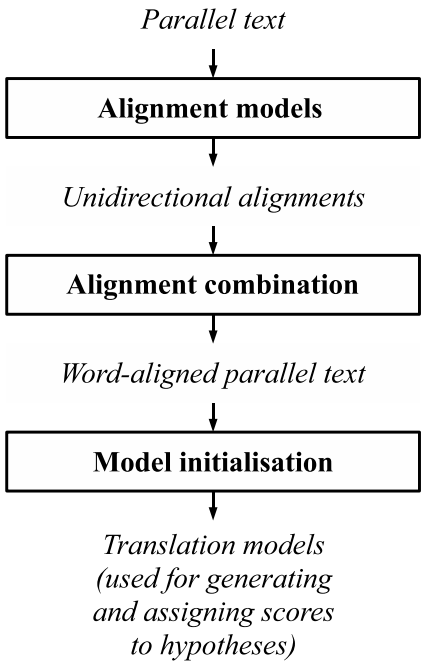


Figure 3.3: Use of alignment models to initialise a machine translation system.

of the translation process. We view this as a generative model, and model the probability of generating f_1^J from e_1^I with alignment a_1^J by

$$\mathrm{P}(a_1^J, f_1^J | e_1^I). \tag{3.3}$$

Sometimes a word in the target sentence is not a direct translation of any word in the source sentence (for example, there is no word in the French sentence corresponding to the English word *and*). We wish to allow for this in the models by introducing a null token e_0 or `NULL`, and set $a_j = 0$ if f_j does not align to any target word e_i with $1 \leq i \leq I$. Alignments to the null token must be considered when training the models.

The alignment that maximises the joint probability of the target sentence and alignment given the source sentence is the known as the *Viterbi alignment* and is calculated by finding:

$$\hat{a}_1^J = \underset{a_1^J}{\operatorname{argmax}} \mathrm{P}(f_1^J, a_1^J | e_1^I) \tag{3.4}$$

The *Viterbi approximation* $\mathrm{P}(f_1^J | e_1^I) \approx \mathrm{P}(f_1^J, \hat{a}_1^J | e_1^I)$ is sometimes used to avoid computation of the full marginal probability during training, particularly if the models are complex.

Given a source sentence $e_1, \ldots, e_I = e_1^I$, we can express the joint probability of a target

sentence $f_1, \ldots f_J = f_1^J$ and alignment $a_1, \ldots, a_J = a_1^J$ as follows:

$$\begin{aligned} \mathrm{P}(f_1^J, a_1^J|e_1^I) &= \mathrm{P}(J|e_1^I)\mathrm{P}(f_1^J, a_1^J|e_1^I, J) \\ &= \mathrm{P}(J|e_1^I)\prod_{j=1}^{J}\mathrm{P}(f_j, a_j|f_1^{j-1}, a_1^{j-1}, e_1^I, J) \\ &= \mathrm{P}(J|e_1^I)\prod_{j=1}^{J}\mathrm{P}(f_j|a_1^j, f_1^{j-1}, J, e_1^I)\mathrm{P}(a_j|f_1^{j-1}, a_1^{j-1}, J, e_1^I) \end{aligned}$$

Due to the complexity of the models that would be required and insufficient training data to estimate the parameters accurately, we cannot estimate these probability distributions reliably. Therefore a number of simplifying assumptions are made, which are described in this chapter.

3.3 IBM alignment models

Brown et al. (1993) propose five statistical models of the translation process, numbered from 1 to 5 in increasing order of complexity and corresponding number of parameters. Training is carried out using a corpus of sentence-aligned parallel text. The models are trained in order, using Model 1 to provide an initial estimate of the parameters of Model 2, using Model 2 to provide an initial estimate of the parameters of Model 3, and so on. We begin by describing the simplest such model.

3.3.1 Model 1

Model 1 makes the conditional independence assumptions that:

- The target sentence length J is independent of e_1^I and I, and we assume it takes some small value $\mathrm{P}(J|e_1^I) = \epsilon$; we use the models to align sentences whose length is known prior to alignment, so this does not cause problems.
- For each target word, all alignments (including alignment to `NULL`) are equally likely and do not depend on the particular word or its position on the sentence:

$$\mathrm{P}(a_j|f_1^{j-1}, a_1^{j-1}, J, e_1^I) = \frac{1}{I+1} \text{ for } 0 \leq j \leq J; \tag{3.5}$$

- Once the alignments have been determined, the target word depends only on the source word to which it is aligned, i.e.

$$\mathrm{P}(f_j|a_1^j, f_1^{j-1}, J, e_1^I) = t(f_j|e_{a_j}), \tag{3.6}$$

  where $t(f_j|e_{a_j})$ is the *word-to-word translation probability* of f_j given e_{a_j}.

Then the probability of generating f_1^J from e_1^I with alignment a_1^J is

$$\mathrm{P}(f_1^J, a_1^J|e_1^I) = \frac{\epsilon}{(I+1)^J}\prod_{j=1}^{J}t(f_j|e_{a_j}). \tag{3.7}$$

We take the sum over all possible alignments to give the probability of generating f_1^J from e_1^I.

$$\begin{aligned} \mathrm{P}(f_1^J|e_1^I) &= \sum_{a_1^J} \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j|e_{a_j}) \\ &= \frac{\epsilon}{(I+1)^J} \sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} t(f_j|e_{a_j}) \end{aligned} \tag{3.8}$$

3.3.1.1 Enhancements to Model 1

Moore (2004) discusses the limitations of Model 1 and proposes several changes to improve its performance. The 'garbage collection' problem occurs because rare source words tend to align to too many words in the target language. For example, if a source word occurs only once in the parallel text, the probability assigned to it generating each of the words in the sentence to which it is paired will be too high. This problem is solved by smoothing the word-to-word translation probabilities with a uniform distribution.

Further gains are found by initialising the translation probabilities using a heuristic parameter estimate based on the log likelihood ratio rather than using uniform translation probabilities as is traditional. This causes fewer iterations of EM to be required until convergence and appears to prevent over-fitting to training data.

Model 1 also suffers from structural problems which cannot be corrected without making the model more complex. It is many-to-one, which fails to model the fact that the actual alignments can be many-to-many or one-to-many (a phrase in the source language translates as a single word in the target language). It suffers from distortion, in that the position of a target word is independent of the position of the source word to which it is aligned, whereas real alignments tend to consist of contiguous sequences of words in the target language being translated as a contiguous sequence in the source language, so that the alignment matrix is close to diagonal. It also does not model fertility, i.e. the tendency that some source words tend to generate more words than others, so the probability that a given source word generates a target word is unaffected by how many other words that source word generates.

3.3.2 Model 2

For Model 2, the assumptions are less restrictive. We assume that the alignment position a_j of word f_j depends on j as well as on the lengths of the sentences J, I, i.e.

$$\mathrm{P}(a_j|f_1^{j-1}, a_1^{j-1}, J, e_1^I) = a(a_j|j, J, I), \tag{3.9}$$

for $1 \leq j \leq J$ and $0 \leq a_j \leq I$ where $\sum_{i=0}^{I} a(i|j, J, I) = 1$. This has the effect of encoding that alignments are often monotonic and alignment links occur close to the diagonal of the alignment matrix. Then

$$\mathrm{P}(f_1^J, a_1^J|e_1^I) = \epsilon \prod_{j=1}^{J} a(a_j|j, J, I) t(f_j|e_{a_j}); \tag{3.10}$$

$$\mathrm{P}(f_1^J|e_1^I) = \epsilon \sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} a(a_j|j, J, I) t(f_j|e_{a_j}). \tag{3.11}$$

3.3.3 Fertility-based models

Model 1 and Model 2 operate by selecting, for each target word, a source word to which it aligns. The remainder of the models are different. In manual alignments, we see that different source words have a tendency to be aligned with different numbers of target words. We define a random variable Φ_e to be the number of words to which e is connected in an alignment; we view this as the number of words generated by e and describe Φ_e as the *fertility* of e.

The generative model used is described as follows. We first decide the fertility ϕ_i of each source word e_i, and then decide the target words $\tau_i = \tau_{i,1}, \ldots, \tau_{i,\phi_i}$ it generates. This sequence of target words (which can be empty if the fertility is zero) is known as a *tablet*, and the collection of tablets is known as a *tableau*. After that, we decide the positions of each word in the tableau in the target sentence; write $\pi_{i,k}$ for the position of $\tau_{i,k}$. We express the joint distribution of the fertility, translation and reordering given the source sentence as

$$\mathrm{P}(\pi_0^I, \tau_0^I | e_1^I) = \mathrm{P}(\phi_0^I | e_1^I) \mathrm{P}(\tau_0^I | \phi_0^I, e_1^I) \mathrm{P}(\pi_0^I | \tau_0^I, e_1^I). \tag{3.12}$$

Once the tableau has been decided, its reordering does not depend on the word fertilities. The component probabilities are expressed as follows:

- Decide on the fertility for each word in turn

$$\mathrm{P}(\phi_0^I | e_1^I) = \prod_{i=1}^{I} \mathrm{P}(\phi_i | \phi_1^{i-1}, e_1^I) \times \mathrm{P}(\phi_0 | \phi_1^I, e_1^I) \tag{3.13}$$

- Decide on the target words

$$\mathrm{P}(\tau_0^I | \phi_0^I, e_1^I) = \prod_{i=0}^{I} \prod_{k=1}^{\phi_i} \mathrm{P}(\tau_{i,k} | \tau_{i,1}^{i,k-1}, \tau_0^{i-1}, \phi_0^I, e_1^I) \tag{3.14}$$

- Decide the positions of the words generated, positioning the words aligned to `NULL` last

$$\begin{aligned} \mathrm{P}(\pi_0^I | \tau_0^I, \phi_0^I, e_1^I) = & \prod_{i=1}^{I} \prod_{k=1}^{\phi_i} \mathrm{P}(\pi_{i,k} | \pi_{i,1}^{i,k-1}, \pi_1^{i-1}, \tau_0^I, \phi_0^I, e_1^I) \\ & \times \prod_{k=1}^{\phi_0} \mathrm{P}(\pi_{0,k} | \pi_{0,1}^{0,k-1}, \pi_1^I, \tau_0^I, \phi_0^I, e_1^I) \end{aligned} \tag{3.15}$$

The words of the tableau τ and reordering π determine a target string $\mathbf{f}$ and alignment $\mathbf{a}$, but several different pairs τ, π may lead to the same $\mathbf{f}, \mathbf{a}$. Define $\langle \mathbf{f}, \mathbf{a} \rangle$ to be the set of all such pairs. We take the sum over these pairs to find the probability:

$$\mathrm{P}(\mathbf{f}, \mathbf{a} | \mathbf{e}) = \sum_{(\tau,\pi) \in \langle \mathbf{f}, \mathbf{a} \rangle} \mathrm{P}(\tau, \pi | \mathbf{e}). \tag{3.16}$$

IBM Models 3, 4 and 5, described in the following sections, are examples of fertility-based models.

3.3.4 Model 3

Model 3 is a fertility-based model, with the following modelling assumptions made to simplify the probability distributions described above:

- The *fertility probability* $\mathrm{P}(\phi_i|\phi_1^{i-1}, e_1^I) = n(\phi_i|e_i)$ depends only on e_i, for $1 \leq i \leq I$. The fertility of a given source word depends only on that word and not on its neighbours or their fertilities.
- $\mathrm{P}(\tau_{i,k}|\tau_{i,1}^{i,k-1}, \tau_0^{i-1}, \phi_0^I, e_1^I) = t(\tau_{i,k}|e_i)$. The translation depends only on the source and target word pair, and not on any previous source or target words.
- $\mathrm{P}(\pi_{i,k} = j|\pi_{i,1}^{i,k-1}, \pi_1^{i-1}, \tau_0^I, \phi_0^I, e_1^I) = d(j|i, J, I)$. The reordering depends only on the position of the target word, the position of the source word and the lengths of the two sentences. We refer to $d(j|i, J, I)$ as the *distortion probability*.

3.3.5 Model 4

In manually-aligned translations, a sequence of words (or phrase) in the source language is translated to a phrase in the target language, and these phrases are then reordered as units. Since the distortion probabilities of Model 3 move each word in a tablet independently, the model does not account for this. Model 4 attempts to overcome this limitation by using two types of distortion model: one is applied to position the first word in a tablet; the other positions the remaining words in the tablet relative to the first word.

The probability associated with positioning the first word $\tau_{i,1}$ in a tablet τ_i is defined to be

$$\mathrm{P}(\pi_{i,1} = j|\pi_1^{i-1}, \tau_0^I, \phi_0^I, e_1^I) = d_1(j - c_{[i-1]}|\mathcal{A}(e_{[i-1]}), \mathcal{B}(f_j)), \tag{3.17}$$

where $c_i = \left\lceil \frac{1}{\phi_i} \sum_{k=1}^{\phi_i} \pi_{i,k} \right\rceil$ is the ceiling of the average of the positions in the target sentence of the words in the tablet τ_i. We call c_i the *center* of τ_i, as it intuitively the "middle" of the tablet's positioning in the target sentence. $\mathcal{A}, \mathcal{B}$ are functions over the source and target vocabularies respectively that take a small number of values, and effectively group words into classes depending on the effect they have on the ordering in the translation.

To place subsequent words in the tablet, the following probability is used:

$$\mathrm{P}(\pi_{i,k} = j|\pi_{i,1}^{i,k-1}, \pi_1^{i-1}, \tau_0^I, \phi_0^I, e_1^I) = d_{>1}(j - \pi_{i,k-1}|\mathcal{B}(f_j)). \tag{3.18}$$

This enforces the constraint that $\pi_{i,k} > \pi_{i,k-1}$, so the words are placed left to right, but does not force the words to take consecutive positions in the target sentence.

3.3.6 Model 5

Models 3 and 4 are *deficient*, which means that non-zero probability is assigned to invalid target sentences, such as those where several target words lie on top of each other in one position and where other positions have no word, i.e. $\pi_{i,k} = \pi_{i',k'}$ for some $(i, k) \neq (i', k')$. While removing the dependence of the distortion probability on previously positioned words causes the model to be much simpler, it is the cause of deficiency. In addition, Model 4 allows the positioning of words before the first position or after the last position in the target sentence.

Model 4 produces some of the best quality alignment results; however the fact that it is deficient means it is not possible to use efficient dynamic programming algorithms on it. Alignment models are hard to include in the decoding process since in the worst case, the number of possible alignments between sentences increases exponentially with the length of the sentence. (Knight, 1999) shows that, even for simple statistical models, the decoding process is NP-complete.

Model 5 removes deficiency at the expense of making the model complicated to the extent that it is rarely used. For this reason, it will not be discussed further.

3.4 Training of IBM alignment models

The alignment models are trained on a large corpus of parallel text data, comprising sentence pairs $\{(\mathbf{e}^{(s)}, \mathbf{f}^{(s)})\}_{s=1}^{S}$, where $\mathbf{e}^{(s)}$ is a sentence in the source language, $\mathbf{f}^{(s)}$ is a sentence in the target language and the two sentences have the same meaning. We aim to find an alignment $\mathbf{a}^{(s)}$ for each sentence pair.

We train using the iterative Expectation Maximisation algorithm (Dempster et al., 1977; Neal and Hinton, 1999), which performs maximum likelihood estimation for data in which some variables are unobserved (in this case, the alignment variables are unobserved). We wish to update the model parameters θ to maximise the log likelihood of the parallel text under the model, i.e. we maximise

$$L_\theta(\mathbf{e}^{(1)}, \mathbf{f}^{(1)}, \ldots, \mathbf{e}^{(S)}, \mathbf{f}^{(S)}) = \log \prod_{s=1}^{S} \mathrm{P}_\theta(\mathbf{f}^{(s)}|\mathbf{e}^{(s)}). \tag{3.19}$$

The algorithm consists of two steps at each iteration: during the E-step, we find the value of the *auxiliary function*, i.e. the expected value of the log likelihood with respect to the conditional distribution of the alignment given the sentences under the current estimate of the model parameters θ_t:

$$Q(\theta|\theta_t) = \sum_{s=1}^{S} \sum \mathbf{a}^{(s)} \mathrm{P}_{\theta_t}(a^{(s)}|\mathbf{e}^{(s)}, \mathbf{f}^{(s)}) \log \mathrm{P}_\theta(\mathbf{f}^{(s)}, \mathbf{a}^{(s)}|\mathbf{e}^{(s)}) \tag{3.20}$$

During the M-step, we update the model parameters θ to those which maximise the auxiliary function:

$$\theta_{t+1} = \operatorname*{argmax}_\theta Q(\theta|\theta_t) \tag{3.21}$$

Model 1 and Model 2 can be trained by exact EM. However, the structure of Models 3 and 4 means means it is impossible to compute sums over all possible alignments (which is required for computation of the auxiliary function) so an approximation is used: the sum is instead taken over the Viterbi alignment and those alignments within a neighbourhood of the Viterbi alignment (see Brown et al. (1993) for further details).

3.5 Hidden Markov Model alignment

Vogel et al. (1996) propose a Hidden Markov Model (HMM) model for word-to-word alignment of sentences of parallel text, where the words of the source sentence are viewed as states of

the HMM and emit target words according to a probability distribution. The sentence-level alignment model is effectively split into two components: the *alignment component* models the transitions between the states of the HMM and the *translation component* controls the emission of target words from those states. The main benefit of such a model is that, unlike IBM Model 4, it can be trained efficiently using the exact EM algorithm, and full Baum-Welch re-estimation of probability distributions can take place (although Vogel et al. (1996) use only Viterbi training).

In alignments produced by a human translator, the alignment of one word depends strongly on the alignment of the previous word and we require a model that captures this dependency. We assume that each alignment a_j depends only on the previous alignment (the Markov assumption) and define a hidden Markov model by associating each state i with word e_i for $1 \leq i \leq I$. We also assume that f_j depends only on the word e_{a_j} to which it is aligned and that f_j is emitted from a state i with probability $t(f_j|e_i)$. The transitions between the states of the HMM are governed by a probability distribution $a(i|i', I)$, with $a(i|i', I)$ the probability of emitting a word from state i given the last word was emitted from state i' (for $1 \leq i, i' \leq I$). Then the joint probability of target sentence f_1^J and alignment a_1^J given e_1^I is

$$\mathrm{P}(f_1^J, a_1^J|e_1^I) = \mathrm{P}(J|e_1^I) \prod_{j=1}^{J} [a(a_j|a_{j-1}, I)t(f_j|e_{a_j})] \tag{3.22}$$

The model includes a probability $\mathrm{P}(J|e_1^I)$ for the length of the target sentence, which is not needed during alignment (since we know the lengths of the sentences and our aim is to estimate the other parameters better).

As presented by Vogel et al. (1996), one flaw in this model is that it does not account for the fact that there may be target words that are not connected to any word in the source sentence. We wish to consider this case and, as with the IBM models; Och and Ney (2000a) introduce the null word e_0 to the HMM alignment model to account for unaligned target words. This is positioned arbitrarily at the start of the sentence: therefore models which make use of distance between source words will not work without modification. The null token is included by introducing another I states into the HMM so that each state i representing emission from source word e_i has a corresponding state $i + I$ which allows for the possibility of a transition to the null word but retains information about the last non-null emitting word.

3.5.1 Estimation of HMM transition probabilities

One of the benefits of decomposing the model into translation probabilities $t(f|e)$ and transition probabilities $a(i|i', I)$, giving the probability of emitting from source word i given the previous target word was generated from $e_{i'}$, is the ability to modify the components separately. We estimate transition probabilities $\bar{a}(i|i', I)$ from parallel text training data using EM, but modifications such as interpolation with another model or smoothing are often used to form the distribution $a(i|i', I)$ actually used in translation.

The alignment parameters can be estimated in a number of ways. The HMM transition probabilities may depend only on the jump width (Och and Ney, 2000a; Vogel et al., 1996), estimated using the formula

$$\bar{a}(i|i', I) = \frac{c(i - i')}{\sum_{i''=1}^{I} c(i'' - i')}, \tag{3.23}$$

where $c(d)$ is the count of jumps of distance d in the training data. We then set the probability of a transition to e_0 as p_0 and the transition probabilities are

$$a(i+I|i',I) = \begin{cases} p_0, & \text{if } i = i' \\ 0, & \text{if } i \neq i' \end{cases} = p_0\delta(i,i') \tag{3.24}$$

i.e. a transition from a state i to the corresponding state $i+I$ occurs with probability p_0, but it is not possible to enter the the state $i'+I$ from state i for $i' \neq i$. Similarly, if the HMM has just emitted a word from `NULL`, it can emit another word from `NULL` but cannot emit from a state $i'+I$ corresponding to a different word:

$$a(i+I|i'+I,I) = p_0\delta(i,i'), \tag{3.25}$$

The transition probabilities to the other states are modified to take account of the fact that probability mass has been assigned to transition to a null state:

$$a(i|i',I) = (1-p_0)\bar{a}(i|i',I) \tag{3.26}$$
$$a(i|i'+I,I) = (1-p_0)\bar{a}(i|i',I). \tag{3.27}$$

For better estimation of infrequent transitions, we can smooth by interpolation with a uniform distribution (Och and Ney, 2000a):

$$a(i|i',I) = \lambda \times \frac{1}{I} + (1-\lambda) \times \bar{a}(i|i',I) \tag{3.28}$$

The transition probabilities can also depend on the position in the sentence rather than just the size of the jump (Deng and Byrne, 2005a). The alignment probabilities are therefore estimated using

$$a(i|i',I) = \frac{c(i,i',I)}{\sum_{i''=1}^{I} c(i'',i',I)}. \tag{3.29}$$

This is then interpolated with uniform and jump width distributions to reduce data sparsity.

3.6 Word-to-phrase HMM alignment model

The Markov assumption allows efficient dynamic programming algorithms for training and alignment. However the fertility and distortion models in Model 4 allow it to produce better quality word alignments at the expense of having a complicated and computationally expensive training procedure. Word-to-phrase alignment is considered by Och et al. (1999) but Deng and Byrne (2005a) extends the HMM model to include word-to-phrase alignment. The aim of the word-to-phrase HMM is to combine a training algorithm as efficient as that of the HMM word-to-word alignment model with the capability of producing improved word alignments to rival Model 4.

Define a *phrase* as a single word or as a sequence of consecutive words within a sentence. Consider the the target sentence f_1^J as a sequence of K phrases v_1^K, and assume each phrase v_k is generated as a translation of a single source word e_{a_k} determined by the alignment sequence a_1^K where $a_k \in \{1,\ldots,I\}$. Let ϕ_k be the length of v_k; then $\sum_{k=1}^{K} \phi_k = J$. This is a form of fertility parameter similar to those in Models 3 and 4.

As above, phrases can be spontaneously inserted during translation and not be a direct translation of any source word, and these are viewed as being generated by the NULL token. A *hallucination sequence* h_1^K is defined such that

$$h_k = \begin{cases} 0, & v_k \text{ generated by NULL} \\ 1, & v_k \text{ generated by } e_{a_k} \end{cases} \tag{3.30}$$

This also allows us to keep track of the last source word to generate a phrase, even when the current phrase is generated by NULL, so we don't interrupt the dependencies of the Markov model. It has a similar effect to the null states described in Section 3.5. The transition probabilities are then given by

$$a(a_k|a_{k-1}, h_k, I) = \begin{cases} 1, & a_k = a_{k-1}, h_k = 0 \\ 0, & a_k \neq a_{k-1}, h_k = 1 \\ \bar{a}(a_k|a_{k-1}, I), & h_k = 1 \end{cases} \tag{3.31}$$

The overall joint probability of sentence f_j, phrase sequence v_1^K and alignment $\mathbf{a} = (\phi_1^K, a_1^K, h_1^K, K)$ given the source sentence e_1^I is

$$\begin{aligned} & \mathrm{P}(f_1^J, J, v_1^K, K, a_1^K, h_1^K, \phi_1^K|e_1^I) = \\ & \mathrm{P}(J|e_1^I) && \text{Sentence length} \\ \times\ & \mathrm{P}(K|J, e_1^I) && \text{Phrase count} \\ \times\ & \mathrm{P}(a_1^K, h_1^K, \phi_1^K|K, J, e_1^I) && \text{Word-to-phrase alignment} \\ \times\ & \mathrm{P}(v_1^K|a_1^K, h_1^K, \phi_1^K, K, J, e_1^I) && \text{Word-to-phrase translation} \\ \times\ & \mathrm{P}(f_1^J|v_1^K, a_1^K, h_1^K, \phi_1^K, K, J, e_1^I) && \text{Target phrase segmentation} \end{aligned} \tag{3.32}$$

We make a number of simplifying assumptions. These are discussed, along with the models used for each of the component probabilities, in the following sections.

Sentence length: We assume that the length of the target sentence depends only on the length of the source sentence:

$$\mathrm{P}(J|e_1^I) = \epsilon(J|I) \tag{3.33}$$

This is not needed during alignment of sentences of fixed length however, so it is ignored.

Phrase count: Having chosen a length for the target sentence, we then determine the number of phrases it is divided into; the number of phrases, K, is assumed to depend only on the length of the target sentence:

$$\mathrm{P}(K|J, e_1^I) = \mathrm{P}(K|J) \propto \eta^K. \tag{3.34}$$

K is at most J since each phrase must contain at least one word; the parameter $\eta \geq 1$ is fixed and controls the number, and consequently the length, of the phrases. A larger value of η results in a large number of short phrases. Setting $\eta = 1$ results in a uniform distribution, so splitting a sentence of length J into J single word phrases is as likely as the entire sentence being a phrase.

Word-to-phrase alignment: The alignment is a Markov process that specifies the lengths of the target phrases and their alignment to the source words. The hallucination process h_k is an independent and identically distributed process, with a parameter p_0 that can be tuned to control the number of phrases that align to the NULL token.

$$d(h) = \begin{cases} p_0, & h = 0 \\ 1 - p_0, & h = 1 \end{cases} \tag{3.35}$$

Given the length of the source sentence, we assume that a_k depends only on a_{k-1} and the hallucination variable h_k. The phrase length model $n(\phi|e)$ is a fertility parameter that governs the length of the phrases that are produced by the source word e; a value for $n(\phi|e)$ is stored for each $\phi \in \{1, \ldots, N_{\max}\}$ and source word $e \in V_S$. Here, $N_{\max}$ is the maximum phrase length permitted by the model; by setting a maximum phrase length of 1 in the word-to-phrase alignment model, we obtain a word-to-word HMM alignment model.

The probability is given by

$$\begin{aligned} \mathrm{P}(a_1^K, h_1^K, \phi_1^K | K, J, e_1^I) &= \prod_{k=1}^{K} \mathrm{P}(a_k, h_k, \phi_k | a_{k-1}, \phi_{k-1}, e_1^I) \\ &= \prod_{k=1}^{K} a(a_k|a_{k-1}, h_k, I) d(h_k) n(\phi_k|e_{a_k}) \end{aligned}$$

Word-to-phrase translation: Having determined the length and position of the target phrases, it remains to produce the actual words within these target phrases as translations of the source words to which they are aligned. The translation probability is given by context-independent word-to-word translation

$$\mathrm{P}(v_1^K | a_1^K, h_1^K, \phi_1^K, K, J, e_1^I) = \prod_{k=1}^{K} \mathrm{P}(v_k | e_{a_k}, h_k, \phi_k) \tag{3.36}$$

Writing $v_k = v_k[1], \ldots, v_k[\phi_k]$ for the k^{th} phrase, the probability of generating that phrase is,

$$\mathrm{P}(v_k | e_{a_k}, h_k, \phi_k) = \prod_{l=1}^{\phi_k} t(v_k[l] | h_k \cdot e_{a_k}) \tag{3.37}$$

where $h_k \cdot e_{a_k} = \begin{cases} e_{a_k}, & h_k = 1 \\ \texttt{NULL}, & h_k = 0 \end{cases}$ and $t(f|e)$ is the word-to-word translation probability.

Word-to-phrase translation with bigram translation probabilities: Under the above model, the translation probability $t(f|e)$ is a context-independent unigram model (i.e. each target word in a phrase depends only the source word from which it is generated rather than other words in the phrase) and the order of words in the phrase is arbitrary. This is clearly not an accurate assumption, so a more sophisticated model that captures target language context is also proposed. Bigram translation probabilities $t(f|f', e)$ are used: under this model, each word in a target phrase is allowed to depend on the previous word in the target phrase as well as the source word from which it is generated. The probability

$$\mathrm{P}(v_k | e_{a_k}, h_k, \phi_k) = \prod_{l=1}^{\phi_k} t(v_k[l] | v_k[l-1], h_k \cdot e_{a_k}) \tag{3.38}$$

This allows some integration of target language context into the translation model, thereby helping to prevent unlikely word combinations within the phrase.

Target phrase segmentation: This expression is included to ensure that the sequence of phrases generated matches the target sentence, i.e. that $f_1^J = v_1^K$. Hence

$$\mathrm{P}(f_1^J | v_1^K, a_1^K, h_1^K, \phi_1^K, K, J, e_1^I) = 1_{\{f_1^J = v_1^K\}} = \begin{cases} 1, & \text{if } f_1^J = v_1^K \\ 0, & \text{otherwise} \end{cases} \tag{3.39}$$

Model training: Although it has a complicated state space compared to the word-to-word alignment model, the word-to-phrase model is an HMM; hence the standard Baum-Welch algorithm for HMM parameter estimation can be applied. This means it is efficient to train. Deng (2005a) describes the full details of the training procedure.

3.7 Other forms of alignment model

The models described above are generative, i.e. they view one sentence as being generated from another by a sequence of steps. Moore (2005a) defines an alignment model based on heuristic word association statistics such as log-likelihood ratio and features based on co-occurrence counts. This is extended by Moore (2005b) to form a discriminative model. Other discriminative models have been built, using existing generative models as features (Blunsom and Cohn, 2006; Taskar et al., 2005). Discriminative alignment models are discussed further in Chapter 6.

The alignment models described so far make fairly restrictive assumptions due to the way in which they are constructed. The IBM models are many-to-one, i.e. one source word can align to multiple target words but a target word can align to at most one source word. They also assume that words are arranged into phrases consisting of consecutive words. Discriminative models generally make similar assumptions or use features from generative models making these assumptions. Fraser and Marcu (2007) introduce a new alignment model, LEAF, which addresses some of the problems caused by these assumptions. LEAF is a generative m-to-n model that models the non-consecutive alignments that often occur in human alignments.

The structure of a sentence in a language is modelled as follows: each head word is linked to zero or more non-head words and each non-head word is linked to exactly one head word. Source words can be head words, non-head words or deleted (i.e. no translation appears in the target sentence); target words can be head words, non-head words or spurious (i.e. not generated from any word in the source sentence). The model tries to provide a representation of semantic features needed to determine translational correspondence and the alignment process ensures head words are aligned with head words. The model can be trained using EM Viterbi training after initialisation from HMM alignments, and the generative model can be decomposed so its components in each direction can be used as features of a log-linear model for discriminative training.

3.7.1 Constrained EM training

During standard EM, the training procedure does not necessarily have the effect of using the model parameters for their intended purposes. The parameters are simply estimated to fit the training data, and this can guide the model to focus on explaining irrelevant patterns in the data. Graca et al. (2008) introduce a constrained EM algorithm for unsupervised training of alignment models, which allows constraints to be placed on the model parameters so that they take the intended meaning. This is accomplished by replacing the alignment posterior distribution within EM with a distribution as close as possible to it that also satisfies the constraints specified.

For simplicity, write $\mathbf{x}$ for the observed variables $\mathbf{e}, \mathbf{f}$. If we take context into account, $\mathbf{x}$ can also include the source word contexts $\mathbf{c}$. We then replace the posterior alignment probability

distribution $p_\theta(\mathbf{a}|\mathbf{x})$ by probability distribution $q(\mathbf{a})$ satisfying bounding constraints on the expectation of the form

$$E_q[f(\mathbf{x}, \mathbf{a})] \leq b. \tag{3.40}$$

We can specify equality by using the additional constraint $E_q[-f(\mathbf{x}, \mathbf{a})] \leq -b$, and can specify multiple constraints in one equation by using a vector of functions

$$E_q[\mathbf{f}(\mathbf{x}, \mathbf{a})] \leq \mathbf{b}. \tag{3.41}$$

We choose the distribution $\hat{q}$ closest to p_θ that satisfies these constraints, i.e.

$$\hat{q} = \underset{q}{\operatorname{argmin}} \operatorname{KL}\left(q(\mathbf{a})||p_\theta(\mathbf{a}|\mathbf{x})\right) \text{ s.t. } \mathbf{E}_q[\mathbf{f}(\mathbf{x}, \mathbf{a})] \leq \mathbf{b}, \tag{3.42}$$

where KL is the Kullback-Leibler divergence. This technique can be used to force the models trained in each translation direction to agree, the intuition being that the models make different mistakes and forcing them to agree rules out errors made by only one model (Ganchev et al., 2008).

3.8 Evaluation of alignment quality

In order to determine the performance of our alignment models, we wish to evaluate the quality of word alignments against those produced by human translators. Our primary aim when developing a system is to maximise the translation quality measured by BLEU score obtained when translating the test data. We hope that good quality alignments will lead to good translations, and our aim is then to choose the measure of alignment quality that correlates best with the BLEU score. In this way, we can investigate the effects of changes to the alignment model without performing end-to-end translation. We can also use a measure of alignment quality as a training criterion when training alignment models.

In theory, better alignment links under a metric should result in improved translation performance, when those links are used to build a machine translation system. However, while this can be the case, there are instances where improved alignment quality does not lead to better translations: this is discussed in Section 3.8.3.

Determining the translation correspondence is a difficult problem with no clearly-defined objective solution. One translator may view a word as a translation of another word, while another may view it as spurious and aligned to nothing in the other language. Therefore a further limitation of the use of manually-aligned data is that there may be inter-annotator disagreement (Melamed, 1998), or indeed intra-annotator disagreement if a translator does not always make the same decision regarding a word. However, examination of alignment quality relative to manual alignments has proved useful and we do not have a reliable alternative; therefore we make use of manual alignments for evaluation of our models.

3.8.1 Alignment Error Rate

The *Alignment Error Rate* (AER) is a metric for evaluating the quality of an alignment against a manually produced reference alignment. Given a set of word alignment links A and

a reference set of alignment links R, define the *precision* and *recall* as

$$\text{Precision}(A, R) = \frac{|R \cap A|}{|A|} \tag{3.43}$$

$$\text{Recall}(A, R) = \frac{|R \cap A|}{|R|} \tag{3.44}$$

The AER is defined as

$$\text{AER}(A, R) = 1 - \frac{2 \times \text{Precision}(A, R) \times \text{Recall}(A, R)}{\text{Precision}(A, R) + \text{Recall}(A, R)} = 1 - \frac{2 \times |A \cap R|}{|A| + |R|} \tag{3.45}$$

(Och and Ney (2000b) originally introduced AER and allowed annotators to specify a set S of *sure* links, which must be present in an alignment, and a set P of *possible* links, which may be present but need not be present. They defined $\text{Precision}(A, P) = \frac{|P \cap A|}{|A|}$ and $\text{Recall}(A, S) = \frac{|S \cap A|}{|S|}$ so that a recall error occurs if a sure link is not found, and a precision error occurs if a link is found that is not even in the list of possible links. However, simply using the set of sure links and modifying the definitions accordingly was found to give better results (Fraser and Marcu, 2006a)).

3.8.2 Weighting of precision and recall

Precision and recall are important properties of the alignments; a higher precision and lower recall leads to more phrases or rules being extracted. The AER of alignments used to generate the translation models does not always correlate well with the BLEU score, i.e. increases in word alignment quality measured by AER do not necessarily result in an increase in translation quality. Fraser and Marcu (2006a) explore the relationship between alignment quality and statistical machine translation performance, focusing on finding a metric for measuring alignment quality that correlates with, and is predictive of, machine translation quality measured by BLEU. The paper finds that unbalanced precision and recall are not penalised, and that it is possible to maximise AER by favouring precision over recall, i.e. by simply guessing few alignment links, and suggest an alternative to AER that correlates more closely with BLEU.

Alternative measures of alignment quality are proposed: the *F-measure with Sure and Possible* is

$$F_{SP}(A, P, S, \alpha) = \frac{1}{\frac{\alpha}{\text{Precision}(A,P)} + \frac{1-\alpha}{\text{Recall}(A,S)}} \tag{3.46}$$

and the *F-measure* is

$$F(A, S, \alpha) = \frac{1}{\frac{\alpha}{\text{Precision}(A,S)} + \frac{1-\alpha}{\text{Recall}(A,S)}}, \tag{3.47}$$

where the weight α determines the balance between the relative importance of precision and recall. A small value of α weights recall higher. We then use $1 -$ F-measure as a measure of alignment quality, and it can be used in the same way as AER (in fact, AER is simply a special case of F-measure with $\alpha = \frac{1}{2}$): a large value of $1 - F$ corresponds to many errors in alignment. F-measure is found to be the best-performing metric, subject to correct tuning of α.

3.8.2.1 Investigation of the properties of F-measure

In order to examine the relationship between AER and F-measure and their effects on alignment quality, we look at contours of equal F-measure. We aim to minimise AER or, equivalently, to maximise F-measure. Now, a constant F-measure means that

$$\frac{1}{\frac{\alpha}{P} + \frac{1-\alpha}{R}} = c \tag{3.48}$$

$$\implies \quad \frac{PR}{\alpha R + (1-\alpha)P} = c \tag{3.49}$$

$$\implies \quad PR = c\alpha R + c(1-\alpha)P \tag{3.50}$$

$$\implies \quad P(R - c + c\alpha) = c\alpha R \tag{3.51}$$

$$\implies \quad P = \frac{c\alpha R}{R - c(1-\alpha)} \tag{3.52}$$

Figure 3.4 shows the variation of F-measure with precision and recall, for varying values of α. Plotting lines of constant F-measure for $\alpha = 0.1, 0.5, 0.9$, we can see graphically the effect of changing these parameters. We aim to find the alignment model parameters that place the F-measure as high on the surface as possible. Since the models typically have precision and recall greater than 50%, we are most interested in the quadrant of the graph where precision > 0.5 and recall > 0.5. For small α the contours are close together on the recall axis, which means that a small change in recall produces a comparatively large change in F-measure, whereas a change in precision has little effect. This illustrates the importance of matching the alignment quality measure to the translation system, depending on whether it performs best with high-precision or high-recall alignments.

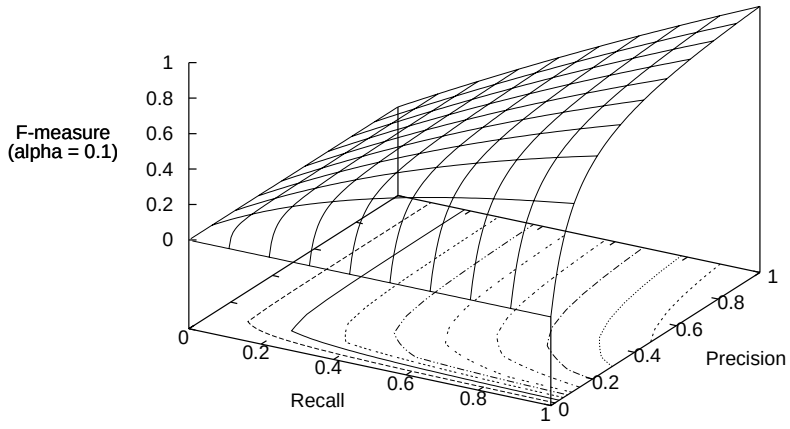
3.8.2.2 Extraction of links by posterior probability

If we extract alignments by posterior probability, we can adjust the probability threshold according to whether we favour precision or recall. A low threshold allows more links to be output, which increases recall but decreases precision; a high threshold means the link must have a high posterior probability in order to be output, so precision is high but recall is low. A large number of links causes fewer phrase pairs to be generated for translation. Experiments with the TTM showed that the posterior probability had very little effect on the overall translation quality, though there were large differences in the number of alignment links output and the number of phrases extracted using the alignments.

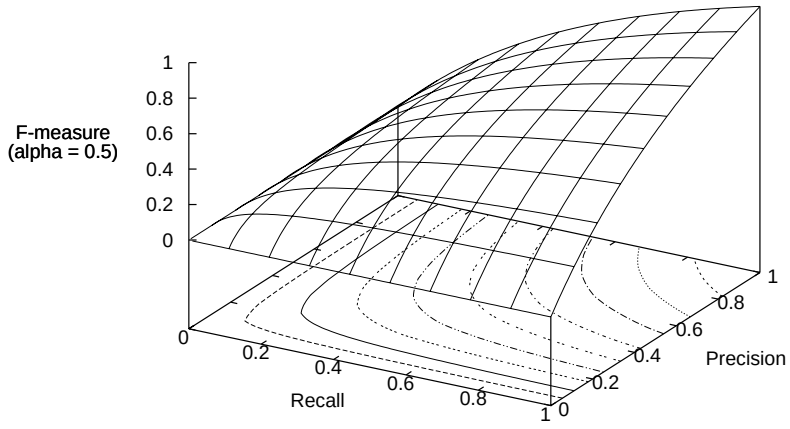
3.8.2.3 Evaluation of alignment models using the precision-recall curve

Our proposed method of comparison of alignment models is to use extraction of alignment links by posterior probability to plot a precision-recall curve. If the curve of the first model falls entirely outside that of the second, we can conclude that the first model produces better alignments: for a given recall, its precision is higher and for a given precision, its recall is higher. Regardless of the choice for the posterior extraction threshold λ, the first model will produce better alignments measured by precision, recall or F-measure.

$\alpha = 0.1$:



$\alpha = 0.5$:



$\alpha = 0.9@$

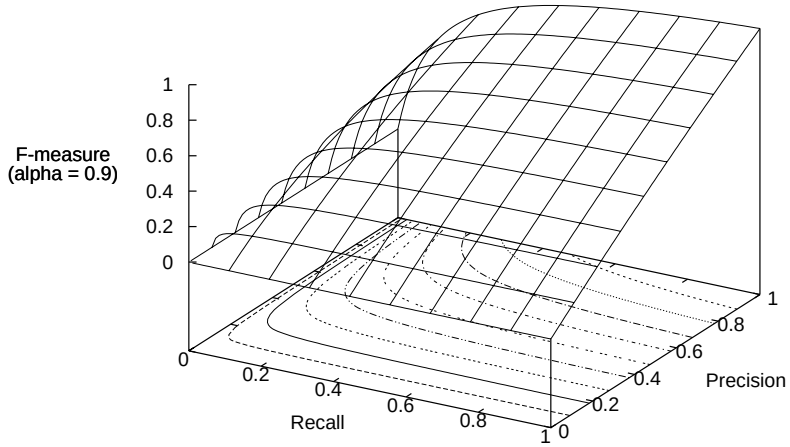


Figure 3.4: Variation of F-measure with precision and recall, for varying values of α

3.8.3 Relationship between alignment quality and translation quality

It is an open matter of debate as to whether an improvement in alignment quality actually leads to an improvement in translation quality. A major difficulty is that of finding an alignment evaluation metric that correlates strongly with translation quality, measured by BLEU, TER, human evaluation or other metrics described in Section 2.8. The alignment characteristics required depend on the system being used and the language pair we are translating between. Various papers report that high precision alignments are better for phrase-based SMT (Ayan and Dorr, 2006; Chen and Federico, 2006); high recall alignments are suited to n-gram translation (Mariño et al., 2006); high precision alignments are better for French to English but high recall performs better for Arabic to English, both with phrase-based systems (Fraser and Marcu, 2006a).

Ganchev et al. (2008) examine whether better alignments do indeed lead to improved translation, using the constrained models proposed by Graca et al. (2008). They find that the improved quality of alignments does lead to improved translation. However, they also extract links by the posterior probability of the link and show that the precision-recall curve for the new models is entirely outside that for the old models. Therefore an improvement in AER means that there is an improvement in precision and recall, which means there is also an increase in F_α-measure for every value of α and the relative effects of precision and recall are not tested.

Vilar et al. (2006) perform experiments using two different types of alignment model and find that manipulation of alignments can significantly degrade the AER without adversely effecting translation quality. Their salient conclusion is that future work on alignment oriented to machine translation should always report results on translation quality. Ittycheriah and Roukos (2005) present a maximum entropy model for word alignment that provides large gains in AER over HMM and Model 4 alignment models, but find no improvement in translation quality.

The evidence presented suggests that the optimum value for α, and therefore optimum balance between precision and recall, is highly language-dependent, and even dependent on the data set used. The correlation of alignment quality metrics with translation quality metrics other than BLEU may also lead to different results. This raises doubts as to the utility of using alignment quality as an indicator of translation quality and suggests that alignment models should be evaluated to as great an extent as possible by the quality of translation they produce.

3.8.3.1 Consistent phrase error rate

Ayan and Dorr (2006) investigate the relationship between alignment quality and translation quality. The *consistent phrase error rate* (CPER) is presented as an alternative to evaluation by AER that correlates better with translation quality. Rather than directly comparing the quality of the alignments obtained, it looks at the phrases produced by the hypothesised alignments compared to those phrases extracted when the reference manual alignments are used. Let P_A be the set of phrases generated by a hypothesis alignment and P_R be the set of phrases generated by a manual alignment and define the precision and recall as

$$\text{Precision}(P_A, P_R) = \frac{|P_R \cap P_A|}{|P_A|}; \text{Recall}(P_A, P_R) = \frac{|P_R \cap P_A|}{|P_R|}. \tag{3.53}$$

Figure 3.5: Two alignments that have the same translational correspondence but different AER and F-measure scores.

Then

$$\text{CPER} = 1 - \frac{2 \times \text{Precision}(P_A, P_R) \times \text{Recall}(P_A, P_R)}{\text{Precision}(P_A, P_R) + \text{Recall}(P_A, P_R)}. \tag{3.54}$$

Contrary to (Fraser and Marcu, 2006a), they conclude that precision-oriented alignments yield better translation output than recall-oriented alignments, translating more words and using longer phrases.

3.8.4 Computation of AER and F-measure

Sometimes two alignments which lead to the same phrases being extracted can have different AER and F-measure scores (Fraser and Marcu, 2007). This is because some links between phrase pairs are not essential when determining the phrase pairs, but are implied by the links already present. An example of this is shown in Figure 3.5. Taking Figure 3.5 as a simple example, the AER of the first alignment is $1 - \frac{2\times3}{3+4} = \frac{1}{7}$ (assuming the second alignment is the reference alignment) whereas that of the second is $1 - \frac{2\times4}{4+4} = 0$, yet these two alignments will produce the same phrase pairs with the same phrase-to-phrase translation probabilities.

Given an alignment A, we define a square Boolean matrix X whose entries represent arcs, i.e.

$$x_{mn} = \begin{cases} 1, & \text{if there is an arc from } m \text{ to } n \\ 0, & \text{otherwise} \end{cases} \tag{3.55}$$

for $1 \leq m, n \leq I + J$. This is the *adjacency matrix* of the graph. X is given by

$$\begin{aligned} x_{i,I+j} &= 1, \text{ for } (i,j) \in A \\ x_{i,i} &= 1, \ \forall i \text{ (i.e. each source word is linked to itself)} \\ x_{I+j,I+j} &= 1, \ \forall j \text{ (i.e. each target word is linked to itself)} \\ x_{m,n} &= 0, \text{ otherwise} \end{aligned}$$

We can then use Warshall's algorithm (Warshall, 1962) to find the transitive closure $\bar{A}$ of the graph, i.e. the smallest graph such that any two nodes that are linked by any path are also linked directly. This has the effect of adding links between words that are implicitly linked to each other but do not have an explicit link., but does not change the collection of phrases extracted from an alignment using the *phrase-extract* algorithm. For proof of this, consider an alignment $(i,j) \in \bar{A} \setminus A$ added by taking the transitive closure.

- Assume that $i \in [i_1, i_2]$, i.e. it is part of the source phrase in the phrase pair $e_{i_1}^{i_2}, f_{j_1}^{j_2}$. If $j \notin [j_1, j_2]$, then j cannot be aligned by A to any word in $[i_1, i_2]$ (by properties of the phrase extraction): therefore j is in the phrase $[j_1, j_2]$ defined by A. This means that taking the transitive closure does not change any existing phrase pairs.

Input: Adjacency matrix X of a graph G
Output: Adjacency matrix X^* giving the transitive closure of G
foreach $k = 1, \ldots, I + J$ **do**
  foreach $m = 1, \ldots, I + J$ **do**
    foreach $n = 1, \ldots, m - 1$ **do**
      $x_{mn} = x_{mn} \vee (x_{mk} \wedge x_{kn})$
    end
  end
end

Figure 3.6: Warshall's Algorithm for finding the transitive closure of a graph

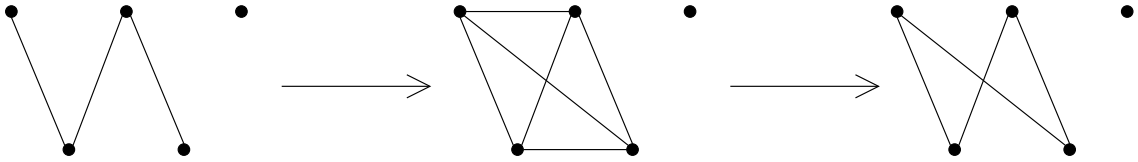


Figure 3.7: Finding the closure of an alignment: first the transitive closure of the alignment graph is found, then the intra-sentence links are removed.

- Assume that $i \notin [i_1, i_2]$. By symmetry with the above, $j \notin [j_1, j_2]$. This means that no new phrase pairs are created by the addition of the new alignment links.

Warshall's Algorithm, shown in Figure 3.6, essentially works by starting with the original graph and looping through each node in turn, testing to see if any two other nodes are linked via a path through that node; a link is added directly between those nodes if they are linked. Since the arcs are undirected, the adjacency matrix X is symmetric, i.e. $x_{mn} = x_{nm}$, and we can assume it is triangular to reduce computation. The output of the algorithm and the alignments resulting from the transitive closure are illustrated in Figure 3.7.

3.9 Preparing word alignments for use in translation

Once we have built the alignment models, we use them to align the corpus of training data and use the alignments to extract phrases and rules for use in a machine translation system. This section discusses the ways in which alignments can be obtained from the models.

3.9.1 Generation of alignments from models

The most common method of extracting alignment links from many-to-one models is to use the **Viterbi alignment**, the single alignment sequence assigned highest probability by the model:

$$\hat{a}_1^J = \underset{a_1^J}{\operatorname{argmax}} \mathrm{P}(a_1^J | e_1^I, f_1^J) = \underset{a_1^J}{\operatorname{argmax}} \mathrm{P}(f_1^J, a_1^J | e_1^I) \tag{3.56}$$

The alignment links output are then

$$A_{\text{Viterbi}} = \{(\hat{a}_j, j) : 1 \leq \hat{a}_j \leq I\}. \tag{3.57}$$

We exclude alignments to the null token, i.e. $a_j = 0$, as alignment to it is simply a method of indicating that the target word does not align to any word in the source sentence. One problem with the Viterbi alignment is that it enforces the many-to-one constraint of the alignment model, which may not be accurate. Extracting alignment links by the **posterior probability** of the individual alignment links can alleviate this issue. Consider the case where one word in the target language has the same meaning as a two word phrase in the source language, for example the English phrase *dentist's appointment* translates to the German compound noun *Zahnarzttermin*. Under the Viterbi alignment, *Zahnarzttermin* can only be aligned to only one of *dentist's* or *appointment*; using extraction by posterior probability, both links may have a sufficiently high probability that they are included. For each possible alignment link (i, j), we evaluate the posterior probability of that link under the model and include the link if the probability is above a set threshold λ. Then

$$A_{\text{posterior}}(\lambda) = \{(i,j) : 1 \leq i \leq I, 1 \leq j \leq J, P(a_j = i|f_1^J, e_1^I) > \lambda\} \tag{3.58}$$

For Model 1, the posterior probability of a link is given by

$$\mathrm{P}_{\mathrm{M1}}(a_j = i|f_1^J, e_1^I) = \frac{t(f_j|e_i)}{\sum_{i'=0}^{I} t(f_j|e_{i'})}. \tag{3.59}$$

For Model 2, it is

$$\mathrm{P}(a_j = i|f_1^J, e_1^I) = \frac{a(i|j, I, J)t(f_j|e_i)}{\sum_{i'=0}^{I} a(i'|j, I, J)t(f_j|e_{i'})} \tag{3.60}$$

For the word-to-word HMM,

$$\mathrm{P}(a_j = i|f_1^J, e_1^I) = \frac{\alpha_j(i)\beta_j(i)}{\sum_{i=1}^{I} \alpha_J(i)}, \tag{3.61}$$

where $\alpha_j(i)$ and $\beta_j(i)$ are the forward and backward variables computed by the Baum-Welch algorithm. All of these probabilities are computed during standard EM training and can easily be used to extract alignment links.

3.9.2 Symmetrisation

A property of many alignment models examined is that they are asymmetric and produce many-to-one alignments only, i.e. one source word can be aligned to multiple target words but each target word can be aligned to only one source word. This is a problem as alignments are frequently many-to-many or one-to-many. Consider swapping the source and target languages: the many-to-one alignments will become one-to-many.

To account for the fact that the models are asymmetric and to obtain a more balanced set of alignment links, we train models and perform alignment in both directions, leading to two sets of alignments:

$$A_{E\to F} = \{(a_j, j) : a_j > 0\}, \text{ and} \tag{3.62}$$

$$A_{F\to E} = \{(i, a'_i) : a'_i > 0\}. \tag{3.63}$$

Again we exclude alignments to the null token as we do not want it to appear in phrases. These can then be combined in a number of ways to give the set of alignments that is used to train the machine translation system:

- Intersection: $A_{\cap} = A_{E \to F} \cap A_{F \to E}$. This gives high precision (many hypothesised links are correct), since the alignments are judged to be correct by both models, but low recall (some correct links may be output by both models).
- Union: $A_{\cup} = A_{E \to F} \cup A_{F \to E}$. This gives high recall, since it contains all alignment links predicted by the two models, but low precision.
- A balance between precision and recall can be obtained by using an alternative method. The intersection is taken and heuristic methods are used for adding alignments that occur only in the output of one of the models. Generally, alignment links adjacent to existing links are added as long as either of the two words being joined is unaligned. Methods for doing this are presented by Koehn et al. (2003).

This process is known as *symmetrisation*. Koehn et al. (2003) investigate the use of IBM Models 1 to 4 to generate alignments and find that, with their phrase-based translation system, the choice of alignment heuristic has a more significant effect on translation quality than the models used to create the initial alignments. The determination of the phrase pair inventory is therefore of high importance.

Fossum et al. (2008) propose a new method for finding a set of alignment links: they begin with the union alignment and train a model that removes links from the union. They use syntactic constraints based on the rules extracted for a syntax-based system, structural conditions on the alignment and lexical features to train a discriminative model, using a set of manually-aligned sentences. This new set of links is used to initialise a syntactic translation model.

Huang (2009) presents a method of using the posterior probability of alignment links (under one or more alignment models, possibly in both directions) to determine confidence measures for alignment at the sentence and individual alignment link level. They aim to improve the alignment quality by selecting high confidence alignment links from multiple word alignments of the same sentence pair.

3.10 Extraction of phrases for use in phrase-based translation

Phrase-based translation requires a collection of phrase pairs (comprising one source language phrase and one target language phrase) that may be translations of each other. These pairs of phrases are extracted from parallel text using the alignment models described above: the word-to-word alignments can be used to produce phrase pairs using the simple fact that two phrases align if the words within them align. The collection of phrase pairs is known as the *Phrase Pair Inventory* (PPI). A limit is set so that phrases in the inventory do not exceed a maximum length M; this avoids unnecessary complexity.

It should be noted at this point that the technique discussed in this section is purely algorithmic and the phrases are not syntactically motivated; they are simply sequences of words, which can be aligned and therefore translated. While this may not appear to be elegant, it has been shown to work effectively and has the added advantage that we do not need syntactic models for the languages involved.

3.10.1 The phrase-extract algorithm

The *phrase-extract* algorithm (Och, 2002) works on the premise that two phrases align if the words within them align. Given a generalised word alignment $A_{\mathbf{e},\mathbf{f}}$, for parallel text sentences $\mathbf{e}, \mathbf{f}$, it finds all subsequences of words $e_{i_1}^{i_2}$ and $f_{j_1}^{j_2}$ that satisfy

$$\forall (i,j) \in A_{\mathbf{e},\mathbf{f}}, i \in [i_1, i_2] \iff j \in [j_1, j_2], \tag{3.64}$$

i.e. all words in $f_{j_1}^{j_2}$ are aligned to a word in $e_{i_1}^{i_2}$ (if they are aligned to any word at all), and all words of $e_{i_1}^{i_2}$ are aligned to a word in $f_{j_1}^{j_2}$ (if they are aligned). We also require that there is at least one alignment link between the phrases, i.e. that for some $(i,j) \in A$ for some $i \in\in [i_1, i_2], j \in [j_1, j_2]$. The phrase pairs that satisfy these criteria are put into the PPI.

3.10.2 Phrase pair induction

With the procedures detailed above, there are foreign phrases that appear in the parallel text that will not be included in the PPI because a suitable English phrase has not been found due to word alignment errors. Deng and Byrne (2005a) introduce a new method for finding phrase pairs, where the selection of phrase pairs is not based on a single alignment. Given subsequences of words $e_{i_1}^{i_2}$ and $f_{j_1}^{j_2}$ from the parallel text sentences e_1^I and f_1^J, define

$$A(i_1, i_2; j_1, j_2) = \{a_1^J : a_j \in [i_1, i_2] \iff j \in [j_1, j_2]\}. \tag{3.65}$$

This is the set of alignments such that $(e_{i_1}^{i_2}, f_{j_1}^{j_2})$ is a phrase pair. We can find the probability of the phrases occurring as translation pairs:

$$\mathrm{P}(f_1^J, A(i_1, i_2; j_1, j_2)|e_1^I) = \sum_{a_1^J \in A(i_1,i_2;j_1,j_2)} \mathrm{P}(f_1^J, a_1^J|e_1^I) \tag{3.66}$$

We can then find the *phrase-to-phrase posterior distribution*

$$\mathrm{P}_{E\to F}(A(i_1, i_2; j_1, j_2)|f_1^J, e_1^I) = \frac{\mathrm{P}(f_1^J, A(i_1, i_2; j_1, j_2)|e_1^I)}{\mathrm{P}(f_1^J|e_1^I)}, \tag{3.67}$$

where $\mathrm{P}(f_1^J|e_1^I) = \sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I)$. An advantage of using the HMM model is that the posterior distribution can be found easily using a modified form of the forward algorithm; using Model 4, there is no efficient way to calculate the expression in 3.66.

Since we have alignments in both directions, the phrase-to-phrase posterior distribution can be found in both directions; we define an analogue to A for target to source alignments:

$$A'(i_1, i_2; j_1, j_2) = \{a'^I_1 : i \in [i_1, i_2] \iff a'_i \in [j_1, j_2]\}. \tag{3.68}$$

The phrase-to-phrase posterior distribution for the $F \to E$ direction is then

$$\mathrm{P}_{F\to E}(A'(i_1, i_2; j_1, j_2)|f_1^J, e_1^I) = \frac{\sum_{a'^I_1 \in A'(i_1,i_2;j_1,j_2)} \mathrm{P}(e_1^I, a'^I_1|f_1^J)}{\mathrm{P}(e_1^I, a'^I_1|f_1^J)} \tag{3.69}$$

Both of these phrase-to-phrase posterior probabilities are then used to select phrases for inclusion in the PPI, which is built up as follows. First, the Viterbi alignments are found in each direction and are merged as in Section 3.9.2. The *phrase-extract* algorithm is then used to extract phrase pairs of phrases of maximum length M; the resulting PPI is known as the *Viterbi Phrase-Extract PPI*. The posterior probabilities are then used to add to the PPI using a number of heuristic criteria. Various parameters can be adjusted to determine the balance between coverage and ensuring the quality of the phrases extracted.

3.10.3 Phrase translation table

The above process results in a set of source language phrases $\mathcal{V}_{\text{src}}$ and a set of target language phrases $\mathcal{V}_{\text{tgt}}$. For translation we require an estimate, for a source language phrase $\mathbf{u} \in \mathcal{V}_{\text{src}}$ and target language phrase $\mathbf{v} \in \mathcal{V}_{\text{tgt}}$, of the *phrase-to-phrase translation probability* $\mathrm{P}(v|u)$. We use the maximum likelihood estimate with no smoothing:

$$\mathrm{P}_{\mathrm{ML}}(v|u) = \frac{c(u,v)}{\sum_{v' \in \mathcal{V}_{\text{tgt}}} c(u,v')}, \tag{3.70}$$

where $c(u,v)$ is a count of the number of times that the source phrase u is paired with the target phrase v in the training corpus.

3.11 Extraction of phrases for hierarchical phrase-based translation

The rules for a hierarchical phrase-based system are also extracted from a word-aligned parallel corpus $\{\mathbf{e}^{(s)}, \mathbf{f}^{(s)}, A^{(s)}\}_{s=1}^{S}$. We extract rules consistent with the alignments in two steps: in the first step we extract *initial phrase pairs* using the `phrase-extract` algorithm, in the second step we obtain rules by looking for phrases that contain other phrases and replacing the subphrases with non-terminal symbols.

The set of rules is then defined to be the smallest set satisfying:

1. If $\langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$ is an initial phrase pair then

$$X \to \langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle \tag{3.71}$$

   is a rule.

2. If $X \to \langle \gamma, \alpha \rangle$ is a rule (where γ and α are both sequences of terminals and non-terminals) and $\langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$ is an initial phrase pair such that $\gamma = \gamma_1 f_{j_1}^{j_2} \gamma_2$ and $\alpha = \alpha_1 e_{i_1}^{i_2} \alpha_2$, then

$$X \to \langle \gamma_1 X_{\boxed{\mathrm{k}}} \gamma_2, \alpha_1 X_{\boxed{\mathrm{k}}} \alpha_2 \rangle \tag{3.72}$$

   is a rule (where k is an index of non-terminal not used in γ and α).

3.11.1 Filtering of rules and model training

This process generates a large number of rules, which causes slow training and decoding if all of them are used. It also causes problems during decoding as many derivations involving similar rules have exactly the same translation. The rules are filtered in a number of ways to avoid this.

The length of phrases is limited, the number of non-terminal symbols in a rule is restricted and a further limit is put on the total number of non-terminals and terminals in a rule. Rules containing adjacent non-terminal symbols in the source language are removed. Each rule is forced to have at least one pair of aligned words outside the non-terminal groups to ensure some kind of translational correspondence. Iglesias et al. (2009a) filter further by excluding

words with two non-terminals with the same order on the source and target side and consider only the 20 most frequent translations for each rule (Iglesias et al., 2009b).

In order to estimate the probability of a rule occurring, we would like to know the number of times it has been applied in the translation of the parallel text. However, each sentence has more than one derivation using the rules and we do not know which derivation is the appropriate one. Therefore, the rule counts are estimated using heuristics that approximate the numbers of times the rules are used in the training data.

3.12 Implementation of alignment models

The alignment models investigated in this thesis are implemented using the Machine Translation Toolkit (MTTK) (Deng and Byrne, 2005b). This is an alignment modelling package capable of training Model 1, Model 2, word-to-word HMM and word-to-phrase HMM alignment models, including those with bigram translation tables.

In our experiments, we follow the standard MTTK training procedure, starting with Model 1 and gradually increasing model complexity via Model 2 and the word-to-word HMM; then a sequence of word-to-phrase HMM models with increasing maximum phrase length, followed by a word-to-phrase HMM model. The exact sequence used is as follows (Deng and Byrne, 2006):

- Initialise translation tables as uniform distributions;
- Train IBM Model 1 parameters with 10 iterations of EM;
- Train IBM Model 2 parameters with 5 iterations of EM (starting with uniform alignment probabilities);
- Initialise word-to-word HMM alignment model using word alignment counts from Model 2 Viterbi alignments of the parallel text;
- Estimate word-to-word HMM parameters with 5 iteration of EM;
- For $N = 2, 3$ up to the maximum phrase length $N_{\max}$: estimate parameters of word-to-phrase HMM alignment models using 5 iterations of EM.

3.13 Summary

Alignment models are an important part of machine translation, since a corpus of word-aligned parallel text is required for initialisation of the translation model. There has been much work investigating alignment models, with the aim of producing accurate alignments while making approximations to keep models sufficiently simple that their parameters can be accurately estimated and they are computationally tractable. This chapter introduces the alignment models that will be used for experiments in this thesis, and describes how they are used to produce rules and phrases for statistical machine translation systems.

Alignment quality is often measured using various metrics, with the aim being that an alignment that scores well under a particular metric will also lead to a good translation system. The optimum metric appears to depend on language, size of data set and the type of translation system being used. Alignment quality metrics can provide a good guide to translation quality and is useful for comparison of alignment models. However, the ultimate measure of an alignment model's utility is the quality of translation when a system is trained on its output.

CHAPTER 4

Rescoring Translation Lattices with Alignment Models

4.1 Introduction

This chapter introduces algorithms for using alignment models to rescore lattices of translation hypotheses from a machine translation system. Prior to this research, the MTTK alignment models described in Chapter 3 were used only to generate alignments between the sentence pairs in the training data, with these alignments used to extract phrase pairs and estimate phrase-to-phrase translation probabilities for use in translation. The alignment models themselves were not used in the translation for computational reasons: since the number of possible alignments increases exponentially with the length of the sentences, the decoding process (i.e. determining the best translation given the sentence in the foreign language) is NP-complete even for simple statistical models (Knight, 1999). The alignment models used by the Transducer Translation Model (TTM) during hypothesis generation and decoding are simple and constrained to maintain computational tractability, so although the order of words and phrases varies between languages, it is not possible to represent this accurately during translation. For example, the MJ-1 (Maximum Jump 1) reordering model allows transpositions of phrases but allows a phrase to move at most one place along a sentence. For more information on the TTM, see the description in Chapter 2.

We aim to improve the quality of the hypotheses output by the TTM system, using information from the more sophisticated MTTK alignment models during the translation process itself. One method of doing this is to use the models to rescore translation hypotheses. This chapter accomplishes two things: the first is lattice-to-string alignment, leading to the formulation of an algorithm for producing a lattice that encodes all possible alignments of each sentence in a lattice of translation hypotheses with a source sentence. The second is the use of such lattices for rescoring lattices using alignment models.

4.2 Lattice representation of translation hypotheses

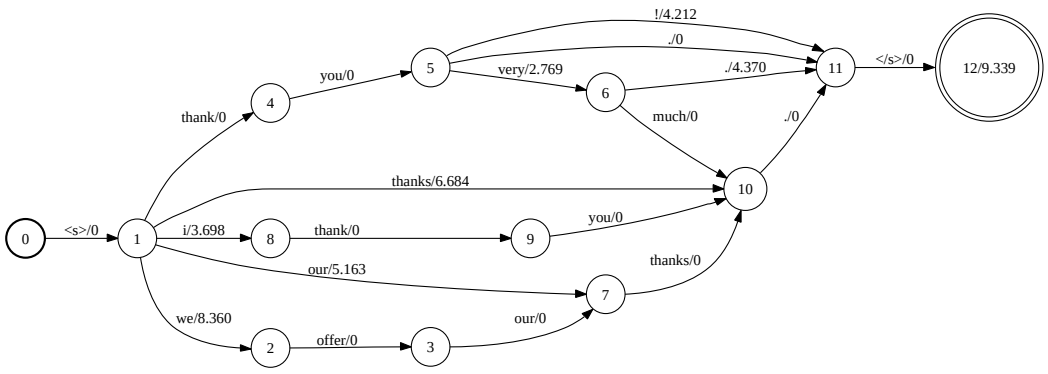
An intermediate stage in the translation process is a lattice of possible translation hypotheses. This lattice is a weighted finite state acceptor that accepts a sequence of words if it matches one of the translation hypotheses; the cost of a path through the acceptor is the cost of the sentence associated with that path. This lattice representation is very space-efficient, since many translation hypotheses are similar and two sentences with a single difference can be represented by two paths through the lattice differing by only one arc.

Figure 4.1 shows a lattice of hypotheses generated by the TTM as a translation of the short French sentence `<s> merci . </s>`. The sentence start tag `<s>` and sentence end tag `</s>` are symbols inserted to indicate the beginning and end of the sentence. Each arc has associated with it a label corresponding to a word of English (e.g. `very`) and a cost (e.g. 2.769). The final state also has a cost associated with it (9.339 in this case). The lattice can be pruned to remove unlikely hypotheses while retaining those assigned high probability by the model (see Figure 4.2).

A common method of improving translation quality is the use of n-best list rescoring, where the top n translations (i.e. those with minimal cost, or maximal probability) are output by the the translation model and additional sources of information are used for rescoring, such as a more accurate language model that is too complex to use in first-pass decoding. Initial experiments to rescore n-best lists of translation hypotheses using alignment models were successful, resulting in gains of over 2 BLEU points over the baseline system. However, this approach is inefficient, since sentences with identical substrings are rescored separately rather than using the similarity to perform the calculation only once. It can also result in search errors, where the best hypothesis according to the rescoring objective is lost, because it did not originally appear in the top n sentences and other sentences are discarded in the formation of the n-best list. Lattice-based rescoring is a technique that can be used to overcome such difficulties, as long as a suitable algorithm for doing so can be found.

4.3 Encoding word-to-word alignment information in translation lattices

The structure of the lattices of hypotheses output by the system is a result of the translation process, dependent on the language model, the way in which the sentences are segmented into phrases, the phrase-to-phrase translation and the model for reordering of phrases. A different structure is required for a lattice that also encodes information about the way in which hypotheses may be aligned to the source sentence from which they were generated: we examine the structure of such a lattice here. Our goal is a lattice that encodes all possible




```
<s> thank you . </s>
<s> thank you very much . </s>
<s> i thank you . </s>
<s> thank you ! </s>
<s> our thanks . </s>
<s> thanks . </s>
<s> thank you very . </s>
<s> we offer our thanks . </s>
```

Figure 4.1: Example lattice produced by the TTM, and the translation hypotheses that are accepted by it, output as an n-best list.

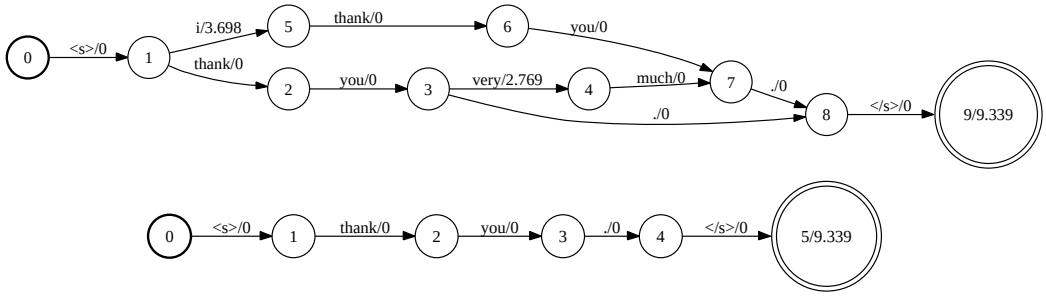


Figure 4.2: The effect of pruning on the lattice of hypotheses. Pruning thresholds of 4 and 0 respectively were used.

alignments of each hypothesis sentence with the input sentence, along with the probability of each alignment.

Let $\mathcal{L}$ denote the lattice of translation hypotheses for the input sentence $\mathbf{e} = e_1^I$. We build a new lattice $\mathcal{L}'$ to represent the alignment under a generative model of the translation hypotheses with the input sentence. We view the hypotheses as being generated from the input sentence (i.e. it is the source in our alignment model) and simultaneously align each hypothesis with the input. Due to the dependencies within the models, this is done differently for Model 1 and the HMM alignment model. We do not work directly with the probabilities,

since multiplication of a large number of small probabilities can cause numerical underflow. The FST framework (Allauzen et al., 2007) can deal effectively with costs on the arcs of the FST, and we work with negative log probabilities as the costs of the arcs. Finding a path with minimal cost is equivalent to finding a path with maximal probability, since

$$p > q \iff \log p > \log q \iff -\log p < -\log q. \tag{4.1}$$

Each arc x of the lattice has a number of attributes:

- x.start node from which the arc starts
- x.end node at which the arc ends
- x.cost cost of the arc
- x.input input to the arc (for a transducer)
- x.output symbol output by the arc, also referred to as its label

For simplicity of description, we assume that the outputs of the arcs are English words (although the procedures described work equally well for other languages) and that we are generating the target sentence $\mathbf{f}$ from source sentence $\mathbf{e}$. For abbreviation, write $x.\mathrm{f} = x.\mathrm{output}$ to stand for the English word on the arc and $x.\mathrm{a} = x.\mathrm{input}$ to stand for the alignment to the source sentence. Note that these algorithms all rely on the input lattice being determinised and having no ϵ-transitions: the FST toolkit can be used to ensure this before they are used.

4.3.1 Representation of Model 1 Alignment

This section describes the representation of all possible alignments under Model 1 of each sentence of a lattice of hypotheses with the source sentence e_1^I. For simple rescoring, we do not need to generate such a lattice for the reasons described in Section 4.6.1; however, this method allows us to represent all alignments.

Given the lattice of translation hypotheses $\mathcal{L}$, we create a new lattice $\mathcal{L}_{M1}$ that encodes alignments of each hypothesis with the source sentence e_1^I. We begin by cloning the nodes of $\mathcal{L}$: the new lattice will have exactly the same nodes but more arcs to keep track of all possible alignments. For each arc x of $\mathcal{L}$, we create $I+1$ arcs in $\mathcal{L}_{M1}$; label these $x^{(0)}, \ldots, x^{(I)}$, with arc $x^{(i)}$ corresponding to the alignment of the word on the arc with the i^{th} word of the source sentence. For each $x^{(i)}$, write $\bar{x}^{(i)}$ for the arc of the original lattice from which it is derived, i.e. $\bar{x}^{(i)} = x$. The input to the arc is the state from which the target word is emitted, and the output of the arc is that target word. Figure 4.3 shows how each arc is replaced by $I+1$ arcs which encode the alignments to form the new lattice. This lattice is an FST that represents all possible alignments of each of the translation hypotheses with the source sentence.

For each hypothesis sentence f_1^J and alignment a_1^J of this hypothesis with the source sentence e_1^I, there is a path x_1^J through the FST representing that sentence and alignment, with

$$x_j.\mathrm{output} = x_j.\mathrm{f} = f_j \tag{4.2}$$

$$x_j.\mathrm{input} = x_j.\mathrm{a} = a_j \tag{4.3}$$

$$x_j.\mathrm{cost} = -\log\left[\frac{1}{I+1}t(f_j|e_{a_j})\right] \tag{4.4}$$

Conversely, each path through the FST represents a hypothesis sentence and an alignment of that sentence to the source sentence. The FST takes an alignment sequence as input and

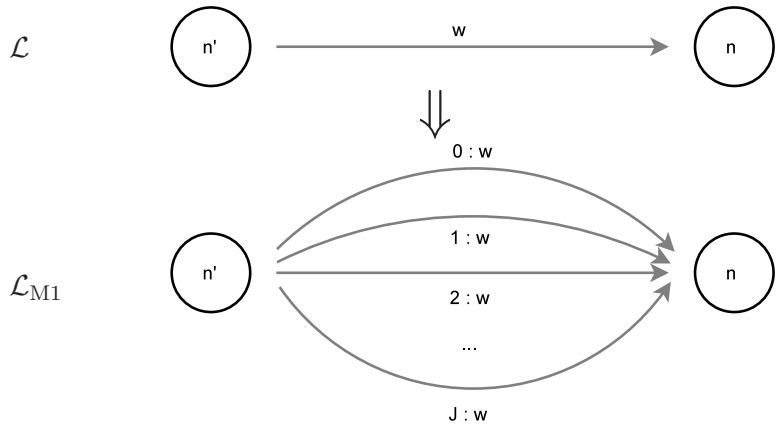


Figure 4.3: Diagram showing how the lattice is modified to encode Model 1 alignment information

outputs possible target sentences, with the cost of a target sentence being the negative log probability of that sentence with the given alignment sequence.

4.3.1.1 Cost of a partial path

Having encoded information about all possible alignments of each hypothesis in the lattice, we turn our attention to considering the probabilities associated with the alignments. For each partial path through the lattice $\mathcal{L}_{\mathrm{M1}}$, along arcs $x_1, x_2, \ldots, x_j$, define

$$\psi(x_1, x_2, \ldots, x_j) = \prod_{k=1}^{j} \frac{1}{I+1} t(x_k.\mathrm{f}|f_{x_k.\mathrm{a}}). \tag{4.5}$$

This is the probability of generating the target word sequence $x_1.\mathrm{f}, \ldots, x_k.\mathrm{f}$ with alignments $x_1.\mathrm{a}, \ldots, x_k.\mathrm{a}$ from e_1^I under Model 1. There will be multiple paths to each node corresponding to differing hypotheses and alignments of those hypotheses, but one of those paths will have a higher probability than the others. For a node n, write $\psi^*(n)$ for the probability of the best partial path (i.e. the one with the highest probability) to that node. Due to the independence assumptions of the model, this can be calculated by examining all arcs x leading to n and the nodes from which they start:

$$\begin{aligned} \psi^*(n) &= \max_{\text{paths } x_1, \ldots, x \text{ to } n} \psi(x_1, \ldots, x) \\ &= \max_{\text{arcs } x \to n} \left[\psi^*(x.\mathrm{start}) \frac{1}{J+1} t(x.\mathrm{f}|e_{x.\mathrm{a}}) \right] \end{aligned} \tag{4.6}$$

$\psi^*(n)$ can be computed iteratively, starting with the first node in the lattice. The best alignment has probability $\psi^*(N)$, where N is the final node of the lattice. Note that in the case that lattice has more than one final state, the best alignment has probability $\max_{\text{final nodes } n} \psi^*(n)$.

The cost of each partial path is given by

$$\begin{aligned} \phi(x_1, x_2, \ldots, x_j) &= -\log \psi(x_1, x_2, \ldots, x_j) \\ &= -\sum_{k=1}^{j} \log \left[\frac{1}{I+1} t(x_k.\mathrm{f}|e_{x_k.\mathrm{a}}) \right]. \end{aligned} \tag{4.7}$$

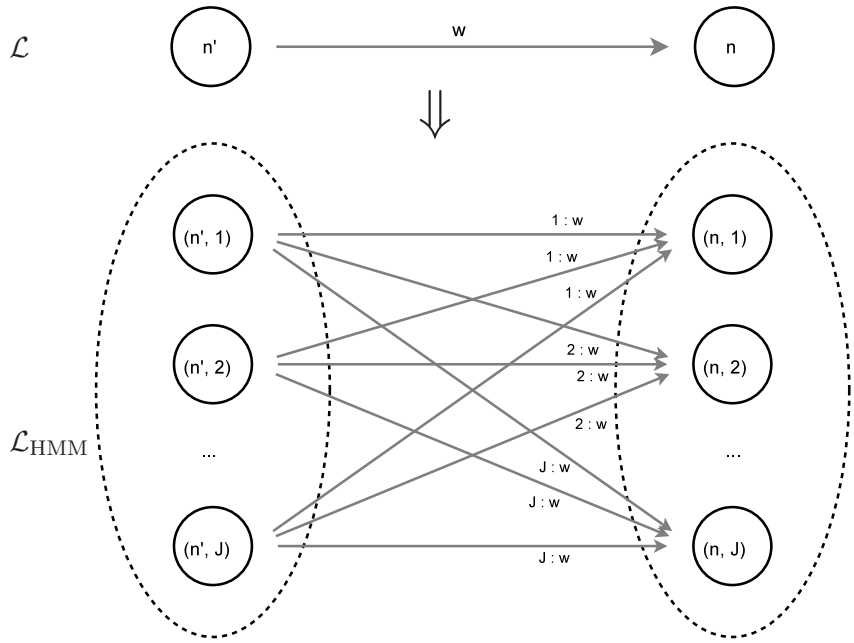


Figure 4.4: Diagram showing how the lattice is modified to encode HMM alignment information. Each node of $\mathcal{L}$ is replaced by J nodes in $\mathcal{L}'$. Where nodes n and n' are connected in $\mathcal{L}$, (n, i) is connected in $\mathcal{L}'$ to (n', i') for all $i', i \in \{1, \ldots, I\}$.

4.3.2 Representation of HMM Alignment

In order to represent alignments between the sentences generated by an HMM, we need to take into account the additional dependencies of the model, namely that the alignment of a target word to the source sentence depends on the alignment of the previous word. We form a new lattice $\mathcal{L}_{\text{HMM}}$ by replacing each node n of $\mathcal{L}$ with I nodes $(n, 1), \ldots, (n, I)$. For each pair of nodes $n', n \in \mathcal{L}$ and each arc x from n' to n and each pair (i', i) with $1 \leq i, i' \leq I$, create arcs $x^{(i',i)}$ from (n', i') to (n, i) with

$$\begin{aligned} x^{(i',i)}.\text{input} &= i \\ x^{(i',i)}.\text{output} &= x.\text{output} \\ x^{(i',i)}.\text{cost} &= -\log[a(i|i', I)t(f|e_i)] \end{aligned}$$

The state from which the previous word was emitted affects the transition probability and we need the lattice to encode information to keep track of the previous state. All the arcs leading to each node have the same input; all arcs leading to node (n, i) have $x.\text{input} = x.\text{a} = i$. For convenience, write $(n, i).\text{a}$ for the common alignment of all arcs leading to (n, i). Following an arc x with output f from (n', i') to (n, i) is equivalent to moving from state i' to state i and generating English word f from foreign word e_i.

Again, every possible hypothesis sentence and alignment with the source sentence is represented by a path through the lattice, and conversely each path represents a hypothesis

sentence and alignment. A path $(n_0, j_0), (n_1, i_1), \ldots, (n_I, i_I)$ through the lattice $\mathcal{L}_{\text{HMM}}$ via arcs $x_1, \ldots, x_J$ corresponds to the alignment a_1^J of the sentence f_1^J with e_1^I with

$$\begin{aligned} x_j.\text{output} &= f_j && (4.8)\\ x_j.\text{input} &= a_j && (4.9)\\ x_j.\text{start.a} &= a_{j-1} && (4.10)\\ x_j.\text{cost} &= -\log\left[a(a_j|a_{j-1}, I)t(f_j|e_{a_j})\right] && (4.11) \end{aligned}$$

for $j = 1, \ldots, J$.

4.3.2.1 Cost of a partial path

For a partial path through the lattice $\mathcal{L}_{\text{HMM}}$ passing through nodes $(n_0, i_0), (n_1, i_1), \ldots, (n_j, i_j)$ of the lattice via arcs $x_1, \ldots, x_j$, define

$$\begin{aligned} \psi(x_1, x_2, \ldots, x_j) &= \prod_{k=1}^{j} a(i_k|i_{k-1}, I)t(e_k|e_{i_k}) && (4.12)\\ &= \prod_{k=1}^{j} a(x_k.\text{a}|x_{k-1}.\text{a}, I)t(x_k.\text{f}|e_{x_k.\text{a}}) && (4.13) \end{aligned}$$

For a node (n, i), write $\psi^*(n, i)$ for the probability of the best partial path through the lattice to that node. This is given by taking the maximum probability over all paths, and $\psi^*(n, i)$ is computed iteratively starting from the start node in the lattice.

$$\begin{aligned} \psi^*(n, i) &= \max_{\text{paths } x_1, \ldots, x \text{ to } (n,i)} \psi(x_1, x_2, \ldots, x)\\ &= \max_{\text{arcs } x \text{ to } (n,i)} \psi^*(x.\text{start})a(i|x.\text{start.a}, I)t(x.\text{f}|e_i), \end{aligned}$$

where x.start.a is the common alignment of all arcs leading to x.start. Again, we define the cost of a partial path by

$$\begin{aligned} \phi(x_1, x_2, \ldots, x_j) &= -\log \psi(x_1, x_2, \ldots, x_j)\\ &= -\sum_{k=1}^{j} \log\left[a(x_k.\text{a}|x_{k-1}.\text{a}, I)t(x_k.\text{f}|e_{x_k.\text{a}})\right]. && (4.14) \end{aligned}$$

4.4 Rescoring a single hypothesis and N-best rescoring

Before looking at rescoring the full lattice of translation hypotheses, we look at rescoring a single sentence $\mathbf{f} = f_1^J$ that is hypothesised by the system as a translation of $\mathbf{e} = e_1^I$. The original motivation, using the source-channel model and the TTM translation system, was to improve the translation quality by replacing the simple translation probability estimated by the TTM with a more accurate one from the alignment model. Consider the TTM decision rule:

$$\hat{\mathbf{e}} = \operatorname*{argmax}_{\mathbf{e}} \text{P}_{\text{LM}}(\mathbf{e})\text{P}_{\text{TTM}}(\mathbf{f}|\mathbf{e}), \qquad (4.15)$$

where $\mathrm{P}_{\mathrm{LM}}(\mathbf{e})$ is the language model probability and $\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e}$ is the probability of generating $\mathbf{f}$ from $\mathbf{e}$ under the TTM model. We replace $\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e})$ by $\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e})$ and use the same language model $\mathrm{P}_{\mathrm{LM}}(\mathbf{e})$ used by the TTM, so the decision rule becomes

$$\hat{\mathbf{e}} = \operatorname*{argmax}_{\mathbf{e}} \mathrm{P}_{\mathrm{LM}}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e}). \tag{4.16}$$

However, this fails to make use of the information in the TTM translation model, so we turn to the log-linear framework. Here, we can incorporate the alignment model scores by viewing them as additional features in the log-linear model. We use the decision rule

$$\begin{aligned} \hat{\mathbf{e}} &= \operatorname*{argmax}_{\mathbf{e}} \mathrm{P}(\mathbf{e}|\mathbf{f}) \\ &= \operatorname*{argmax}_{\mathbf{e}} \sum_{m=1}^{M'} \lambda_m h_m(\mathbf{e}, \mathbf{f}) \end{aligned} \tag{4.17}$$

where the first M features from the existing translation model that generated the sentence and $M' - M$ features are added during rescoring.

Features we can use include sentence-to-sentence translation probabilities under the alignment models in Chapter 3, i.e. Model 1, Model 2, word-to-word HMM, word-to-phrase HMM with varying maximum phrase lengths. We can use the models in both translation directions, i.e. we can use those which define the probability $\mathrm{P}(\mathbf{f}|\mathbf{e})$ of generating $\mathbf{f}$ from $\mathbf{e}$ and those which define the probability $\mathrm{P}(\mathbf{e}|\mathbf{f})$ of generating $\mathbf{e}$ from $\mathbf{f}$.

These can all be used in N-best list rescoring but some are more suitable for use in lattice rescoring due to the way in which the hypotheses are represented. We look at probabilities here: to use a probability distribution P as a feature function, simply take the value of the feature associated with that probability distribution to be $h_{\mathrm{P}} = -\log \mathrm{P}$. Sometimes we may wish to use the Viterbi approximation to the full translation probability for speed reasons, i.e. rather than using $h = -\log \mathrm{P}(\mathbf{e}|\mathbf{f})$ we use $h_{\mathrm{Viterbi}} = -\log \operatorname{argmax}_{\mathbf{a}} \mathrm{P}(\mathbf{e}, \mathbf{a}|\mathbf{f})$.

4.5 Rescoring lattices of hypotheses

We have so far demonstrated a method for encoding the translation probabilities under the Model 1 and word-to-word HMM alignment models of all possible alignments of each hypothesised sentence in the lattice. We wish to combine this information with the costs already present in the lattices of hypotheses, which represent probabilities from the language model, phrase segmentation, phrase-to-phrase translation and other components of the translation process. In the log linear framework, we can view this as adding the Model 1 or HMM probability as an additional feature.

Rescoring can be used as a final step in translation, taking the least-cost path through the lattice to give the 1-best hypothesis. However, we can also output a lattice representing all alignments of each English translation hypothesis with the foreign sentence. This allows lattice rescoring to be performed as an intermediate step, and the new lattice can be passed on to a different rescoring procedure.

4.5.1 Using existing lattice costs

For rescoring a lattice with an alignment model, we construct lattices by the method demonstrated in Section 4.3. Rather than discarding the cost assigned by the translation system, we add it to the cost assigned by the alignment model, with a suitable weight λ determined by simple tuning or by discriminative training.

For Model 1, consider an arc x in the lattice $\mathcal{L}_{\mathrm{M1}}$. Let $c_{\mathcal{L}}(\bar{x})$ be the cost in $\mathcal{L}$ of the arc from which the arc x of $\mathcal{L}_{\mathrm{M1}}$ is derived, and λ be the feature weight given to the alignment model's cost. Then the cost of x is defined to be

$$c_{\mathrm{M1}}(x) = c_{\mathcal{L}}(\bar{x}) + \lambda \times -\log\left[\frac{1}{J+1} t(x.\mathrm{e}|f_{x.\mathrm{a}})\right]. \tag{4.18}$$

The cost of a partial path through the lattice via arcs $x_1, \ldots, x_j$ becomes

$$\phi(x_1, \ldots, x_j) \;=\; \sum_{k=1}^{j} \left(c_{\mathcal{L}}(\bar{x}_k) - \lambda \log\left[\frac{1}{I+1} t(x_k.\mathrm{f}|f_{x_k.\mathrm{a}})\right]\right). \tag{4.19}$$

Define $\phi^*(n)$ as the cost of the best path to a node n. Then

$$\begin{aligned}
\phi^*(n) &= \min_{\text{paths } x_1,\ldots,x \text{ to } n} \phi(x_1, \ldots, x) \\
&= \min_{n' \to n} \min_{x:\, n' \to n} \phi^*(n') + c_{\mathcal{L}_{\mathrm{M1}}}(x) \\
&= \min_{n' \to n} \min_{x:\, n' \to n} \phi^*(n') + c_{\mathcal{L}}(\bar{x}) - \lambda \log\left[\frac{1}{J+1} t(x.\mathrm{f}|e_{x.\mathrm{a}})\right].
\end{aligned}$$

Similarly for the HMM, we define the cost of an arc x from (n', i') to (n, i) in $\mathcal{L}_{\mathrm{HMM}}$ to be

$$\begin{aligned}
c_{\mathrm{HMM}}(x) &= c_{\mathcal{L}}(\bar{x}) + \lambda \times -\log\left[a(x.\mathrm{a}|x.\mathrm{start.a}, J) t(x.\mathrm{f}|e_{x.\mathrm{a}})\right] \\
&= c_{\mathcal{L}}(\bar{x}) + \lambda \times -\log\left[a(i|i', I) t(x.\mathrm{f}|e_i)\right]
\end{aligned}$$

The cost of a partial path through the lattice $\mathcal{L}_{\mathrm{HMM}}$ via arcs $x_1, \ldots, x_j$ is

$$\phi(x_1, \ldots, x_j) \;=\; \sum_{k=1}^{j} \left\{ c_{\mathcal{L}}(\bar{x}_k) - \lambda \log\left[a(x_k.\mathrm{a}|x_{k-1}.\mathrm{a}, I) t(x_k.\mathrm{f}|e_{x_k.\mathrm{a}})\right]\right\} \tag{4.20}$$

The cost of the best path to node (n, i) is given iteratively by

$$\begin{aligned}
\phi^*(n,i) &= \min_{\text{paths } x_1,\ldots,x \text{ to } (n,i)} \phi(x_1, \ldots, x) \\
&= \min_{(n',i') \to (n,i)} \min_{x:\, (n',i') \to (n,i)} \left[\phi^*(n', i') + c_{\mathcal{L}'}(x)\right] \\
&= \min_{(n',i') \to (n,i)} \min_{x:\, (n',i') \to (n,i)} \left[\phi^*(n', i') + c_{\mathcal{L}}(\bar{x}) - \lambda\{\log a(i|i', I) t(x.\mathrm{f}|e_i)\}\right]
\end{aligned}$$

4.5.2 Output and pruning of the lattice

The addition of word-to-word alignment information can significantly increase the size of the lattice, since it is possible for each target word in each hypothesis sentence to align to each source word. For a source sentence of length I, the new lattice encoding Model 1 alignment will have the same number of nodes but $I+1$ times as many arcs. For HMM alignment, the new lattice will have I times as many nodes and I^2 times as many arcs. With some sentences reaching 100 words in length, this can be a significant increase in the size of the lattice. We can therefore apply pruning to the lattice as the algorithm is running, to keep the output to a manageable size. A beam search (Aubert, 2002) used is to output at each node a lattice containing all the paths whose costs are within a given threshold T of the best (partial) path to that node. We remove nodes and arcs that are not on a path whose cost is within the threshold.

4.5.2.1 Lattice alignment for Model 1

Define $B(n)$ to be the set of arcs to n that appear on a path whose cost is within T of the best path to n. Then

$$B(n) \quad = \quad \left\{ \begin{array}{ll} \text{arcs } x \text{ to } n: & \phi^*(x.\text{start}) + c_{\mathcal{L}}(\bar{x}) - \lambda \log \left[\frac{1}{I+1} t(x.\text{f}|e_{x.\text{a}}) \right] \\ & < \phi^*(n) + T \end{array} \right\},$$

i.e. we consider each preceding arc, and include only those arcs that can extend a path that keeps the cost within threshold T of the best path to n. We output the lattice containing arcs

$$\bigcup_{n \in \mathcal{L}_{\text{M1}}} B(n).$$

The algorithm used is as follows:

Input: Lattice $\mathcal{L}$, topologically sorted with N nodes
Output: Pruned lattice containing rescored hypotheses
foreach $n = 1, \ldots, N$ **do**
  Calculate $\phi^*(n)$;
  foreach $n' \to n$ **do**
    foreach *arc* x *from* n' *to* n **do**
      foreach $i = 0, 1, \ldots, I$ **do**
        $c = c_{\mathcal{L}}(\bar{x}) - \lambda \log(\frac{1}{I+1} t(x.\text{f}|e_i))$;
        if $\phi^*(n') + c < \phi^*(n) + T$ **then**
          Output arc $n' \to n$ with cost c and input i;
        end
      end
    end
  end
end

(Note $e_0 =$ `NULL`)

4.5.2.2 Lattice alignment algorithm for the HMM model

For the HMM model, the algorithm changes to take account of the fact that the dependencies of the model are different:

Define $B(n,i)$ to be the set of arcs to (n,i) that appear on a path whose cost is within T of the best path to (n,i). Then

$$B(n,i) = \left\{ \begin{array}{c} \text{arcs } x \text{ to } (n,i) : (n',i') = x.\text{start and} \\ \phi^*(n',i') + c_{\mathcal{L}}(\bar{x}) - \lambda \log\left[a(i|i',J)t(x.\text{f}|e_i)\right] < \phi^*(n,i) + T \end{array} \right\},$$

i.e. we consider each preceding arc, and include only those arcs that can extend a path that keeps the cost within threshold T of the best path to (n,i). We output the lattice containing the following arcs:

$$\bigcup_{(n,i) \in \mathcal{L}_{\text{HMM}}} B(n,i).$$

Input: Lattice $\mathcal{L}$, topologically sorted with N nodes
Output: Pruned lattice containing rescored hypotheses
foreach $n = 1, \ldots, N$ **do**
  foreach $i = 1, \ldots, I$ **do**
    Calculate $\phi^*(n,i)$;
    foreach $n' \rightarrow n$ **do**
      foreach $i' = 1, \ldots, I$ **do**
        foreach *arc* x *from* n' *to* n **do**
          $c = c_{\mathcal{L}}(\bar{x}) - \lambda \log\left[a(i|i',I)t(x.\text{f}|e_i)\right]$;
          if $\phi^*(n',i') + c < \phi^*(n,i) + T$ **then**
            Output arc from (n',i') to (n,i) with score c;
          end
        end
      end
    end
  end
end

4.5.3 Discussion of pruning method

This method of pruning has the advantage that it is guaranteed to include all paths within the required threshold of the best path. Consider any node n, and any arc x to n that is not on a path whose cost is within T of the best path. Then x is not in any path whose cost is within T of the best path through n. Therefore x is not in any path through the lattice whose cost is within T of the best path. It can however lead to some nodes that have no path to any final state: these should be removed.

The algorithm is greedy, in that at each point it makes the locally optimum decision but does not consider the global optimum. A disadvantage of this is that partial paths through the lattice can be extended even if there is no path through the eventual lattice containing them. One way of overcoming this is to compute, for each node, the cost of the best partial path to a final state; then we can prune by considering the cost of the remainder of the partial path.

Model	Foreign to English probability	English to Foreign probability
Model 1	$\mathrm{P}(e_1^I, a_1^I \mid f_1^J) = \frac{1}{(J+1)^I} \prod_{i=1}^{I} t(e_i \mid f_{a_i})$	$\mathrm{P}(f_1^J, a_1^J \mid e_1^I) = \frac{1}{(I+1)^J} \prod_{j=1}^{J} t(f_j \mid e_{a_j})$
Model 2	$\mathrm{P}(e_1^I, a_1^I \mid f_1^J) = \prod_{i=1}^{I} a(a_i \mid i, I, J) t(e_i \mid f_{a_i})$	$\mathrm{P}(f_1^J, a_1^J \mid e_1^I) = \prod_{j=1}^{J} a(a_j \mid j, I, J) t(f_j \mid e_{a_j})$
HMM	$\mathrm{P}(e_1^I, a_1^I \mid f_1^J) = \prod_{i=1}^{I} a(a_i \mid a_{i-1}, J) t(e_i \mid f_{a_i})$	$\mathrm{P}(f_1^J, a_1^J \mid e_1^I) = \prod_{j=1}^{J} a(a_j \mid a_{j-1}, I) t(f_j \mid e_{a_j})$

Table 4.1: Probabilities of translation in both directions under Model 1, Model 2 and word-to-word HMM

4.6 Features for lattice rescoring

The algorithms of Section 4.5 show how lattices can be created to encode alignment with the source sentence and include Model 1 and HMM alignment model costs in addition to the costs assigned by the translation system whose output we are rescoring. This section looks at these costs and investigates further alignment model features that can be used, discussing their suitability for use in lattice-based rescoring. Table 4.1 shows the sentence-to-sentence probabilities under various alignment models in both translation directions; we examine this to determine which components of the models are best used as features for rescoring.

The total contribution to the cost of the sentence from the Model 1 alignment probability in the foreign to English direction is

$$\lambda \times -\log \frac{1}{J+1} = \lambda \log(J+1). \tag{4.21}$$

This is, in effect, applying a word insertion penalty dependent on the length J of the foreign sentence, i.e. the longer the foreign sentence, the larger the penalty applied to each word of the target sentence. This is counter-intuitive as we expect long sentences to generate long sentences. We do not estimate the Model 1 sentence length distribution $\mathrm{P}(I|J)$ as it is not needed for alignment, though this sentence length distribution would help balance this effect if it were included. One option is to give this part of the Model 1 alignment a separate weight that can be tuned separately, in particular the weight could be negative to encourage more words in the hypothesis for longer source sentences. We choose to omit it altogether and rely on a sentence length distribution as another feature.

Each arc in the lattice may occur on more than one path and some of the paths which include a particular arc may differ in length; therefore it is difficult to use models that depend on the hypothesis length I. This means that we have difficulty using the alignment component $\frac{1}{I+1}$ for the English to foreign Model 1, but this is unimportant since it would effectively be a length penalty. Similarly we do not use the Model 2 alignment probabilities in either direction, $a(a_i|i, I, J)$ and $a(a_j|j, I, J)$, as features. Using the English to foreign HMM alignment probability $a(a_j|a_{j-1}, I)$ is not possible. It is, however, possible to use the foreign to English HMM alignment probability $a(a_i|a_{i-1}, J)$, as well as the HMM word-to-word translation probabilities $t(f|e)$ and $t(e|f)$ and Model 2 word-to-word translation probabilities.

We add cost $-\log t(f_{x.\mathrm{a}}|x.\mathrm{e})$ on each arc, as another feature of the log-linear model. We can then include this in the calculation of the cost of the arcs and the computation of ϕ^* at each node.

4.6.1 Lattice analogue to Viterbi alignment and full translation probability

To find the probability $\mathrm{P}(e_1^I|f_1^J)$, we marginalise over all alignments a_1^I, i.e. we find

$$\mathrm{P}(e_1^I|f_1^J) = \sum_{a_1^I} \mathrm{P}(e_1^I, a_1^I|f_J). \tag{4.22}$$

In the lattice, this can be obtained by taking the sum over all paths *in the log semiring*, i.e.

$$\begin{aligned} -\log \mathrm{P}(e_1^I|f_1^J) &= -\log \sum_{a_1^I} \mathrm{P}(e_1^I, a_1^I|f_J) \\ &= \bigoplus_{a_1^I} -\log \mathrm{P}(e_1^I, a_1^I|f_1^J) \end{aligned} \tag{4.23}$$

where $x \oplus y = -\log(e^{-x} + e^{-y})$. In the tropical semiring, which is often used for machine translation, $x \oplus y = \min(x, y)$; hence taking the sum over all paths gives the cost of the path with minimal cost, i.e. we use the probability of the Viterbi alignment (Equation (3.4)) as an approximation to the true marginal probability:

$$\mathrm{P}(e_1^I|f_1^J) \approx \max_{a_1^I} \mathrm{P}(e_1^I, a_1^I|f_1^J) \tag{4.24}$$

By projecting the lattice onto its outputs, we remove alignment information, leaving a finite state acceptor that accepts a translation hypothesis and assigns a cost. The cost depends on the semiring used by the acceptor: if we use the log semiring, the cost represents the sum over all possible alignments; in the tropical semiring, the cost is that of the Viterbi alignment.

For Model 1 rescoring, determinisation leads to a lattice with exactly the same nodes and arcs as the input, with the costs of the arcs modified to reflect rescoring. For the full alignment $\sum_{j=0}^{J} t(x.\mathrm{e}|f_j)$ is added to the cost of arc x; for Viterbi alignment $\max_j t(x.\mathrm{e}|f_j)$ is added. If we have no need for the alignment information, we do not need to output the complete extended lattice. Instead, we can simply modify the weights on the input lattice adding the Model 1 translation cost for each arc. In this way we output a lattice that is identical in size to the starting lattice, but has the scores on the arc updated to include the Model 1 translation cost. Figure 4.5 shows a lattice before alignment, the same lattice with all possible alignments shown, and the lattice again once it has been determinised in the log semiring.

For rescoring large lattices with HMMs, it may not be practical to use the full alignment probability, and we use the cost of Viterbi alignment to approximate the cost. Using the $\oplus$ operation in the tropical semiring is much more efficient, since it simply takes the minimal cost of the path. Lattice generation can be modified to include only those arcs that will appear on a path representing the Viterbi alignment of a hypothesis sentence, since any other arcs will be discarded.

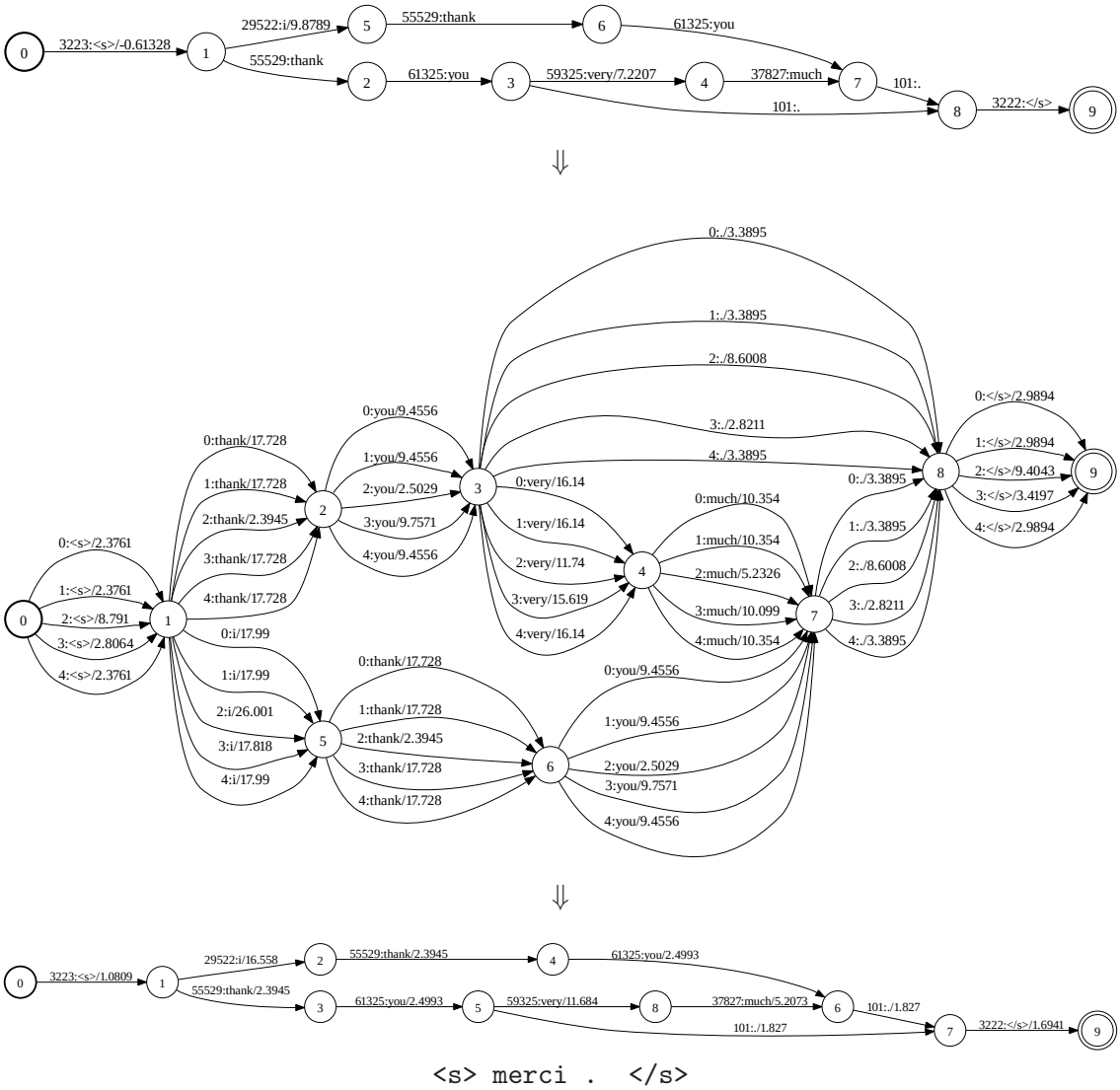


Figure 4.5: The lattice alignment and rescoring process for Model 1: the original lattice (top) is aligned with the source sentence to give all possible alignments of the original lattice and their costs (middle); the result is determinised to give a lattice showing the total translation cost under Model 1 (bottom).

4.7 Summary

This chapter introduces lattice-to-string alignment algorithms for the alignment of lattices of target sentences to the source sentence, for IBM Model 1 and the word-to-word HMM. We demonstrate an algorithm for creating another lattice that encodes the alignment and probability of each alignment for every hypothesis sentence. This lattice can be used for rescoring translation hypotheses using the alignment model. The method of rescoring is determined by the semiring over which we build the lattice, so rescoring can be applied using the Viterbi alignment or the full marginal alignment probability.

Chapter 5
Results of Lattice Rescoring Experiments

5.1 Introduction

Chapter 4 introduces an algorithm for simultaneous alignment of all translation hypotheses in a lattice with a source sentence, and discusses the use of alignment models for the rescoring of lattices of translation hypotheses. This chapter describes experiments carried out to investigate the performance of the different methods of lattice rescoring. Results are presented for the translation of Arabic, Chinese, French and Spanish into English.

Initial experiments were carried out using MTTK alignment models to rescore N-best lists produced by the TTM translation system, before further experiments were carried out using lattice-to-string alignment and lattice rescoring. Combinations of the rescoring schemes described in Section 4.6 were investigated. Investigation was carried out into the effects of pruning the lattice at a variety of thresholds to remove low scoring hypotheses prior to rescoring.

These experiments were carried out using a baseline system that did not use lexical weighting between the phrases in a phrase pair as a feature during translation. Since this alignment model rescoring also makes use of lexical knowledge, further experiments were carried out to determine whether the two methods are complementary, i.e. to determine whether rescoring with alignment models improved over a baseline that already includes lexical information.

5.2 Experimental procedure and data sets

5.2.1 Model training

This section describes how the MTTK alignment models are trained, and their extension to cover unseen words in the test data, as well as the training of the TTM system. Details specific to each experiment are detailed in their respective sections.

MTTK was used to train IBM Model 1 and 2 alignment models followed by a word-to-word HMM, according to the standard MTTK training procedure described in Section 3.12. The models were used to produce alignments of the training data and construct a phrase pair inventory and the associated phrase translation table, as described in Section 3.10. The TTM models were built from these systems, using the MJ-1 reordering model. For some experiments, probability distributions within the TTM were used as features in a log-linear model, as described in Section 2.5.1. In this case, the feature weights were optimised using MERT on a held-out data set.

A word insertion penalty was applied to each word to control the balance between short sentences that do not convey the required information and inserting superfluous words into a translation. The insertion penalty was optimised for the baseline TTM model, but the optimum insertion penalty can vary as the method of rescoring changes; hence a range of values was investigated during rescoring. In the results given, the word insertion penalty is multiplied by the number of words in the sentence and added to the cost; therefore a smaller (i.e. more negative) insertion penalty reduces the cost per word and favours longer sentences.

The various rescoring schemes were evaluated by BLEU score. The 1-best translation hypothesis under each scheme was output, and the translations of the entire test set were evaluated against reference translations.

5.2.2 Issues in alignment model rescoring

The test data contain words that are not in the alignment model vocabulary defined by the training data, and we modify the models to account for unseen words. We have a separate alignment model for each translation direction, i.e. from English to foreign (where English is viewed as the source language) and foreign to English, and both are used for rescoring, so we extend both models in the same way. Firstly, the source and target vocabularies $\mathcal{V}_{\text{src}}, \mathcal{V}_{\text{tgt}}$ of the model are extended to include all the words in the test data, then we add entries to the translation table for the new words. For out-of-vocabulary source words $e \in \mathcal{V}_{\text{src}}$, we assume a uniform distribution of target words, i.e. $t(f|e) = \frac{1}{|\mathcal{V}_{\text{tgt}}|+1}$ for all $f \in \mathcal{V}_{\text{tgt}}$, where $\mathcal{V}_{\text{tgt}}$ is the target language vocabulary, expanded to include the new words. For out-of-vocabulary target words f, we assign a minimum probability $t(f|e) = t_{\min}$ for all $e \in \mathcal{V}_{\text{src}}$.

If the hypothesis space is large, we may wish to apply pruning prior to rescoring the lattice. Pruning reduces the size of the search space but may result in the removal of hypotheses that are assigned a high cost by the original model but a much lower cost by the rescoring process, resulting in search errors. The aim is to balance the size of the search space against the confidence of the TTM hypotheses: if we are confident that the original TTM system will rank good sentences highly, we can prune more aggressively without the risk of losing those sentences. We investigate the effect of varying the threshold when pruning prior to translation.

When the HMM alignment model is trained, we need to estimate $a(i|i', I)$ for all values of I and all values of $1 \leq i, i' \leq I$. Due to data sparsity issues (there are very few long sentences

in training data and it would be hard to estimate the probabilities accurately), we apply an upper limit to the length of the source sentence when training the model. For rescoring a lattice of possible translations using an HMM alignment model to align the translation hypotheses to the input sentence, the length of the input sentence sometimes exceeds this limit; in this case, we back off to the costs in the original lattice produced by the translation system and the 1-best translation hypothesis for that sentence is unchanged.

5.2.3 Evaluation data sets

The rescoring algorithms were evaluated on systems trained for the translation of four different languages into English. The data sets used for training and evaluation are as follows:

- **NIST Arabic to English translation:** The alignment models were trained on the News subset of the NIST 2005 large data track Arabic-English parallel text. The TTM models were built using phrases extracted from the complete NIST Arabic-English parallel text, which was aligned using the MTTK models above. The system was tested on the NIST *eval03* and *eval04* data sets, which contain 663 and 1353 sentences respectively. The language model used was the Cambridge University Engineering Department/Johns Hopkins University 2005 4-gram language model, which is an interpolated Kneser-Ney 4-gram language model trained on the AFP and Xinhua subsets of the LDC Gigaword corpus (Graff et al., 2005) and the English side of the parallel text.

- **NIST Chinese to English translation:** Alignment models were trained on the parallel text allowed for the NIST 2005 Chinese-English evaluation, and used to produce alignments on this parallel text. A TTM system was built on these alignments and tests were carried out using the *eval03* data set for this evaluation, which contains 919 sentences. As for Arabic, the language model was trained on the AFP and Xinhua subsets of the LDC Gigaword corpus and the English side of the parallel text.

- **Europarl French to English translation:** The Europarl corpus is a corpus of parallel text in 11 languages collected from the proceedings of the European Parliament. At the time of these experiments, the corpus contained about 30 million words in each of the official languages of the European Union (Koehn, 2005).

  The Europarl French and English parallel texts allowed for the North American Association of Computational Linguistics (NAACL) 2006 Workshop on Machine Translation shared task on exploiting parallel texts for statistical machine translation (Koehn and Monz, 2006) were used to train models. The data used for rescoring experiments was the 2006 development test set of the shared task. This is a collection of 2000 sentences taken from the Europarl corpus but not included in the training data. The language model used was a trigram language model provided with the collection.

- **TC-STAR Spanish to English translation:** Alignment models were trained on the parallel text allowed for the TC-STAR 2006 evaluation, and a TTM system was initialised from the resulting word-aligned parallel text. Model 1 was used to rescore lattices of translation hypotheses produced for the TC-STAR test set, which contains 1452 sentences. The language model used was a 4-gram model with Kneser-Ney smoothing trained on the English portion of the parallel text.

See Table A.1 for a summary of the parallel text used for training.

5.3 N-best list rescoring with alignment models

We begin our investigation of alignment model re-ranking by using alignment models for rescoring N-best lists of hypotheses. For each sentence to be translated, the TTM was used to produce a lattice of English translation hypotheses, and the best scoring 1000 hypotheses were extracted. These hypotheses were then aligned with the corresponding foreign sentence and the costs recalculated before the best hypothesis for each source sentence was placed in a translated data set to be compared against the references. Results are compared against a baseline formed by taking the 1-best hypothesis from the original TTM lattice for each sentence in the test set.

In the following sections, we use the notation E-F to denote models where the foreign sentence is viewed as being generated from the English sentence and F-E to denote models where the English sentence is generated from the foreign sentence. In all cases we are translating into English; hence English is the *output language* and the foreign sentences are in the *input language*.

5.3.1 Rescoring of Arabic-English translation

The baseline system uses the decision rule described in Chapter 2:

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \, \mathrm{P}_{\mathrm{LM}}(\mathbf{e})\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e}), \tag{5.1}$$

where $\mathrm{P}_{\mathrm{LM}}(\mathbf{e})$ is the language model probability of output sentence $\mathbf{e}$ and $\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e})$ is the probability assigned by the TTM system to generating input sentence $\mathbf{f}$ from output sentence $\mathbf{e}$. We investigated a number of different methods for combining costs for rescoring of the translation hypotheses. Firstly, the TTM translation score was replaced by the MTTK translation score, i.e. the probability of sentence $\mathbf{e}$ is then $\mathrm{P}_{\mathrm{LM}}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e})$, where $\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e})$ is the probability of generating $\mathbf{f}$ from $\mathbf{e}$ under the MTTK alignment model (this is equivalent to using a log-linear model with features $\mathrm{P}_{\mathrm{LM}}(\mathbf{e})$, $\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e})$ and uniform feature weights). All MTTK alignment models were tested: Model 1; Model 2; word-to-word HMM; word-to-phrase HMMs with maximum phrase lengths 2, 3, 4 and finally word-to-phrase model with maximum phrase length 4 and bigram translation table. Figures 5.1 and 5.2 show the results on the NIST Arabic-English *eval03* and *eval04* data sets respectively. Overall, we see that rescoring provides an improvement in translation quality over the baseline system for a range of word insertion penalties, with gains of at least 1 BLEU point for the best-performing models. The results for the more complex word-to-phrase HMMs are similar to, and follow the same patterns as, the word-to-word HMM, so they are omitted for simplicity.

This score making use of the alignment models in the English to Arabic direction cannot be calculated using the lattice rescoring algorithm developed in Section 4.5, due to the difficulty of dealing with each arc appearing of paths of differing length. An alternative is to replace the translation cost with that of the alignment model in the opposite direction, leading to decision rule $\hat{\mathbf{e}} = \operatorname{argmax}_{\mathbf{e}} \mathrm{P}_{\mathrm{LM}}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f})$. While the computation of $\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f})$ is no easier with N-best lists, the lattice alignments later rely on the computation in this direction providing hypotheses with improved translation quality. Figures 5.3 and 5.4 show the results on the NIST Arabic-English *eval03* and *eval04* data sets respectively. Again we see that

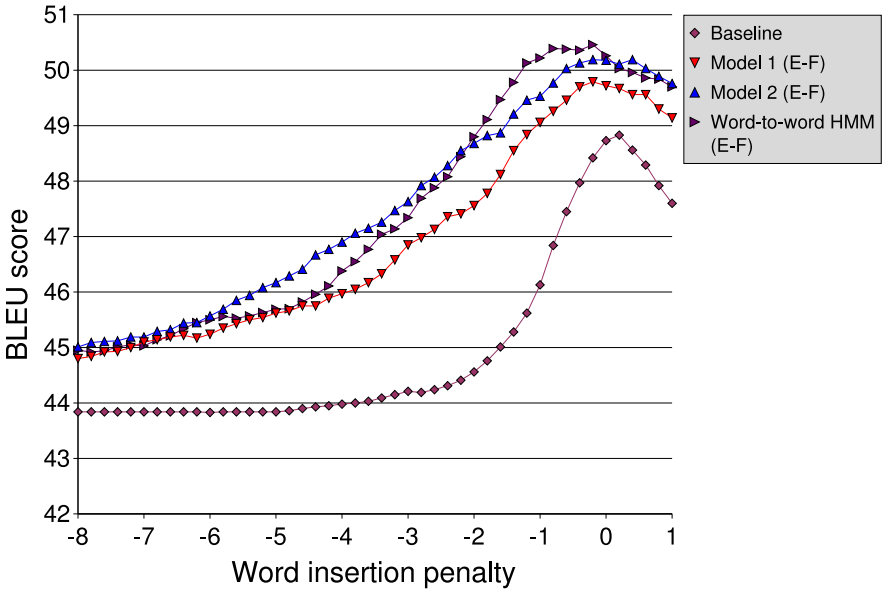


Figure 5.1: N-best rescoring for Arabic-English eval03 *data set, replacing translation model probability with Arabic-English alignment model probability. We maximise* $\mathrm{P}_{LM}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e})$.

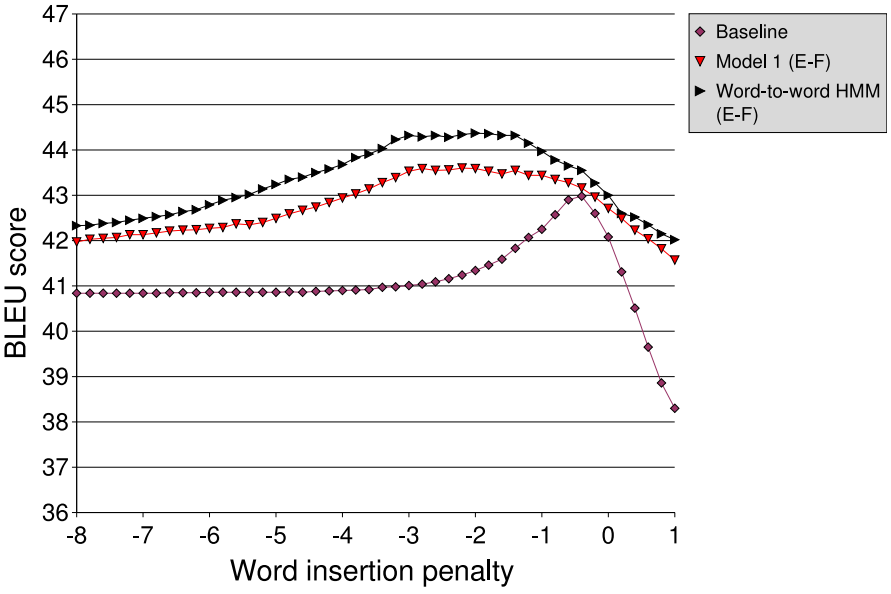


Figure 5.2: N-best rescoring for Arabic-English eval04 *data set, replacing translation model probability with Arabic-English alignment model probability. We maximise* $\mathrm{P}_{LM}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{f}|\mathbf{e})$.

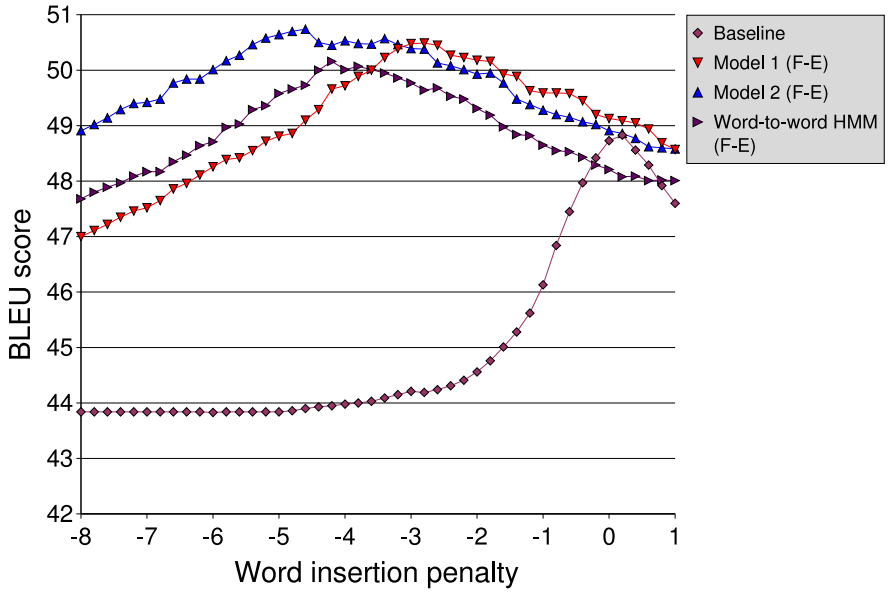


Figure 5.3: N-best rescoring for Arabic-English eval03 *data set, replacing translation model probability with English-Arabic alignment model probability. We maximise* $\mathrm{P}_{LM}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f})$.

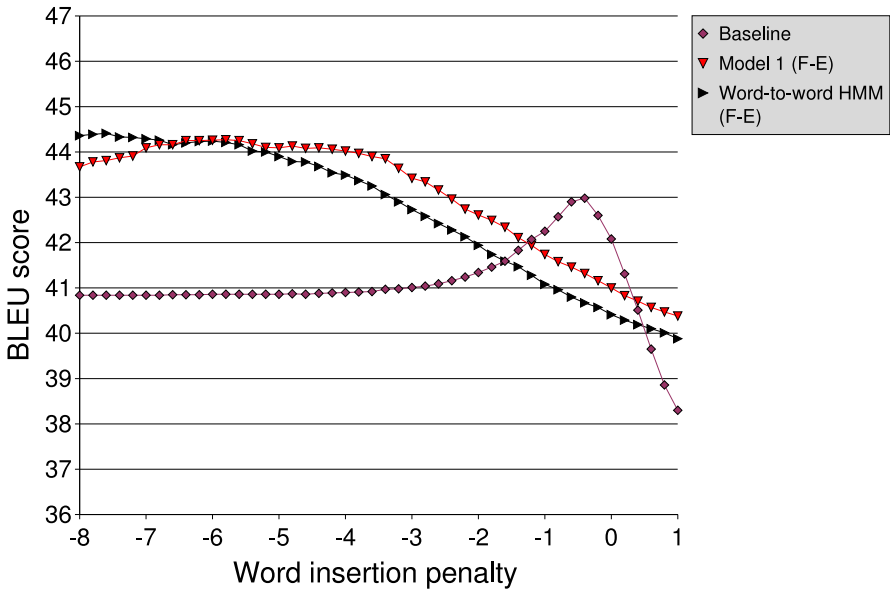


Figure 5.4: N-best rescoring for Arabic-English eval04 *data set, replacing translation model probability with English-Arabic alignment model probability. We maximise* $\mathrm{P}_{LM}(\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f})$.

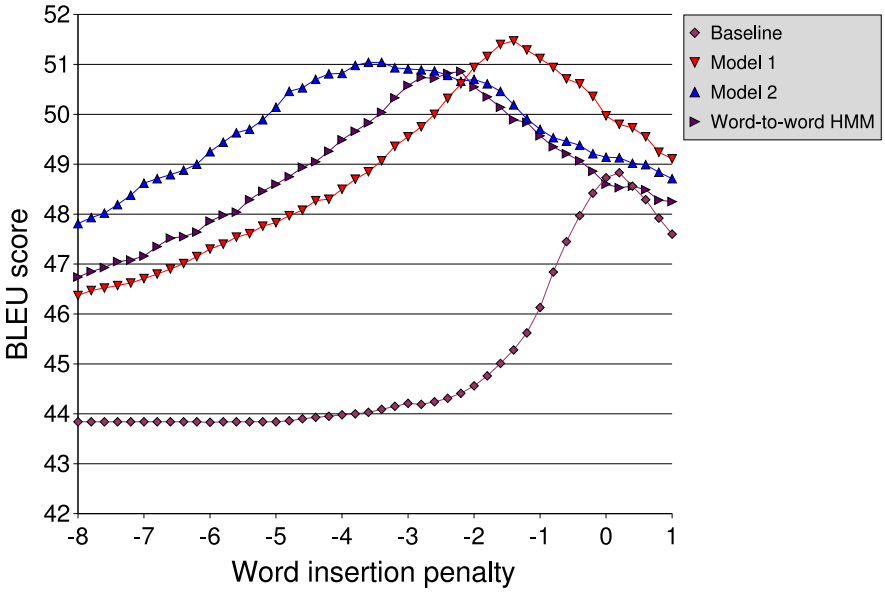


Figure 5.5: N-best list rescoring of the Arabic-English eval03 *data set, maximising* $\mathrm{P}_{LM}(\mathbf{e})\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f})$.

rescoring leads to an increase in translation quality, and the BLEU score using the Arabic-English alignment models is comparable with that obtained obtained using the English-Arabic model.

Ideally, we would like to retain information from as many informative features as possible when determining the best sentence; therefore we would like to include the TTM translation probability in addition to the MTTK probability. The effect of retaining the full TTM score was then investigated, i.e. the decision rule becomes

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}}\, \mathrm{P}_{\mathrm{LM}}(\mathbf{e})\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f}). \tag{5.2}$$

Figures 5.5 and 5.6 show the results on the Arabic-English *eval03* and *eval04* data sets. We see that this strategy is the best performing so far, with an increase in BLEU score of over 2 points on the *eval03* set. This is a significant gain in translation quality, without the use of more training data or the training of more models. Note that this approach is not simply gaining BLEU score by adjusting the word insertion penalty: by varying the insertion penalty for each translation score, we can see that the maximum value of BLEU for the baseline system falls significantly below that achieved through rescoring. We would expect rescoring using the more sophisticated HMMs to lead to improved translation quality, since the models should determine the alignment between the sentences more accurately; however, our results show this not to be the case.

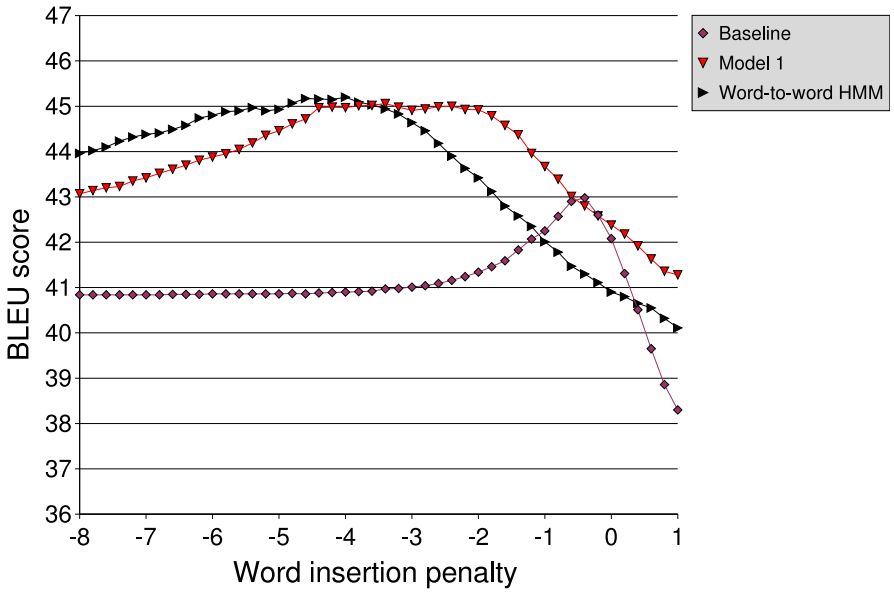


Figure 5.6: N-best list rescoring of the Arabic-English eval04 *data set, maximising* $\mathrm{P}_{LM}(\mathbf{e})\mathrm{P}_{\mathrm{TTM}}(\mathbf{f}|\mathbf{e})\mathrm{P}_{\mathrm{MTTK}}(\mathbf{e}|\mathbf{f})$.

5.4 Comparison of lattice rescoring with N-best rescoring

We now move on to look at rescoring a complete lattice of translation hypotheses rather than taking an N-best list. Lattices of translation hypotheses were generated using the TTM and rescoring was carried out as described in Section 4.5, using the lattice alignment algorithms for Model 1 and the MTTK word-to-word translation HMM model. As discussed in Section 4.6, it is difficult to include the Model 2 score in lattice-based rescoring: due to this difficulty, and the fact that it does not perform significantly differently to Model 1 for N-best list rescoring, it is not used for lattice rescoring.

The baseline figure for a given insertion penalty is obtained by applying the insertion penalty to the lattice and selecting the best path. In this way we can see how the algorithms behave as the word insertion penalty varies. For all lattice rescoring experiments, we use the rescoring scheme that has worked most effectively for the N-best lists, i.e. we use the MTTK foreign to English alignment score in addition to the complete TTM score. We use the tropical semiring, i.e. the cost assigned to a hypothesis by the model corresponds to that of the Viterbi alignment.

A range of pruning thresholds was tested: a pruning threshold too small results in too few hypotheses being present in the lattice for the rescoring to have a significant effect, due to the fact that the majority of hypotheses that would rank highly under the rescoring scheme are removed from the lattice; such results are not presented. Figures 5.7 and 5.8 show the effect of lattice rescoring using Model 1 and HMM alignment models for the *eval03* test set. We see the same patterns for rescoring the *eval04* test set: see Figure 5.9 shows the performance of

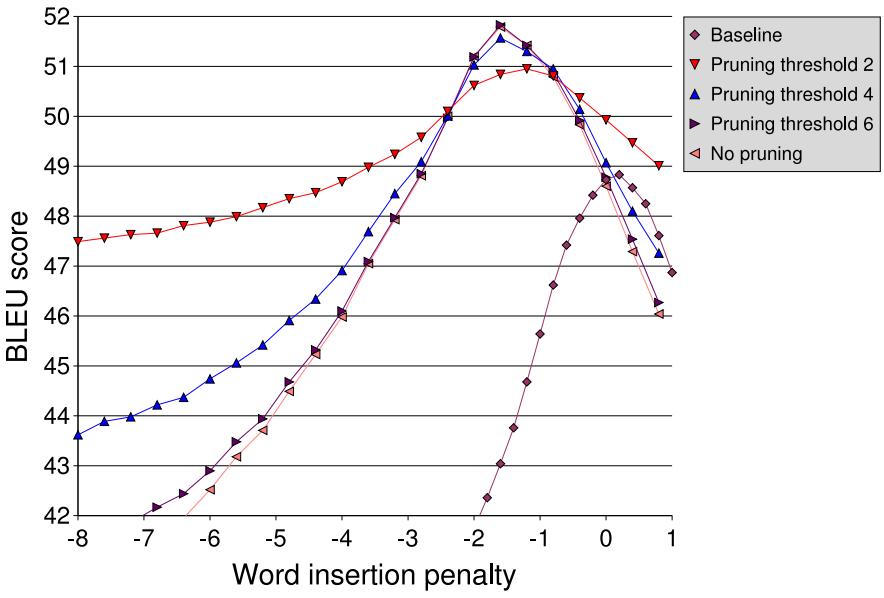


Figure 5.7: Results of lattice rescoring of Arabic-English eval03 *translations using Model 1 with a range of pruning thresholds.*

Model 1 on this data set. Lattice rescoring produces translations at least as good as N-best list rescoring, since all the hypotheses from the N-best list are included in the lattice.

The best scoring combinations of parameters result in significant gains over the baseline: for both data sets, Model 1 produces gains of almost 3 BLEU points if the parameters are adjusted correctly.

5.4.1 Effect of pruning

Pruning of the lattice prior to rescoring removes hypotheses assigned a low probability by the baseline system and can significantly reduce the size of the lattice, which increases the speed of rescoring. The smaller the pruning threshold, the fewer translation hypotheses will remain in the lattice. However, the removal of too many hypotheses may cause search errors, so we balance the desire for small lattices and fast rescoring with the need to retain sufficient sentences that may rank highly after rescoring with the alignment models. N-best list rescoring is also a method of reducing the search space, although lattices corresponding to N-best lists are generally larger; therefore pruning is preferred.

Pruning with a threshold of 2 gives approximately the same performance as 1000-best rescoring. Note that a pruning threshold of 6 produces almost the same result as using no pruning threshold. This indicates that pruning can be applied without adversely affecting results. In fact, it appears that rescoring can increase the score of some of the worst hypotheses and pruning them prior to rescoring can make a small improvement to the BLEU score. We use a pruning threshold of 6 for remaining experiments. Table 5.1 shows a summary of the performance of the rescoring experiments on the Arabic-English translation task. All models include a language model feature $P_{LM}(\mathbf{e})$ in addition to:

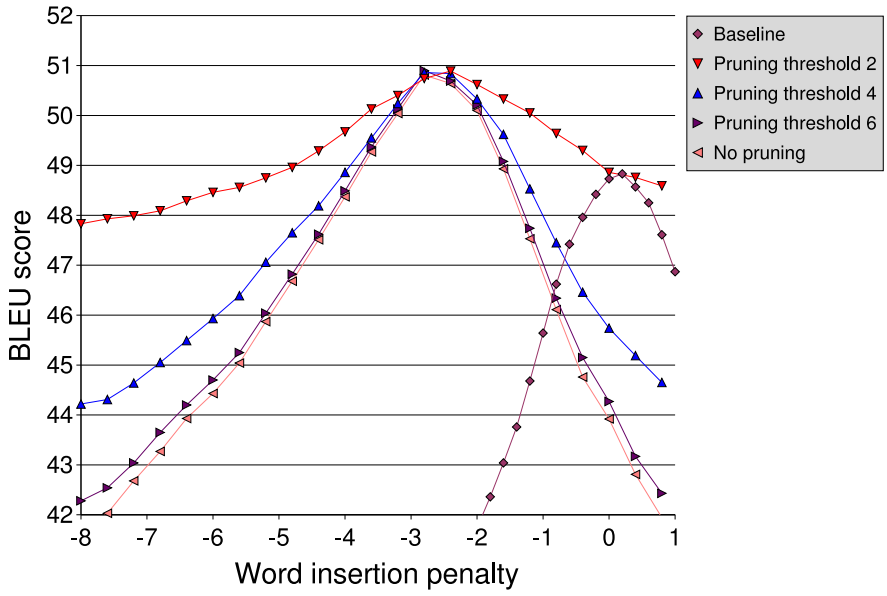


Figure 5.8: Results of lattice rescoring of Arabic-English eval03 *translations using the word-to-word HMM model with a range of pruning thresholds.*

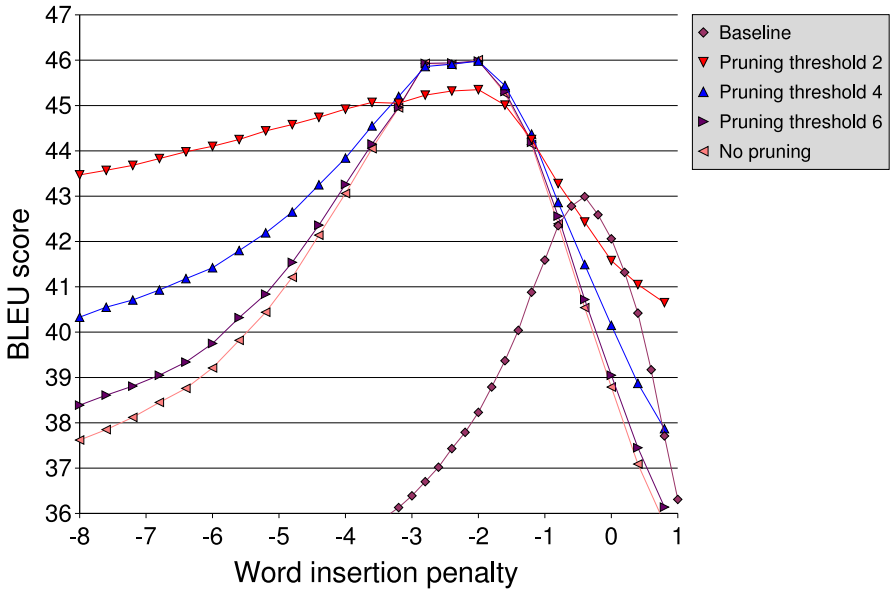


Figure 5.9: Results of lattice rescoring of Arabic-English eval04 *translations using Model 1 with a range of pruning thresholds.*

Model	Features	BLEU score	
		eval03	*eval04*
Baseline	(i)	48.83	42.99
Model 1	(ii)	49.79	43.60
	(iii)	50.49	44.27
	(iv)	50.08	44.41
	(v)	51.47	45.06
	(vi)	**51.83**	**45.98**
HMM	(ii)	50.46	44.37
	(iii)	50.07	44.41
	(iv)	50.23	44.58
	(v)	50.86	45.20
	(vi)	50.69	**46.02**

Table 5.1: Summary of rescoring experiments for Arabic to English translation.

(i) TTM translation model, $P_{TTM}(\mathbf{f}|\mathbf{e})$

(ii) MTTK English to Arabic translation model, $P_{MTTK}(\mathbf{f}|\mathbf{e})$

(iii) MTTK Arabic to English translation model, $P_{MTTK}(\mathbf{e}|\mathbf{f})$

(iv) TTM model $P_{TTM}(\mathbf{f}|\mathbf{e})$; MTTK English to Arabic translation model, $P_{MTTK}(\mathbf{f}|\mathbf{e})$

(v) TTM model $P_{TTM}(\mathbf{f}|\mathbf{e})$; MTTK Arabic to English translation model, $P_{MTTK}(\mathbf{e}|\mathbf{f})$

(vi) TTM model $P_{TTM}(\mathbf{f}|\mathbf{e})$; MTTK Arabic to English translation model, $P_{MTTK}(\mathbf{e}|\mathbf{f})$; lattice rescoring with pruning threshold 6.

We see that the lattice rescoring with Model 1 in the Arabic to English direction, retaining the TTM translation score as a feature, performs best overall (though it is 0.04 BLEU points lower than rescoring using the HMM for the *eval04* set).

5.5 Lattice rescoring for multiple languages

The results for Arabic to English translation show that the rescoring algorithms developed yield improvements in translation performance, and we seek to verify the improvements generalise to different data in different languages. We applied the Model 1 and HMM lattice rescoring algorithms to re-rank a lattice of translation hypotheses for French to English translation of the Europarl *devtest* set. We obtain gains of almost 2 BLEU points over the baseline score. Again, Model 1 performs better than the HMM model. Pruning was necessary for some of the lattices here in order that they could be determinised: fortunately previous results indicate that this should not adversely affect the quality of the results. Figure 5.10 shows the results of rescoring a lattice of hypotheses for the Europarl *devtest* set with Model 1.

Table 5.2 shows the result of lattice rescoring for a number of systems translating four different languages to English. We include results for the Chinese to English *eval04* data set and the TC-STAR Spanish to English *test* set in addition to the languages previously

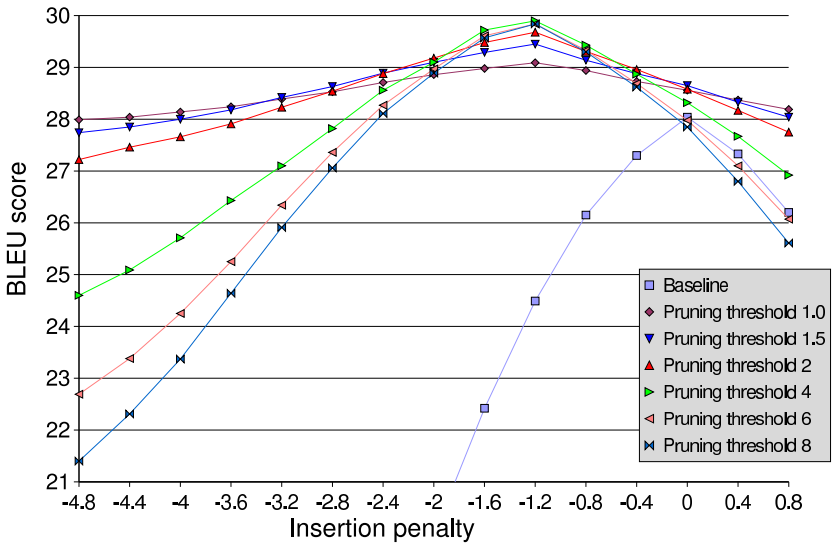


Figure 5.10: Lattice rescoring of Europarl translation lattices using Model 1 with a range of pruning thresholds. The baseline is the Bleu *score obtained by using 1-best score from the TTM lattice.*

Language pair	**Test set**	Bleu **score**		
		Baseline	**Model 1**	**HMM**
Arabic-English	eval03*	48.83	**51.83**	50.90
	eval04*	42.99	46.02	**46.04**
Chinese-English	eval03*	29.50	**31.21**	30.58
French-English	devtest*	28.04	**29.90**	29.26
	devtest (MERT iteration 1)	29.16	**30.67**	30.04
	devtest (MERT iteration 2)	29.78	**30.81**	29.67
	test (MERT iteration 2)	29.23	**30.13**	29.17
	test out-of-domain (MERT it. 2)	20.82	**21.63**	20.84
Spanish-English	TC-STAR 2006 test	46.15	**47.29**	–

Table 5.2: Rescoring of lattices of translation hypotheses using Model 1 and word-to-word HMM alignment models

tested. In each case, we obtain an increases in Bleu score when rescoring with Model 1 and the word-to-word HMM, though Model 1 gives a larger increase. The systems marked with a * use a baseline with the standard TTM model with uniform feature weights, whereas the others use a system where the TTM is formulated as a log-linear model (as described in Section 2.5.1) and the feature weights have been trained using MERT. We see that the gains in translation quality are present even with a baseline that has been improved due to MERT. In most cases, we see an improvement of at least 1 Bleu point as a result of lattice rescoring, and there is also improvement on the out-of-domain test data.

As mentioned in Section 4.6, the Model 1 alignment component $\frac{1}{J+1}$ effectively acts as a

word insertion penalty. A comparison between using this cost as part of the Model 1 rescoring process and omitting it was carried out: as expected, the result was almost identical except it yielded a different optimum insertion penalty.

5.6 Lexical weighting of phrase pairs

Many machine translation systems (Chiang, 2005; Koehn, 2004) use alignment models for lexical weighting of phrase pairs, a technique introduced by Koehn et al. (2003). Instead of using just a phrase-to-phrase translation probability determined by co-occurrence counts of the phrases in the parallel text, word-to-word translation probabilities of the words within the phrases are used to determine how well one phrase translates to another. For a phrase pair ($v = f_1^L, u = e_1^K$) and an alignment A between them, a lexical weight is computed by finding

$$\mathrm{P}_w(v|u, A) = \prod_{j=1}^{L} \frac{1}{|i : (i,j) \in A|} \sum_{(i,j)\in A} t(f_j|e_i), \tag{5.3}$$

where $t(f_j|e_i)$ is the word-to-word translation probability obtained from the alignment model. For a phrase pair with more than one alignment, the lexical weight is taken to be that of the alignment with the highest weight:

$$\mathrm{P}_w(v|u) = \max_A \mathrm{P}_w(v|u, A). \tag{5.4}$$

This lexical weight is used as a feature in the log-linear model of translation. Our baseline system did not use this lexical weighting at the time of these experiments.

The alignment models are not used for reordering of hypothesis sentences; they are restricted to rescoring sentences that have been produced by the basic distortion model within the translation system. This means that the more sophisticated models of alignment will not lead to improved order in the output sentences unless the distortion model is capable of generating those reorderings. Therefore it is possible that the rescoring process simply captures lexical information in a different way to the lexical weighting of phrase pairs. There could be advantages to either approach: using phrase-to-phrase weights can help the phrases in a phrase pair to be good translations of each other, while using alignment models to rescore the entire sentence means that a phrase will be preferred if related words occur in a different phrase pair.

Simple experiments were carried out to see if the two approaches are complementary, i.e. whether the use of both features can improve the quality of translation more than using one feature alone. The translation system for the WMT 10 French-English evaluation, described in Chapter 9, was used to produce lattices of translation hypotheses for rescoring with alignment models. Experiments were carried out on the *Tune* and *Test* sets for this evaluation; these contain 2051 and 2525 sentences respectively. The baseline system includes features for lexical weighting of rule pairs; we aim to find suitable weights for the alignment model scores so that rescoring of the lattices leads to gains in translation quality.

The lattices from the translation system were rescored using Model 1 and HMM alignment models. For Model 1, a parameter sweep was carried out to determine the best weight for the alignment model translation probability while the weight for the $\frac{1}{I+1}$ component of the alignment probability was kept at zero; for the HMM, the weights of the translation probability $t(e|f)$ and alignment component $a(j|j', J)$ were both varied.

The alignment model rescoring did not improve the BLEU score of the resulting lattice for any feature weights and actually degraded the baseline system. From these results, it appears that the inclusion of alignment model rescoring does not improve translation quality for a sufficiently good baseline system with lexical weighting of phrase-to-phrase translation rules. However, it is possible that full integration of the rescoring into MERT may provide gains, since the use of alignment models in this way can account for lexical dependencies between phrases.

5.7 Discussion

The alignment model of the baseline translation system is weak since it has been simplified in order for it to be computationally feasible to use it in decoding (see Section 2.5 for a discussion of the distortion models used during translation). The phrase translation model is also inaccurate as the translation table is simply estimated by counting phrases that co-occur in sentence pairs in the parallel text (see Section 3.10.3 for a description of how the phrase-to-phrase translation probabilities are obtained), rather than estimating the phrase-to-phrase translation probabilities in a more sophisticated way. Including the alignment models in lattice-based or N-best list rescoring is a way of addressing this problem.

The MTTK and TTM models are complementary in that both of them provide independent information, and combining the word level information of the MTTK models with the phrase information of the TTM model produces gains in translation quality. By applying the improved MTTK translation models with better alignment to the search space generated by the TTM, we produce sentence hypotheses that score higher under the BLEU metric.

The performance of lattice rescoring at least matches that of N-best list rescoring, and in some cases exceeds it. This is because lattice alignment is able to efficiently align a greater proportion of the translation hypotheses (all of the hypotheses in most cases) and suffers from fewer search errors. A further benefit of lattice alignment is that it is quicker. The lattice rescoring algorithm was tested with a number of systems in four different languages, and rescoring produced gains in BLEU score for every language pair tested. Therefore, it should also generalise to be effective for other language pairs. Lattice rescoring was included in the Cambridge University Engineering Department system for the NIST 08 Arabic-to-English translation evaluation (Blackwood et al., 2009). Since these algorithms take a lattice as input and output a lattice, they can be used as an intermediate step prior to further rescoring, or additional lattice-based post-processing such as MBR or system combination.

Model 1 generally performs better than the word-to-word HMM, even though it is essentially a lexical weighting for the sentences based on how often (English, foreign) word pairs have co-occurred in training data. We would expect the performance to increase as the complexity of the models increased, since the more complex models with more parameters should be able to model the data better. In particular, Model 1 has a uniform alignment model, so no information about alignment is used in the rescoring and we would expect the HMM to be able to model the alignment much better and therefore provide a more accurate translation score. We know that the word-to-word HMM model represents the alignments of the training data far better than Model 1, and translation systems initialised from the HMM alignments perform much better, so it is perhaps surprising that it fails to perform as well for rescoring.

Although all of these experiments showed improvement in BLEU score as a result of lattice rescoring, it appears to lose its effectiveness when additional rescoring techniques are

used. Blackwood et al. (2008b) introduce phrasal segmentation models for lattice rescoring and perform further rescoring with a 5-gram language model. Although alignment models should, in theory, capture complementary information to both of these, rescoring with Model 1 no longer results in gains in BLEU score, whether it is used before or after the phrasal segmentation model and language model are applied.

The rescoring procedure also appears to lose its effectiveness when used with a baseline system that uses lexical weighting for phrase-to-phrase translation probabilities. It appears that these two features capture approximately the same information and are not complementary.

5.8 Future work

Model 1 word-to-word translation probabilities are used as a feature in some log-linear translation systems, such as the RWTH Aachen system (Bender et al., 2007), where they are used on an N-best list. The work presented here allows the translation probabilities to be used with lattices of translation hypotheses as well as N-best lists, and Model 1 rescoring need not alter the structure of the lattice. It also allows for the use of HMM models in lattice rescoring.

Recent developments in discriminative training for machine translation allow the use of many features and effective estimation of the corresponding weights in the log-linear model (Chiang et al., 2009). Model 1 translation probabilities in both directions can be used as features in log-linear models. The models are generated during the training process, so there is no additional computation required for their calculation and they can be added with relatively little effort. In particular, Model 1 rescoring can be modified so that it does not increase the structure of the lattice. The integration of Model 1 scores into a log-linear model and their inclusion in discriminative training may therefore yield results.

The HMM model can be used in addition as it may provide complementary information: in particular, the alignment component of the model may prove useful if added as a feature in its own right and the weight appropriately tuned. Improved discriminative alignment models presented in Chapter 6 and context-dependent alignment models presented in Chapter 8 can also be added as features in a log-linear model in this way.

CHAPTER 6

Discriminative Training of Alignment Models

6.1 Introduction

Discriminative training is a technique that has been shown to work well for the training of acoustic models in ASR (Valtchev et al., 1997; Woodland and Povey, 2002). Rather than maximising the probability of an acoustic observation given its word sequence as occurs during EM training, the model is trained to maximise the probability of the correct word sequence given an observation while reducing the probability of competing, incorrect hypotheses. The main application of discriminative training to machine translation has been for the tuning of small numbers of feature weights in a translation system (Och, 2003), though some work has been done to adopt a similar approach to alignment modelling (Fraser and Marcu, 2006b; Taskar et al., 2005).

We take inspiration from ASR and aim to discriminatively re-estimate large numbers of model parameters to increase the probability of correct alignments while reducing the probability of incorrect alignments. We do this with the aid of corpora of manually aligned parallel text that have been annotated with correct alignment links.

This chapter introduces discriminative training for some of the statistical alignment models described in Chapter 3. We describe modifications to the training process for Model 1 and the word-to-word HMM alignment model for training by Maximum Mutual Information Estimation (MMIE). We begin by discussing previous work on discriminative training for alignment modelling, along with the differences in the approach presented in this thesis.

Then the algorithm itself is presented, along with a discussion of the issues we face during parameter estimation.

6.2 Previous and related work

There have been a number of previous papers that deal with discriminative training of alignment models. Berger et al. (1996) were the first to do so: they use their discriminative model to perform alignment, in addition to using it for translation.

Fraser and Marcu (2006b) introduce semi-supervised training of alignment models that alternate an Expectation Maximisation step on a large training corpus and a discriminative training step on a smaller manually-aligned sub-corpus. Similar to the discriminative approach to machine translation described in Section 2.1, the alignment model is viewed as a log-linear model composed of features h_m with feature weights λ_m. Then the probability of alignment $\mathbf{a}$ of $\mathbf{f}$ with $\mathbf{e}$ is given by

$$\begin{aligned} \mathrm{P}_{\mathbf{\Lambda}}(\mathbf{f}, \mathbf{a}|\mathbf{e}) &= \frac{\exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{a}, \mathbf{f}, \mathbf{e}))}{\sum_{\mathbf{f}'} \sum_{\mathbf{a}'} \exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{a}', \mathbf{f}', \mathbf{e}))} \\ &= \frac{\exp(\mathbf{\Lambda}^T \mathbf{h}(\mathbf{a}, \mathbf{f}, \mathbf{e}))}{\sum_{\mathbf{f}'} \sum_{\mathbf{a}'} \exp(\mathbf{\Lambda}^T \mathbf{h}(\mathbf{a}', \mathbf{f}', \mathbf{e}))}. \end{aligned} \tag{6.1}$$

The IBM Model 4, described in Section 3.3.5, is decomposed into the following sub-models, (the log of) each of which is used given a weight and used as a feature in a log-linear model:

- word-to-word translation probabilities, $t(f|e)$;
- fertility probabilities $\phi(e)$, governing how many words are generated from source word e;
- distortion probabilities, determining the placement of words in target phrase.

An additional 11 features are added, including translation tables from combinations of alignments in both alignment directions, translation tables using approximate stems, backoff fertility probabilities and penalties for unaligned words in each language. The algorithm used is known as EMD, since there is an expectation (E) step where counts are collected, a maximisation (M) step where the counts are used as in standard EM to train the sub-models, and a discriminative (D) step which optimises the weight vector $\mathbf{\Lambda}$.

Taskar et al. (2005) present a purely discriminative matching approach to word alignment, where none of the features are derived from a generative model. Each pair of words (e_i, f_j) in a sentence pair $(\mathbf{e}, \mathbf{f})$ is associated with a score $s_{ij}(\mathbf{e}, \mathbf{f})$ reflecting the desirability of aligning e_i with f_j. The alignment predicted by the model is then the highest scoring match subject to some constraints. Linear programming techniques are used to find weights $\mathbf{\Lambda}$ such that a loss function, which penalises incorrect alignments, is minimised. The principal feature used is the Dice coefficient (Dice, 1945) to measure the co-occurrence of e and f in training data:

$$\mathrm{Dice}(e, f) = \frac{2c(e, f)}{c(e) + c(f)}, \tag{6.2}$$

where $c(e)$ is the number of times e occurs, $c(f)$ is the number of times f occurs and $c(e, f)$ is the number of times they co-occur in a sentence pair. Other features include distance

features to penalise long distance links, an edge penalty to favour high precision alignments and word similarity features for cognates (words with a common etymological origin that are similar in the two languages being aligned). The dependency between alignment positions is not modelled explicitly as it is in the HMM, but the Dice coefficient of the following word pair is used so that words whose following words co-occur frequently are more likely to align themselves. A word frequency feature equal to the difference in log rank between the words discourages common words from translating to rare ones. Performance of these features alone is reasonable, but adding Model 4 alignments as features gave a 22% improvement over the Model 4 alignments alone. One potential advantage is that they do not use alignment models in both directions, avoiding the need for symmetrisation. However, only one-to-one alignments are permitted, which does not reflect real alignments.

Blunsom and Cohn (2006) use conditional random fields to estimate discriminative alignment probabilities from word-aligned training data. Rather than optimising $\mathbf{\Lambda}$ to maximise the log probability of the training data, they define a prior distribution $\mathrm{P}(\mathbf{\Lambda})$ and find the maximum *a posteriori* (MAP) estimate to reduce over-fitting to the training data:

$$\mathbf{\Lambda}^{\mathrm{MAP}} = \underset{\mathbf{\Lambda}}{\operatorname{argmax}}\, \mathrm{P}_{\mathbf{\Lambda}}(\mathcal{D})\mathrm{P}(\mathbf{\Lambda}), \tag{6.3}$$

where $\mathcal{D} = \{\hat{\mathbf{a}}_s, \mathbf{f}_s, \mathbf{e}_s\}_{s=1}^{S'}$ is the word-aligned training data. They use features similar to Taskar et al. (2005) but use a separate model for each translation direction and allow many-to-one alignments. In addition to the Dice coefficient, translation probabilities from IBM Model 1 are used, as are part-of-speech tags and a bilingual dictionary. Markov features to model the monotonic tendencies of alignments, similar to those used by the HMM model, are used. Neither Taskar et al. (2005) nor Blunsom and Cohn (2006) evaluate their alignment models by building translation systems from those alignments, evaluating solely using AER on a held-out set of manually aligned data.

Lambert et al. (2007) form a discriminative model of alignment, using two features to penalise unlinked words, a link bonus, word and part-of-speech association models, two features to penalise crossing links, a penalty against links that prevent phrase pairs being extracted. They do not use reference alignments to tune the feature weights: instead, translation of a development set is carried out at each iteration of training and the feature weights λ_m are varied to maximise the translation quality of an MT system initialised on the alignment output by the model. This avoids the need for manual alignments, and removes the problem of finding an alignment metric that correlates well with translation quality. However, the size of the training corpus is severely limited, since the whole corpus must be aligned at each iteration of discriminative training and the algorithm used takes 80 iterations to converge. With some of our alignment models taking 60 hours of processor time per iteration on large corpora, it would not be feasible to perform this kind of training on large data sets.

Varea et al. (2002) use discriminative training to determine the weights of features that incorporate context by testing for the presence of words within a window. In contrast with the other methods which use global feature weights, the weights here are estimated separately for each source word e, subject to its occurring sufficiently frequently in the training data, which means we have a separate discriminative translation model for each source word.

All of these previous approaches used discriminative techniques to tune only a small number of feature weights within a log-linear model. The approach presented here differs in that a large number of the model parameters, including word-to-word translation probabilities and alignment probabilities, are discriminatively re-estimated with the aim of moving the

alignments produced by the models closer to those produced manually. The initial steps of training are unsupervised using EM on the full parallel text training data, but we use manual alignments of a smaller data set for supervised training once unsupervised training is complete. The idea is that we make use of the manual alignments to sharpen the models, but retain information from the complete parallel text for which no manual alignments are available. The approach presented here has more in common with work on discriminative training carried out for Automatic Speech Recognition, where the HMM acoustic model is refined using discriminative methods with sequences of labelled phones.

6.3 Introducing MMIE training for alignment models

The MTTK models described in Chapter 3 are trained by Maximum Likelihood Estimation (MLE), in which we estimate the model parameters in order to maximise the likelihood of the training data. Given a parallel text sentence pair $(\mathbf{f}, \mathbf{e})$, we vary the parameters θ of the model during MLE to maximise

$$\log \mathrm{P}_\theta(\mathbf{f}|\mathbf{e}) = \sum_{\text{alignments } \mathbf{a}} \mathrm{P}_\theta(\mathbf{f}, \mathbf{a}|\mathbf{e}). \tag{6.4}$$

The training corpus is made up of multiple sentence pairs: write $E = \{\mathbf{e}_1, \ldots, \mathbf{e}_S\}$ and $F = \{\mathbf{f}_1, \ldots, \mathbf{f}_S\}$ for the source and target parallel text sentences respectively. We assume that the sentence pairs are independent and estimate θ to maximise the probability of generating the target sentences given the source sentences:

$$\mathrm{P}(F|E) = \prod_{s=1}^{S} \mathrm{P}_\theta(\mathbf{f}_s|\mathbf{e}_s) = \prod_{s=1}^{S} \sum_{\mathbf{a}} \mathrm{P}_\theta(\mathbf{f}_s, \mathbf{a}|\mathbf{e}_s). \tag{6.5}$$

We now describe how discriminative training can be used to refine the models after maximum likelihood estimation. For some sentences of the training data, we have a human-produced alignment $\hat{\mathbf{a}}$ between the parallel text sentences $\mathbf{f}, \mathbf{e}$, and we wish to use this information to improve the overall quality of alignment. Assume that we have manual alignments $\hat{\mathbf{a}}_1^S$ for S' sentence pairs $\{(\mathbf{f}_1, \mathbf{e}_1), \ldots (\mathbf{f}_{S'}, \mathbf{e}_{S'})\}$. Our objective is then to improve our alignment model so that it produces the correct alignment for as many of these sentences as possible, and we seek to maximise the probability of this correct alignment while reducing the probability of incorrect alignments that are possible under the models.

We define the *mutual information* of the target sentence $\mathbf{f}$ and alignment $\mathbf{a}$ given the source sentence $\mathbf{e}$ as

$$I_\theta(\mathbf{a}|\mathbf{e}, \mathbf{f}|\mathbf{e}) = \frac{\mathrm{P}_\theta(\mathbf{a}, \mathbf{f}|\mathbf{e})}{\mathrm{P}_\theta(\mathbf{f}|\mathbf{e})\mathrm{P}_\theta(\mathbf{a}|\mathbf{e})} \tag{6.6}$$

During maximum mutual information estimation (MMIE), we aim to maximise this quantity. The rationale behind this is that by maximising the mutual information, we minimise the information needed to specify $\mathbf{a}$ when the target sentence $\mathbf{f}$ is known.

For a reference sentence-level alignment $\hat{\mathbf{a}}$ to be correctly predicted by the model, we require $\mathrm{P}(\hat{\mathbf{a}}|\mathbf{f}, \mathbf{e}) > \mathrm{P}(\mathbf{a}|\mathbf{f}, \mathbf{e})$ for all alignments $\mathbf{a} \neq \hat{\mathbf{a}}$. Motivated by the work carried out

by Normandin (1996) for speech recognition, we aim to modify the model parameters θ to maximise $\mathrm{P}_\theta(\mathbf{a}|\mathbf{f},\mathbf{e})$ on average over the training corpus. Now

$$\begin{aligned}\mathrm{P}_\theta(\hat{\mathbf{a}}|\mathbf{f},\mathbf{e}) &= \frac{\mathrm{P}_\theta(\hat{\mathbf{a}},\mathbf{f}|\mathbf{e})}{\mathrm{P}_\theta(\mathbf{f}|\mathbf{e})} \\ &= \frac{\mathrm{P}_\theta(\hat{\mathbf{a}},\mathbf{f}|\mathbf{e})}{\sum_{\mathbf{a}}\mathrm{P}_\theta(\mathbf{a},\mathbf{f}|\mathbf{e})} \end{aligned} \tag{6.7}$$

The quantity we therefore wish to maximise over the training corpus is:

$$\prod_{s=1}^{S'}\frac{\mathrm{P}_\theta(\hat{\mathbf{a}}_s,\mathbf{f}_s|\mathbf{e}_s)}{\sum_{\mathbf{a}}\mathrm{P}_\theta(\mathbf{a},\mathbf{f}_s|\mathbf{e}_s)}. \tag{6.8}$$

We define the *objective function* $\mathcal{F}$ to be the log of this, i.e.

$$\mathcal{F} = \sum_{s=1}^{S'}\log\frac{\mathrm{P}_\theta(\hat{\mathbf{a}}_s,\mathbf{f}_s|\mathbf{e}_s)}{\sum_{\mathbf{a}}\mathrm{P}_\theta(\mathbf{a},\mathbf{f}_s|\mathbf{e}_s)} \tag{6.9}$$

For simplicity of notation, we introduce a "general" model which has a probability distribution P_g equal to the sum over all alignments, hence our objective function can be written

$$\mathcal{F} = \sum_{s=1}^{S'}\log\frac{\mathrm{P}_\theta(\hat{\mathbf{a}}_s,\mathbf{f}_s|\mathbf{e}_s)}{\mathrm{P}_g(\mathbf{f}_s|\mathbf{e}_s)} \tag{6.10}$$

Maximisation of our objective function is technically conditional maximum likelihood estimation (CMLE) (Nadas, 1983) since it lacks the $\mathrm{P}_\theta(\mathbf{a}|\mathbf{e})$ term in the denominator of Equation (6.7). However, CMLE is often referred to as MMIE in speech recognition literature, and we follow this convention here; hence, we refer to our model training procedure as MMIE.

6.3.1 Parameter estimation

Estimation of parameters using MMIE is more complex than with MLE. There are no closed-form re-estimation formulae, so a common solution is to use gradient descent. Parameter estimation by gradient descent should, with a sufficiently small step size, converge to a local maximum of the objective function. The problem with this is we do not want to use a small step size as this results in slow convergence requiring a large number of iterations, which we cannot afford due to the computational complexity of each iteration. For this reason, research has been carried out into alternative parameter re-estimation algorithms for MMIE. We present a principled alternative to gradient methods based on mathematically-proven methods for optimising rational functions, which have been shown experimentally to require fewer iterations than gradient descent.

The Baum-Eagon inequality (Baum and Eagon, 1967) provides an effective iterative procedure for finding a local maximum of a homogeneous polynomial with positive coefficients. Gopalakrishnan et al. (1991) extend this to rational functions, which allows us to find a local maximum of the objective function $\mathcal{F}$. Gopalakrishnan et al. (1989) propose the following re-estimation formula for the translation probabilities $t_{f,e} = t(f|e)$:

$$\hat{t}_{f,e} = \frac{t_{f,e}\left(\frac{\partial\mathcal{F}}{\partial t_{f,e}} + D\right)}{\sum_{f'\in\mathcal{V}_{\mathrm{tgt}}} t_{f',e}\left(\frac{\partial\mathcal{F}}{\partial t_{f',e}} + D\right)}, \tag{6.11}$$

where D is a constant to be determined and the sum in the denominator is taken over all words f' in the target vocabulary $\mathcal{V}_{\text{tgt}}$.

This update procedure has been shown to work for parameter estimation of HMMs for ASR (Valtchev et al., 1997; Woodland and Povey, 2002). Although the models used for alignment in SMT are different, the principle behind what we want to do remains the same. In ASR we try to increase the probability of the correct word sequence given the acoustic data at the expense of competing, incorrect word sequences; for alignment models we aim to increase the posterior probability of the correct alignment, while driving down the probabilities of incorrect alignments competing for probability mass.

It is likely that the data for which we have manual alignments will be a subset of the full data used to train the model, and there will be words that occur in the full data that do not appear in the manually-aligned subset. It is worth noting that the update rule does not change the translation probability associated with unseen source words e, since $\frac{\partial \mathcal{F}}{\partial t_{f,e}} = 0$ for all f, hence

$$\hat{t}_{f,e} = \frac{t_{f,e}\,(0+D)}{\sum_{f'} t_{f',e}\,(0+D)} = \frac{D t_{f,e}}{D \sum_{f'} t_{f',e}} = t_{f,e} \tag{6.12}$$

For target words f not occurring in the manually-aligned training data, some of the probability mass assigned to that word will be re-assigned to other words that do occur. The translation probability is updated according to

$$\hat{t}_{f,e} = \frac{t_{f,e}\,(0+D)}{\sum_{f'} t_{f',e}\left(\frac{\partial \mathcal{F}}{\partial t_{f',e}} + D\right)} = \frac{D}{\sum_{f'} t_{f',e} \frac{\partial \mathcal{F}}{\partial t_{f',e}} + D} t_{f,e} \tag{6.13}$$

6.3.2 Controlling speed of convergence

The speed of convergence is affected by the constant D in Equation (6.11): the larger the value of D, the less $\hat{t}_{f,e}$ differs from $t_{f,e}$. A small value of D results in faster convergence but can result in oscillation of the parameters (Gopalakrishnan et al., 1989), or negative probabilities being assigned in the case of our models.

It can be proven that an update rule of this form will converge to a local optimum for a sufficiently large value of D (Gopalakrishnan et al., 1991). In practice, though, this value of D is computationally expensive to determine. It can be shown that there is a value $D_{\min}$ such that the parameter update is guaranteed to increase the objective function for all $D \geq D_{\min}$ (Gopalakrishnan et al., 1989); however, $D_{\min}$ is so large that convergence is slow and the procedure is impractical (Normandin, 1991).

Gopalakrishnan et al. (1989) report that using a smaller value of D produces fast convergence, but that convergence is not theoretically proven. These authors use

$$D = \max\left\{\max_{f,e}\left(-\frac{\partial \mathcal{F}}{\partial t_{f,e}}\right), 0\right\} + \epsilon, \tag{6.14}$$

where ϵ is a small positive constant. This ensures that $\frac{\partial \mathcal{F}}{\partial t_{f,e}} + D$ is positive for all f, e, since

$$D > -\frac{\partial \mathcal{F}}{\partial t_{f,e}} \quad \Longrightarrow \quad \frac{\partial \mathcal{F}}{\partial t_{f,e}} + D > 0 \text{ for } \frac{\partial \mathcal{F}}{\partial t_{f,e}} < 0; \tag{6.15}$$

$$D > 0 \quad \Longrightarrow \quad \frac{\partial \mathcal{F}}{\partial t_{f,e}} + D > 0 \text{ for } \frac{\partial \mathcal{F}}{\partial t_{f,e}} > 0. \tag{6.16}$$

Importantly, this ensures all probabilities estimated are positive in this case. Normandin (1996) finds that this value of D consistently produces convergence but that convergence is too slow to be useful.

Further modifications have been tried to improve further the speed of convergence and reduce the number of iterations of training required. Parameters that have little data on which to be trained are likely to be unreliably estimated; additionally, for probabilities close to zero, small changes in their value can cause a large change in the gradient, which reduces the efficiency of a gradient search. One approach to avoiding this problem is to scale each component of the gradient $\frac{\partial \mathcal{F}}{\partial t_{f,e}}$ depending on how often the source word e occurs: this can be done with the use of a scaling factor (Normandin, 1996)

$$s(e,f) = \frac{c_e(f) + c_{e,g}(f)}{\sum_{f'} c_e(f') + c_{e,g}(f')}, \tag{6.17}$$

where $c_e(f)$ is the count of f being emitted from e in the reference alignment and $c_{e,g}(f)$ is the count from the general model. We can also replace the partial derivative by an approximation; this was found to be the fastest converging method by Merialdo (1988):

$$\frac{\partial \mathcal{F}}{\partial t_{f,e}} \approx \frac{c_e(f)}{\sum_{f'} c_e(f')} - \frac{c_{e,g}(f)}{\sum_{f'} c_{e,g}(f')}. \tag{6.18}$$

Valtchev et al. (1997) use MMIE for optimising parameters of HMMs for speech recognition, and use a lower bound on D that keeps the variances of the Gaussian distributions positive. They compare the use of a global constant with the use of phone-specific constants, and the convergence rate was two or more times better for the phone-specific constants. This suggests a possible modification to the translation probability update scheme: rather than setting a global constant D for every word, we can use a source word-specific constant D_e:

$$D_e = \max\left\{\max_f\left(-\frac{\partial \mathcal{F}}{\partial t_{f,e}}\right), 0\right\} + \epsilon. \tag{6.19}$$

This should result in fast convergence for the translation probability distribution for each source word e.

6.4 MMIE for Model 1

This section derives the update equations for MMIE training of Model 1. For simplicity, we begin by considering a single sentence and extend the derivation to consider the entire manually aligned data set later. The joint probability of a target sentence f_1^J and an alignment a_1^J is given by

$$\mathrm{P}(f_1^J, a_1^J | e_1^I) = \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j | e_{a_j}) \tag{6.20}$$

$$\begin{aligned} \mathrm{P}(f_1^J | e_1^I) &= \sum_{a_1^J} \frac{\epsilon}{(I+1)^J} \prod_{j=1}^{J} t(f_j | e_{a_j}) \\ &= \frac{\epsilon}{(I+1)^J} \sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} t(f_j | e_{a_j}) \end{aligned} \tag{6.21}$$

We can change the order of the product and summation (Brown et al., 1993), and this can be written

$$\mathrm{P}(f_1^J|e_1^I) = \frac{\epsilon}{(I+1)^J}\prod_{j=1}^{J}\sum_{i=0}^{I} t(f_j|e_i), \tag{6.22}$$

where e_0 is a null token corresponding to a target word not being aligned to any source word. The quantity we seek to maximise (Equation (6.9)) becomes

$$\mathcal{F} = \log\frac{\prod_{j=1}^{J} t(f_j|e_{a_j})}{\prod_{j=1}^{J}\sum_{i=0}^{I} t(f_j|e_i)} = \log\frac{\mathcal{N}}{\mathcal{D}}, \tag{6.23}$$

cancelling the constant factors and setting the numerator and denominator to be $\mathcal{N}$ and $\mathcal{D}$ respectively. Then

$$\frac{\partial\mathcal{F}}{\partial t_{f,e}} = \frac{\partial}{\partial t_{f,e}}\log\mathcal{N} - \frac{\partial}{\partial t_{f,e}}\log\mathcal{D} \tag{6.24}$$

We now consider the differentiation of the two quantities. For words $f \in \mathcal{V}_{\text{tgt}}$ and $e \in \mathcal{V}_{\text{src}}$, define $\#(e)$ to be the number of times e occurs in sentence e_1^I and $\#(f)$ to be the number of times f occurs in sentence f_1^J. Define $\hat{\#}(f,e)$ to be the number of times f is generated from e under the reference alignment $\hat{a}_1^J$. Then

$$\mathcal{N} = \prod_{f,e} t(f|e)^{\hat{\#}(f,e)} \tag{6.25}$$

$$\begin{aligned}\implies \frac{\partial}{\partial t_{f,e}}\log\mathcal{N} &= \frac{\partial}{\partial t_{f,e}}\left(\sum_{f',e'}\hat{\#}(f',e')\log t(f'|e')\right)\\ &= \hat{\#}(f,e)\frac{1}{t_{f,e}}\end{aligned} \tag{6.26}$$

We can re-write the denominator $\mathcal{D}$ to make differentiation simpler:

$$\begin{aligned}\sum_{i=0}^{I} t(f_j|e_i) &= \sum_{e}\sum_{i=0}^{I} t(f_j|e_i)\delta_e(e_i)\\ &= \sum_{e}\#(e)t(f_j|e)\end{aligned} \tag{6.27}$$

$$\begin{aligned}\implies \mathcal{D} &= \prod_{j=1}^{J}\sum_{i=0}^{I} t(f_j|e_i)\\ &= \prod_{j=1}^{J}\left[\sum_{e}\#(e)t(f_j|e)\right]\\ &= \prod_{f}\left[\sum_{e}\#(e)t(f|e)\right]^{\#(f)}\end{aligned} \tag{6.28}$$

We can then differentiate as follows:

$$\begin{aligned}
\frac{\partial}{\partial t_{f,e}} \log \mathcal{D} &= \frac{\partial}{\partial t_{f,e}} \left(\sum_{f'} \#(f') \log \left[\sum_{e'} \#(e') t(f'|e') \right] \right) \\
&= \sum_{f'} \#(f') \frac{\partial}{\partial t_{f,e}} \log \left[\sum_{e'} \#(e') t(f'|e') \right] \\
&= \sum_{f'} \#(f') \frac{1}{\sum_{e''} \#(e'') t(f'|e'')} \frac{\partial}{\partial t_{f,e}} \left(\sum_{e'} \#(e') t(f'|e') \right) \\
&= \frac{\#(f)\#(e)}{\sum_{e'} \#(e') t(f|e')}. \qquad (6.29)
\end{aligned}$$

Therefore, the partial derivative of the objective function is

$$\frac{\partial \mathcal{F}}{\partial t_{f,e}} = \frac{\hat{\#}(f,e)}{t_{f,e}} - \frac{\#(f)\#(e)}{\sum_{e'} \#(e') t_{f,e'}} \qquad (6.30)$$

and the update rule is

$$\begin{aligned}
\hat{t}_{f,e} &= \frac{\hat{\#}(f,e) - \frac{\#(f)\#(e)t_{f,e}}{\sum_{e'} \#(e') t_{f,e'}} + D t_{f,e}}{\sum_{f'} \left[\hat{\#}(f',e) - \frac{\#(f')\#(e)t_{f',e}}{\sum_{e'} \#(e') t_{f',e'}} + D t_{f',e} \right]} \\
&= \frac{\hat{\#}(f,e) - \frac{\#(f)\#(e)t_{f,e}}{\sum_{e'} \#(e') t_{f,e'}} + D t_{f,e}}{\sum_{f'} \left[\hat{\#}(f',e) - \frac{\#(f')\#(e)t_{f',e}}{\sum_{e'} \#(e') t_{f',e'}} \right] + D} \qquad (6.31)
\end{aligned}$$

6.4.1 Training on the entire corpus of human-aligned data

For reasons of simplicity, the above considers only one sentence. We wish to use the entire training corpus of manually-aligned sentence pairs $(\mathbf{f}_1, \mathbf{e}_1), \ldots, (\mathbf{f}_{S'}, \mathbf{e}_{S'})$ to train the models and we maximise

$$\mathcal{F} = \sum_{s=1}^{S'} \log \frac{\mathrm{P}(\hat{\mathbf{a}}_s, \mathbf{f}_s | \mathbf{e}_s)}{\sum_{\mathbf{a}} \mathrm{P}(\mathbf{a}, \mathbf{f}_s | \mathbf{e}_s)} \qquad (6.32)$$

Then $\frac{\partial \mathcal{F}}{\partial t_{f,e}}$ becomes

$$\frac{\partial \mathcal{F}}{\partial t_{f,e}} = \sum_{s=1}^{S'} \left(\frac{\hat{\#}_s(f,e)}{t_{f,e}} - \frac{\#_s(f)\#_s(e)}{\sum_{e'} \#_s(e') t_{f,e'}} \right) \qquad (6.33)$$

and the update equation is

$$\hat{t}_{f,e} = \frac{\sum_{s=1}^{S'} \left(\hat{\#}_s(f,e) - \frac{\#_s(f)\#_s(e)t_{f,e}}{\sum_{e'} \#_s(e') t_{f,e'}} \right) + D t_{f,e}}{\sum_{f'} \sum_{s=1}^{S'} \left[\hat{\#}_s(f',e) - \frac{\#_s(f')\#_s(e)t_{f',e}}{\sum_{e'} \#_s(e') t_{f',e'}} \right] + D} \qquad (6.34)$$

Accumulator	Description
$n_s(f,e)$	Number of times f generated from e under reference alignment, $\hat{\#}_s(f,e)$
$m_s(f,e)$	$\#_s(f)\#_s(e)t_{f,e}$
$x_s(f)$	$\sum_{e'} \#_s(e')t_{f,e'}$

Table 6.1: Accumulators defined for collecting sentence-level counts required for MMIE

6.4.2 Computing statistics

We now have update equations for estimation of the parameters; however we need to be able to compute the statistics required to carry out the update equations. The quantities in the update equation (Equation (6.34)) can be expressed as:

$$\hat{\#}_s(f,e) = \sum_{j=1}^{J} \delta(f_j = f \wedge e_{\hat{a}_j} = e) = \sum_{j=1}^{J} \delta_f(f_j)\delta_e(e_{\hat{a}_j}) \tag{6.35}$$

$$\begin{aligned} \#_s(e)\#_s(f)t_{f,e} &= \sum_{j=1}^{J}\sum_{i=1}^{I} \delta(f_j = f \wedge e_i = e)t_{f_j,e_i} \\ &= \sum_{j=1}^{J}\sum_{i=1}^{I} \delta_f(f_j)\delta_e(e_i)t_{f_j,e_i} \end{aligned} \tag{6.36}$$

$$\sum_{e'} \#_s(e')t_{f,e'} = \sum_{f' \in \{f_1,\ldots,f_J\}} \sum_{i=1}^{I} \delta_f(f')t_{f',e_i} \tag{6.37}$$

In order to calculate these statistics, we define accumulators for each sentence s, whose function can be viewed as adding terms from the $(I+1) \times J$ matrix of translation probabilities $X_{e_0^I, f_1^J}$ whose entries are given by $x_{i,j} = t(f_j|e_i)$. Table 6.1 shows the accumulators defined and Figure 6.1 illustrates which terms from the matrix are added. These are calculated per sentence, and the resulting counts are used to calculate

$$\sum_s \left(\hat{\#}_s(f,e) - \frac{\#_s(f)\#_s(e)t_{f,e}}{\sum_{e'} \#_s(e')t_{f,e'}} \right) = \sum_s \left(n_s(f,e) - \frac{m_s(f,e)}{x_s(f)} \right). \tag{6.38}$$

This is accumulated over the whole corpus for each word pair (f,e). The algorithm used to calculate these statistics is shown in Figure 6.2. These values can then be substituted into the update equation (Equation (6.34)) during parameter estimation.

6.5 MMIE for the word-to-word HMM

For the word-to-word HMM, the calculation proceeds in the same way as that for Model 1 but we change the estimation procedure to reflect the probability distribution assigned by the

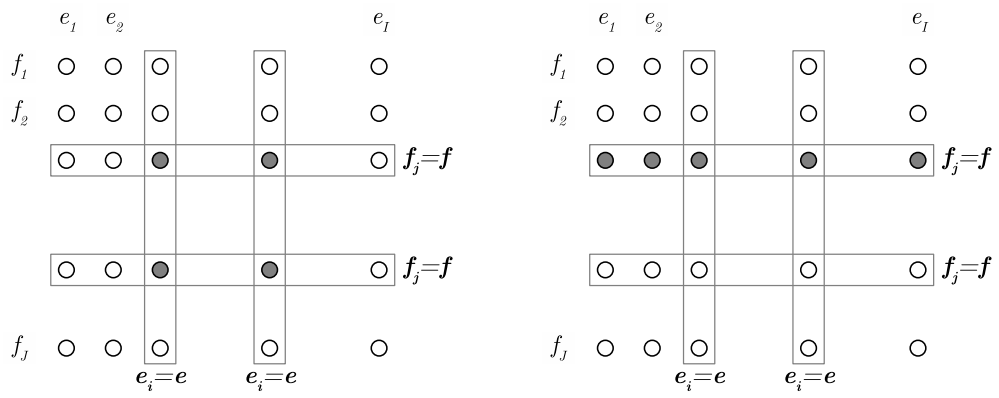


Figure 6.1: Graphical representation of which terms are added to give the required statistics during MMIE for Model 1. For $\#_s(e)\#_s(f)t_{f,e}$ (left), we add to the accumulator for every position i in the source sentence position where $e_i = e$ and every position j in the target sentence for which $f_j = f$. For $\sum_{e'} \#_s(e')t_{f,e'}$ (right), we only add at the first target position j with $f_j = f$.

Input: T-table, parallel text sentences, reference alignment
Output: required statistic
foreach $s = 1, \ldots, S'$ **do**
  foreach $j = 1, \ldots, J_s$ **do**
    $n(f_j, e_{a_j})$ ++;
  end
  foreach $j = 1, \ldots, J_s$ **do**
    foreach $i = 1, \ldots, I_s$ **do**
      $m(f_j, e_i) += t_{f_j,e_i}$;
      if *haven't seen word* f_j *already* **then**
        $x(f_j) += t_{f_j,e_i}$;
      end
    end
  end
  foreach f **do**
    foreach e **do**
      $y(f, e) += n(f, e) - \frac{m(f,e)}{x(f)}$;
    end
  end
end

Figure 6.2: Algorithm used for calculation of Model 1 MMIE statistics. Note that if a target word f_j occurs multiple times in a sentence, we only accumulate $x(f_j)$ the first time f_j occurs in that sentence.

model. Here we wish to maximise

$$\begin{aligned}
\mathcal{F} &= \log \frac{\mathrm{P}(\hat{\mathbf{a}}, \mathbf{f}|\mathbf{e})}{\sum_{\mathbf{a}} \mathrm{P}(\mathbf{a}, \mathbf{f}|\mathbf{e})} \\
&= \log \frac{\prod_{j=1}^{J} a(\hat{a}_j|\hat{a}_{j-1}, I) t(f_j|e_{\hat{a}_j})}{\sum_{a_1^J} \prod_{j=1}^{J} a(a_j|a_{j-1}, I) t(f_j|e_{a_j})} \\
&= \log \frac{\mathcal{N}}{\mathcal{D}} \\
&= \log \mathcal{N} - \log \mathcal{D} \qquad (6.39)
\end{aligned}$$

where

$$\mathcal{N} = \prod_{j=1}^{J} a(\hat{a}_j|\hat{a}_{j-1}, I) t(f_j|e_{\hat{a}_j}) \qquad (6.40)$$

$$\mathcal{D} = \sum_{a_1^J} \prod_{j=1}^{J} a(a_j|a_{j-1}, I) t(f_j|e_{a_j}) \qquad (6.41)$$

As in the case of Model 1, we update the translation probabilities, so we again calculate the partial derivative with respect to $t_{f,e}$. Differentiating $\log \mathcal{N}$ is straightforward, but the calculation for the $\log \mathcal{D}$ is more complicated.

$$\begin{aligned}
\frac{\partial}{\partial t_{f,e}} \log \mathcal{N} &= \frac{\partial}{\partial t_{f,e}} \left(\sum_{j=1}^{J} \log a(\hat{a}_j|\hat{a}_{j-1}, I) + \sum_{j=1}^{J} \log t(f_j|e_{\hat{a}_j}) \right) \\
&= \sum_{j=1}^{J} \frac{\partial}{\partial t_{f,e}} \log t(f_j|e_{\hat{a}_j}) \\
&= \hat{\#}(f, e) \frac{1}{t_{f,e}} \qquad (6.42)
\end{aligned}$$

We differentiate $\log \mathcal{D}$ by calculating $\frac{\partial \mathcal{D}}{\partial t_{f,e}}$ and applying the chain rule to obtain $\frac{\partial}{\partial t_{f,e}} \log \mathcal{D}$:

$$\begin{aligned}
\frac{\partial}{\partial t_{f,e}} \mathcal{D} &= \sum_{a_1^J} \frac{\partial}{\partial t_{f,e}} \left(\prod_{j=1}^{J} a(a_j|a_{j-1}, J) t(f_j|e_{a_j}) \right) \\
&= \sum_{a_1^J} \left(\prod_{j=1}^{J} a(a_j|a_{j-1}, J) \right) \frac{\partial}{\partial t_{f,e}} \left(\prod_{j=1}^{J} t(f_j|e_{a_j}) \right) \\
&= \sum_{a_1^J} \left(\prod_{j=1}^{J} a(a_j|a_{j-1}, J) \right) \left(\frac{\#_{a_1^J}(f,e)}{t_{f,e}} \prod_{j=1}^{J} t(f_j|e_{a_j}) \right) \\
&\quad \text{(where } \#_{a_1^J}(f,e) \text{ is the number of times } f \text{ is aligned to } e \text{ under alignment } a_1^J\text{)} \\
&= \sum_{a_1^J} \frac{\#_{a_1^J}(f,e)}{t_{f,e}} \prod_{j=1}^{J} a(a_j|a_{j-1}, J) t(f_j|e_{a_j}) \qquad (6.43) \\
\frac{\partial}{\partial t_{f,e}} \log \mathcal{D} &= \frac{1}{\mathcal{D}} \frac{\partial}{\partial t_{f,e}} \mathcal{D} \\
&= \frac{\sum_{a_1^J} \frac{\#_{a_1^J}(f,e)}{t_{f,e}} \prod_{j=1}^{J} a(a_j|a_{j-1}, J) t(f_j|e_{a_j})}{\sum_{a_1^J} \prod_{j=1}^{J} a(a_j|a_{j-1}, J) t(f_j|e_{a_j})} \\
&= \frac{1}{t_{f,e}} \frac{\sum_{a_1^J} \#_{a_1^J}(f,e) \mathrm{P}(f_1^J, a_1^J|e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I)} \qquad (6.44)
\end{aligned}$$

Then the update rule for the HMM word-to-word translation probabilities becomes

$$\hat{t}_{f,e} = \frac{\hat{\#}(f,e) - \frac{\sum_{a_1^J} \#_{a_1^J}(f,e) \mathrm{P}(f_1^J, a_1^J|e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I)} + D t_{f,e}}{\sum_{f'} \left[\hat{\#}(f',e) - \frac{\sum_{a_1^J} \#_{a_1^J}(f',e) \mathrm{P}(f_1^J, a_1^J|e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I)} \right] + D} \qquad (6.45)$$

6.5.1 Update rule for HMM transition probabilities

We also wish to update the HMM transition probabilities to discriminate between correct and incorrect alignments; hence we compute the partial derivatives with respect to these. Let $a_{i,i'}$ denote the transition probability $a(i|i', I)$. Then, for a single sentence pair, the partial derivative of $\log \mathcal{N}$ with respect to $a_{i,i'}$ is

$$\begin{aligned}
\frac{\partial}{\partial a_{i,i'}} \log \mathcal{N} &= \sum_{j=1}^{J} \frac{\partial}{\partial a_{i,i'}} \log a(\hat{a}_j|\hat{a}_{j-1}, I) \\
&= \sum_{j=1}^{J} \frac{1}{a_{i,i'}} \delta_i(a_j) \delta_{i'}(a_{j-1}) \\
&= \hat{\#}(i' \to i) \frac{1}{a_{i,i'}}, \qquad (6.46)
\end{aligned}$$

where $\hat{\#}(i' \to i)$ is the number of transitions from i' to i under the reference alignment $\hat{a}_1^J$. Similarly, define $\#_{a_1^J}(i' \to i)$ to be the number of transitions from i' to i under an arbitrary alignment a_1^J. We can then differentiate $\mathcal{D}$ as follows:

$$\begin{aligned}
\frac{\partial}{\partial a_{i,i'}}\mathcal{D} &= \sum_{a_1^J} \frac{\partial}{\partial a_{i,i'}} \left(\prod_{j=1}^{J} a(a_j|a_{j-1}, I) t(f_j|e_{a_j}) \right) \\
&= \sum_{a_1^J} \frac{\partial}{\partial a_{i,i'}} \left(\prod_{j=1}^{J} a(a_j|a_{j-1}, I) \right) \left(\prod_{j=1}^{J} t(f_j|e_{a_j}) \right) \\
&= \sum_{a_1^J} \left(\frac{\#_{a_1^J}(i' \to i)}{a_{i,i'}} \prod_{j=1}^{J} a(a_j|a_{j-1}, I) \right) \left(\prod_{j=1}^{J} t(f_j|e_{a_j}) \right) \\
&= \sum_{a_1^J} \frac{\#_{a_1^J}(i' \to i)}{a_{i,i'}} \prod_{j=1}^{J} a(a_j|a_{j-1}, I) t(f_j|e_{a_j}). \qquad (6.47)
\end{aligned}$$

Finally, we use the chain rule to calculate $\frac{\partial}{\partial a_{i,i'}} \log \mathcal{D}$:

$$\begin{aligned}
\frac{\partial}{\partial a_{i,i'}} \log \mathcal{D} &= \frac{1}{\mathcal{D}} \frac{\partial}{\partial a_{i,i'}} \mathcal{D} \\
&= \frac{\sum_{a_1^J} \frac{\#_{a_1^J}(i' \to i)}{a_{i,i'}} \prod_{j=1}^{J} a(a_j|a_{j-1}, I) t(f_j|e_{a_j})}{\sum_{a_1^J} \prod_{j=1}^{J} a(a_j|a_{j-1}, J) t(f_j|e_{a_j})} \\
&= \frac{1}{a_{i,i'}} \frac{\sum_{a_1^J} \#_{a_1^J}(i' \to i) \mathrm{P}(f_1^J, a_1^J|e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I)} \\
&= \frac{1}{a_{i,i'}} \sum_{a_1^J} \#_{a_1^J}(i' \to i) \mathrm{P}(a_1^J|e_1^I, f_1^J) \qquad (6.48) \\
&\quad \text{(since } \textstyle\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I) = \mathrm{P}(f_1^J|e_1^I) \text{ and } \frac{\mathrm{P}(f_1^J, a_1^J|e_1^I)}{\mathrm{P}(f_1^J|e_1^I)} = \mathrm{P}(a_1^J|e_1^I, f_1^J)\text{)}
\end{aligned}$$

The update rule for the HMM transition probabilities is then

$$\begin{aligned}
\hat{a}_{i,i'} &= \frac{a_{i,i'} \left(\frac{\partial \mathcal{F}}{\partial a_{i,i'}} + D \right)}{\sum_{i''} a_{i'',i'} \left(\frac{\partial \mathcal{F}}{\partial a_{i'',i'}} + D \right)} \\
&= \frac{\hat{\#}(i' \to i) - \sum_{a_1^J} \#_{a_1^J}(i' \to i) \mathrm{P}(a_1^J|e_1^I, f_1^J) + D a_{i,i'}}{\sum_{i''} \left[\hat{\#}(i' \to i'') - \sum_{a_1^J} \#_{a_1^J}(i' \to i'') \mathrm{P}(a_1^J|e_1^I, f_1^J) \right] + D} \qquad (6.49)
\end{aligned}$$

6.5.2 Use of statistics calculated using the forward-backward algorithm

We can re-write these expressions so it becomes clear that they can be calculated using the forward-backward algorithm. The sum in the numerator of the translation probability update

equation (6.45) is

$$\begin{aligned}
\frac{\sum_{a_1^J} \#_{a_1^J}(f,e)\mathrm{P}(f_1^J, a_1^J|e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I)} &= \frac{\sum_{a_1^J} \#_{a_1^J}(f,e)\mathrm{P}(f_1^J, a_1^J|e_1^I)}{\mathrm{P}(f_1^J|e_1^I)} \\
&= \sum_{a_1^J} \#_{a_1^J}(f,e)\mathrm{P}(a_1^J|f_1^J, e_1^I) \\
&= \sum_{a_1^J}\sum_{i=1}^{I}\sum_{j=1}^{J} \delta_{e,f}(e_i,f_j)\delta_i(a_j)\mathrm{P}(a_1^J|f_1^J, e_1^I) \\
&= \sum_{i=1}^{I}\sum_{j=1}^{J} \delta_{e,f}(e_i,f_j)\sum_{a_1^J} \delta_i(a_j)\mathrm{P}(a_1^J|f_1^J, e_1^I) \\
&= \sum_{i=1}^{I}\sum_{j=1}^{J} \delta_{e,f}(e_i,f_j)\mathrm{P}(a_j = i|f_1^J, e_1^I) \qquad (6.50)
\end{aligned}$$

The forward and backward probabilities used during EM training for HMMs are $\alpha_j(i) = \mathrm{P}(a_j = i, f_1^j|e_1^I)$ and $\beta_j(i) = \mathrm{P}(f_{j+1}^J|a_j = i, e_1^I)$ respectively. Now the posterior probability of aligning f_j to e_i is $\mathrm{P}(a_j = i|f_1^J, e_1^I)$ and can be written in terms of these probabilities:

$$\begin{aligned}
\mathrm{P}(a_j = i, f_1^J|e_1^I) &= \mathrm{P}(a_j = i, f_1^j|e_1^I)\mathrm{P}(f_{j+1}^J|a_j = i, f_1^j, e_1^I) \\
&= \mathrm{P}(a_j = i, f_1^j|e_1^I)\mathrm{P}(f_{j+1}^J|a_j = i, e_1^I) \\
&= \alpha_j(i)\beta_j(i). \qquad (6.51)
\end{aligned}$$

The overall probability of the target sentence being generated by the source sentence can be written in terms of the forward probabilities:

$$\begin{aligned}
\mathrm{P}(f_1^J|e_1^I) &= \sum_{i=1}^{I} \mathrm{P}(a_J = i, f_1^J|e_1^I) \\
&= \sum_{i=1}^{I} \alpha_J(i) \qquad (6.52)
\end{aligned}$$

Therefore

$$\mathrm{P}(a_j = i|f_1^J, e_1^I) = \frac{\alpha_j(i)\beta_j(i)}{\sum_{i=1}^{I} \alpha_J(i)}. \qquad (6.53)$$

These statistics are already computed during the forward-backward algorithm; hence we can use the statistics generated during standard EM training for MMIE training.

For the transition probabilities, we can similarly apply the assumptions of the HMM model to turn the quantity we want to find into something that can be expressed using the statistics calculated by the forward-backward algorithm. The probability of generating f_1^J from e_1^I with

f_{j-1} aligned to $e_{i'}$ and f_j aligned to e_i is

$$\begin{aligned}
& \mathrm{P}(a_j = i, a_{j-1} = i', f_1^J | e_1^I) \\
= \; & \mathrm{P}(a_{j-1} = i', f_1^{j-1} | e_1^I) \mathrm{P}(a_j = i | a_{j-1} = i', f_1^{j-1}, e_1^I) \\
& \times \mathrm{P}(f_j | a_j = i, a_{j-1} = i', f_1^{j-1}, e_1^I) \mathrm{P}(f_{j+1}^J | a_j = i, a_{j-1} = i', f_1^j, e_1^I) \\
= \; & \mathrm{P}(a_{j-1} = i', f_1^{j-1} | e_1^I) \mathrm{P}(a_j = i | a_{j-1} = i', I) \\
& \times \mathrm{P}(f_j | a_j = i, a_{j-1} = i', e_1^I) \mathrm{P}(f_{j+1}^J | a_j = i, e_1^I). \qquad (6.54) \\
& \qquad (6.55)
\end{aligned}$$

We apply the assumptions of model, namely that each alignment depends only on the previous alignment and each target word depends only on the source word to which it is aligned, to obtain

$$\mathrm{P}(a_j = i, a_{j-1} = i', f_1^J | e_1^I) = \alpha_{j-1}(i') a(i|i', I) t(f_j|e_i) \beta_j(i) \qquad (6.56)$$

Therefore

$$\mathrm{P}(a_j = i, a_{j-1} = i' | f_1^J, e_1^I) = \frac{\alpha_{j-1}(i') a(i|i', I) t(f_j|e_i) \beta_j(i)}{\sum_{i=1}^I \alpha_J(i)} \qquad (6.57)$$

and

$$\begin{aligned}
\sum_{a_1^J} \#_{a_1^J}(i' \to i) \mathrm{P}(a_1^J | f_1^J, e_1^I) &= \sum_{a_1^J} \sum_{j=1}^J \delta_{i,i'}(a_j, a_{j-1}) \mathrm{P}(a_1^J | f_1^J, e_1^I) \\
&= \sum_{j=1}^J \mathrm{P}(a_j = i, a_{j-1} = i' | f_1^J, e_1^I) \\
&= \sum_{j=1}^J \frac{\alpha_{j-1}(i') a(i|i', I) t(f_j|e_i) \beta_j(i)}{\sum_{i=1}^I \alpha_J(i)} \qquad (6.58)
\end{aligned}$$

The update equations for the translation probabilities and alignment probabilities respectively are then

$$\hat{t}_{f,e} = \frac{\hat{\#}(f,e) - \sum_{i=1}^I \sum_{j=1}^J \delta_{e,f}(e_i, f_j) \frac{\alpha_j(i)\beta_j(i)}{\sum_{i=1}^I \alpha_J(i)} + D t_{f,e}}{\sum_{f'} \left[\hat{\#}(f', e) - \sum_{i=1}^I \sum_{j=1}^J \delta_{e,f'}(e_i, f_j) \frac{\alpha_j(i)\beta_j(i)}{\sum_{i=1}^I \alpha_J(i)} \right] + D} \qquad (6.59)$$

$$\hat{a}_{i,i'} = \frac{\hat{\#}(i' \to i) - \sum_{j=1}^J \frac{\alpha_{j-1}(i') a(i|i', I) t(f_j|e_i) \beta_j(i)}{\sum_{i=1}^I \alpha_J(i)} + D a_{i,i'}}{\sum_{i''} \left[\hat{\#}(i' \to i'') - \sum_{j=1}^J \frac{\alpha_{j-1}(i') a(i''|i', I) t(f_j|e_i) \beta_j(i'')}{\sum_{i=1}^I \alpha_J(i'')} \right] + D} \qquad (6.60)$$

6.5.3 Training on the entire corpus of human-aligned data

Again we consider the entire training corpus rather than a single sentence. Define $\#_s^{a_1^J}(f,e)$ to be the number of times f is aligned to e under alignment a_1^J, and $\hat{\#}_s(f,e)$ to be the number of times f is aligned to e under the reference alignment $\hat{a}_1^J$ of sentence s. Define $\#_s^{a_1^J}(i' \to i)$

to be the number of transitions from i' to i under alignment a_1^J and $\hat{\#}_s(i' \to i)$ the number of transitions under the reference alignment.

$$\hat{t}_{f,e} = \frac{\sum_{s=1}^{S'} \left[\hat{\#}_s(f,e) - \frac{\sum_{a_1^J} \#_s^{a_1^J}(f,e) \mathrm{P}(f_1^J, a_1^J | e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J | e_1^I)} \right] + D t_{f,e}}{\sum_{f'} \sum_{s=1}^{S'} \left[\hat{\#}^s(f',e) - \frac{\sum_{a_1^J} \#_s^{a_1^J}(f',e) \mathrm{P}(f_1^J, a_1^J | e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J | e_1^I)} \right] + D} \tag{6.61}$$

For the transition probability, we must take into account I, the length of the source sentence. For simplicity, write $a(i|i', I) = a_{i,i',I}$.

$$\hat{a}_{i,i',I} = \frac{\sum_{s=1}^{S'} \delta_I(I^s) \left[\hat{\#}_s(i' \to i) - \frac{\sum_{a_1^J} \#_s^{a_1^J}(i' \to i) \mathrm{P}(f_1^J, a_1^J | e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J | e_1^I)} \right] + D a_{i,i',I}}{\sum_{i''} \sum_{s=1}^{S'} \delta_{I^s,I} \left[\hat{\#}_s(i' \to i'') - \frac{\sum_{a_1^J} \#_s^{a_1^J}(i' \to i'') \mathrm{P}(f_1^J, a_1^J | e_1^I)}{\sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J | e_1^I)} \right] + D} \tag{6.62}$$

As previously discussed, these statistics can be can be computed using a modified form of the forward-backward algorithm.

6.6 Practical issues in MMI training of alignment models

6.6.1 Alignments to NULL

An alignment to NULL is used when a target word is not a direct translation of any source word. The human alignments only include a link when there is judged to be a translational correspondence between the source and target word. This leaves a number of target words unaligned: we add links so that these are aligned to NULL, and train the translation probabilities accordingly. Figure 6.3 shows the addition of the NULL state to the source sentence and alignment of unaligned target words.

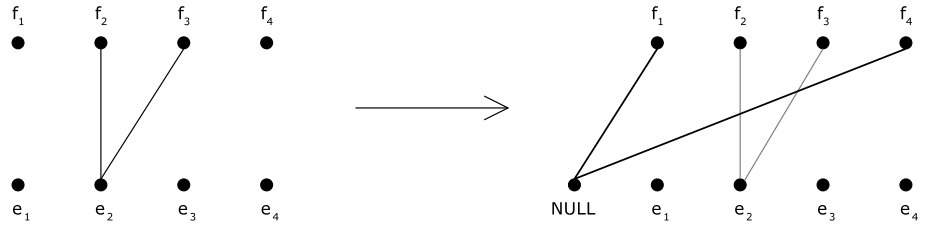


Figure 6.3: Transformation of manual alignments prior to MMI training: the original alignment (left) is transformed by adding links to NULL for each target word that is not already aligned

6.6.2 Many-to-one alignments

As discussed in Section 3.2, the models used assume each target word is generated by only one source word, with alignments not satisfying this property given zero probability. Many of the reference alignments contain many-to-one links, i.e. many source words link to the same target word, so we must consider how to use these in training. The update equations make use of the fact that the reference alignment is a valid alignment under the model and has an associated probability defined by the model, so we must modify the reference alignments to account for this.

For each target word with more than one link to a source word, a number of heuristic strategies are possible:

- Choose the first or last link
- Choose the link (i, j) with the highest translation probability $t(f_j|e_i)$
- Choose the link with the largest posterior probability, according to the model

6.6.2.1 Alternative optimisation criteria

To overcome the problems with many-to-one alignments, we consider an alternative quantity to optimise. Consider the links between sentences e_1^I and f_1^J as a set of pairs

$$A = \{(i, j) : 0 \leq i \leq I, 0 \leq j \leq J, e_i \text{ is linked to } f_j\} \tag{6.63}$$

and define the likelihood

$$l(A) = \prod_{(i,j)\in A} \mathrm{P}((i, j)|f_1^J, e_1^I), \tag{6.64}$$

where $\mathrm{P}((i, j)|f_1^J, e_1^I)$ is the posterior probability of e_i being aligned to f_j under an alignment model (the model used here does not matter; the same principle applies for all alignment models). This quantity is defined for all possible alignments between the sentences, even those that are not valid under any of our alignment models. In particular, many-to-one alignments are assigned likelihood in this way.

If A represents a valid alignment under Model 1 or 2, i.e. $A = \{(a_j, j) : 1 \leq j \leq J\}$ for some alignment sequence a_1^J, the likelihood $l(A)$ is equivalent to the posterior probability of the alignment a_1^J; hence optimisation of the posterior probability of the alignment is equivalent to optimisation of the likelihood in this case. For a proof of this, see B).

6.7 Summary

In this chapter, we derive an algorithm for using a development corpus of manually-aligned parallel text sentence pairs to improve the quality of Model 1 and word-to-word HMM alignment models. We present parameter re-estimation equations for discriminative training of the models using statistics derived from the manually-aligned data. A challenge of the method is setting a constant that governs the size of the change in model parameters and hence the speed of convergence; we describe a variety of techniques for determination of this constant. We also discuss methods of training when the reference alignment is not valid under the model.

Chapter 7
Evaluation of Discriminative Alignment Models

7.1 Introduction

This chapter investigates the properties of the discriminative training procedures for alignment models described in Chapter 6. Alignment models are trained using EM on a large parallel text in the usual way (with no additional information about the alignments between sentences); then MMIE is used to discriminatively train the models using a smaller word-aligned parallel text corpus. We investigate MMI training for the Model 1 and word-to-word HMM alignment models for Arabic-English and Chinese-English alignment.

Evaluation of the resulting models is carried out by measurement of AER against reference alignments; translation systems were initialised from the alignments output by the model to determine their effect on overall translation quality for Arabic to English translation.

7.2 Data sets and experimental procedure

The data sets used for training and testing were provided by LDC via the AGILE project and for the NIST machine translation evaluation; the languages used were Arabic and Chinese. Table A.2 gives a summary of the data sets used. The Arabic text was morphologically

analysed and tokenised using the MADA tagger described in Section 2.9.1. Where experiments were carried out to determine the effect of the models on translation quality, the *Tune.text.nw* and *SysCombTune.text.nw* sets provided by the AGILE project were used; these contain 1385 and 1500 sentences respectively of newswire data.

As discussed in Section 6.3.2, determination of the constant used in the update equations is difficult. We investigate two schemes for determining D, which are described in this section. Using an *adaptive constant* allows D to be varied depending on the result of the previous iteration of training, and a *word-specific constant* allows the speed of convergence of different source words to vary independently of each other.

Adaptive training

Inspired by algorithms used for training neural networks (Bishop, 1996), we use a global constant for all source words e and adapt the constant according to the outcome of the last iteration of training. First we set an initial constant D_0. Then, after each iteration of training, the constant is modified according to the posterior probability of the training data. Define L_k to be the value of the log posterior probability of the training data after iteration k of training:

$$L_k = \mathcal{F}|_{\theta=\theta_k} = \sum_{s=1}^{S'} \log \mathrm{P}_k(\hat{\mathbf{a}}_s|\mathbf{f}_s, \mathbf{e}_s), \tag{7.1}$$

where θ_k is the model parameters at iteration k and $\mathrm{P}_k(\hat{\mathbf{a}}_s|\mathbf{f}_s, \mathbf{e}_s)$ is the posterior probability of the reference alignment $\hat{\mathbf{a}}_s$ between sentences $\mathbf{f}_s$ and $\mathbf{e}_s$ at iteration k.

We update D as follows:

$$D_{k+1} = \begin{cases} D_k, & \text{if } L_k > L_{k-1}, \\ & \text{i.e. average per-sentence log probability} \\ & \text{of correct alignment increases} \\ 2D_k, & \text{if } L_k < L_{k-1}, \\ & \text{i.e. average per-sentence log probability} \\ & \text{of correct alignment decreases} \end{cases} \tag{7.2}$$

In the case where D_k is increased, we also begin training from the previous iteration; i.e. we discard the iteration where the previous value of D_k was too small to produce an increase in L_k. Training is halted according to two criteria:

- after a specified number of successful iterations, i.e. those where L_k is increased
- after a specified number of iterations where L_k does not increase: in this case we assume the training has converged so that we are sufficiently close to a local maximum.

Use of word-specific constants in the update equation

The constant D_e was determined separately for each source word e, using the formula

$$D_e = \max\left\{\max_f\left(-\frac{\partial \mathcal{F}}{\partial t_{f,e}}\right), 0\right\} + \epsilon. \tag{7.3}$$

This scheme has two advantages over using a single global constant for all source words. The first is speed of convergence: a satisfactory model should be reached more quickly because

we can take larger steps towards the local maximum. The second is an implementational issue: determining a single constant requires a complete pass through the training data prior to parameter estimation, evaluating $\frac{\partial \mathcal{F}}{\partial t_{f,e}}$ for each source word e and target word f, whereas the use of a word-specific constant allows us to set the constant one source word at a time when estimating the translation table.

7.3 Evaluation of discriminative Model 1

The AGILE v4.1 Arabic-English parallel text was used to train ten iterations of Model 1. The human-aligned subset of this data was used to carry out MMI training of Model 1 according to the update equations described in Section 6.4. The adaptive method of setting the constant D was used.

Figures 7.1 and 7.2 show the performance of the models in both translation directions as MMI training progresses. In each case, there is a significant increase in the average log posterior probability per sentence of the training data, accompanied by an increase in precision and recall, and a corresponding reduction in AER, when the models are used to align the data. For Arabic to English translation, precision and recall drop for the first two iterations of training even though the log likelihood of the data increases; however, these quantities increase as training progresses further. For Arabic to English, we achieve a drop in AER from 47.2 to 40.8, while for English to Arabic AER drops from 53.8 to 48.0.

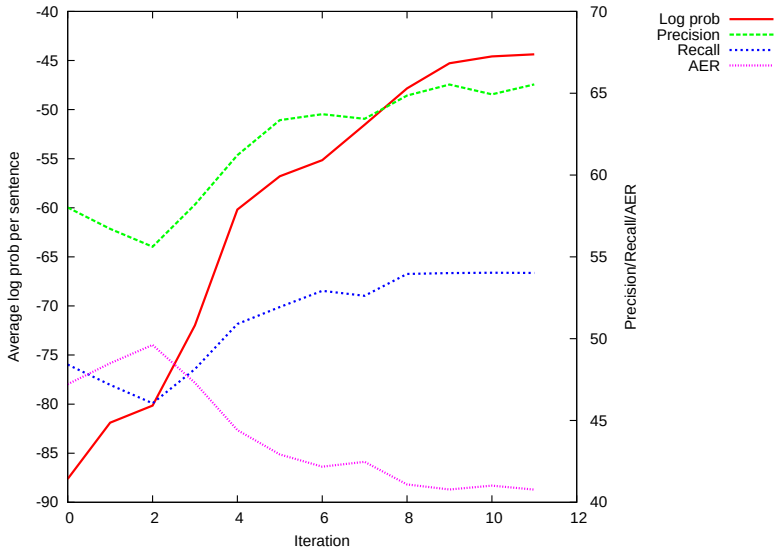


Figure 7.1: Graph showing how log posterior probability (Equation 6.9), precision, recall and AER vary as training progresses in Arabic to English direction (AGILE v4.1 data)

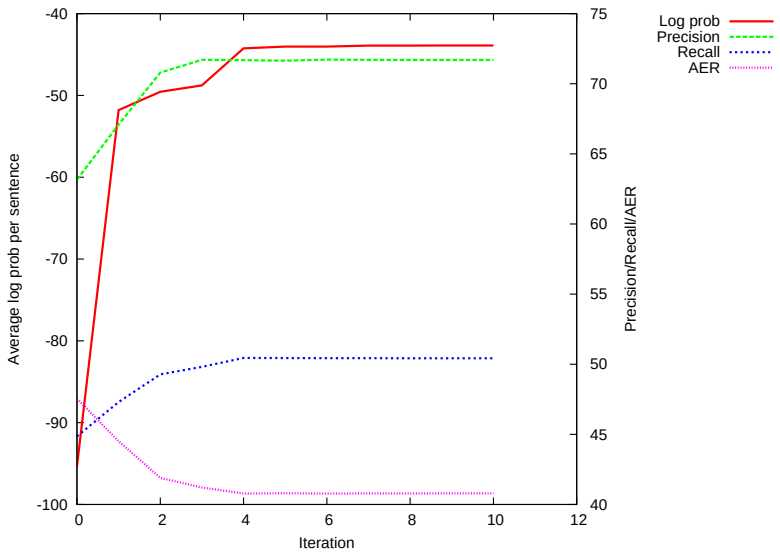


Figure 7.2: Graph showing how log posterior probability, precision, recall and AER vary as training progresses in English to Arabic direction (AGILE v4.1 data)

7.4 Evaluation of discriminative HMM alignment models

The Arabic AGILE v4.1 data was used to train ten iterations of Model 1 in each direction, followed by five iterations of Model 2 and five of HMM training. MMI training was then performed on the resulting HMM models, using three-quarters of the manually-aligned data with the remainder held out for evaluation of the model. An adaptive constant was used, initially set to $D = 100$. Training was halted when ten successive iterations of MMI failed to increase the likelihood of the training data.

Figure 7.3 shows the average per-sentence posterior log probability of the correct alignment for each iteration training in the Arabic to English direction, and the AER of the model on the training set. Figure 7.4 shows the same information for the English to Arabic direction. For both directions, the log probability and AER of the held-out data set were also measured. Log probability of the training set rises at every iteration due to the nature of the training (the iteration is discarded if we fail to increase the log probability). The AER and log probability of the test set are variable for the initial iterations of training, but their behaviour becomes more consistent as training proceeds and an increased value for the constant D causes smaller updates to be made to the model. Four fold cross-validation was carried out; the same patterns hold for the remaining partitions of the data.

Figure 7.5 shows the behaviour of the union of the alignments produced by the two models, by comparing its AER to those of the models in each translation direction separately. While the models in both directions show a substantial drop in AER, the decrease of the union is larger than either of the two directions individually: it drops from 31.4 to 22.6, a relative drop of 28%. This indicates that the quality of union of the alignments may increase more due to discriminative training than the alignments in either direction. If the same is true for the

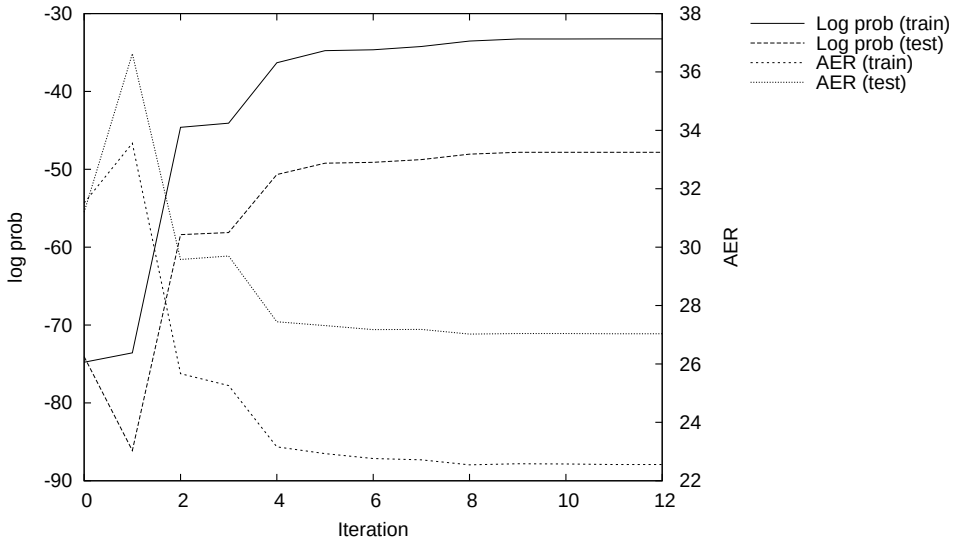


Figure 7.3: Graph showing how posterior log probability of the correct alignment and AER vary on training and test data sets as training progresses, for Arabic to English (AGILE v4.1 data)

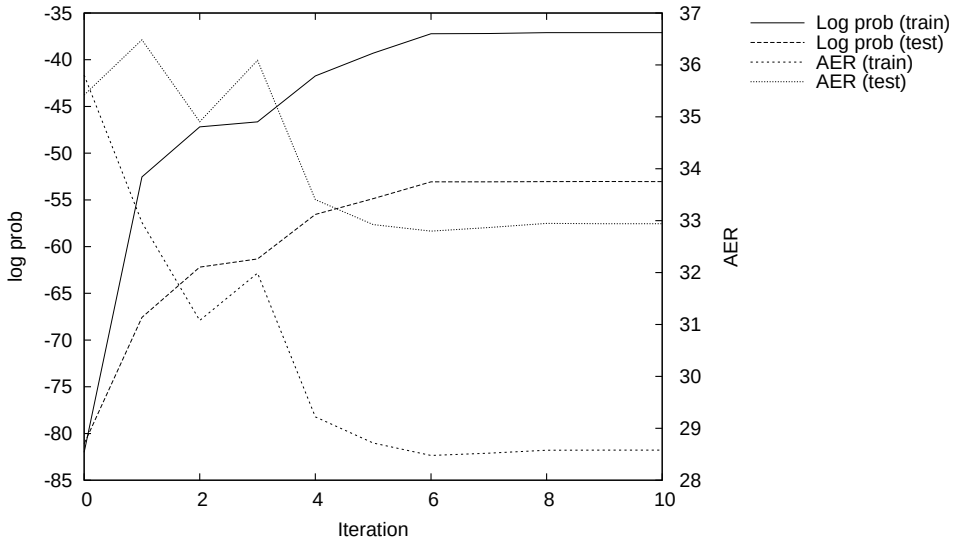


Figure 7.4: Graph showing how posterior log probability of the correct alignment and AER vary on training and test data sets as training progresses, for English to Arabic (AGILE v4.1 data)

entire parallel text, i.e. the quality of the union of the alignments improves significantly, the quality of translation may improve since the translation system is initialised from the union of the alignments.

One drawback of this training procedure is that the value of the global constant D needs to be very large in order that the none of the parameters estimated is negative; however, such

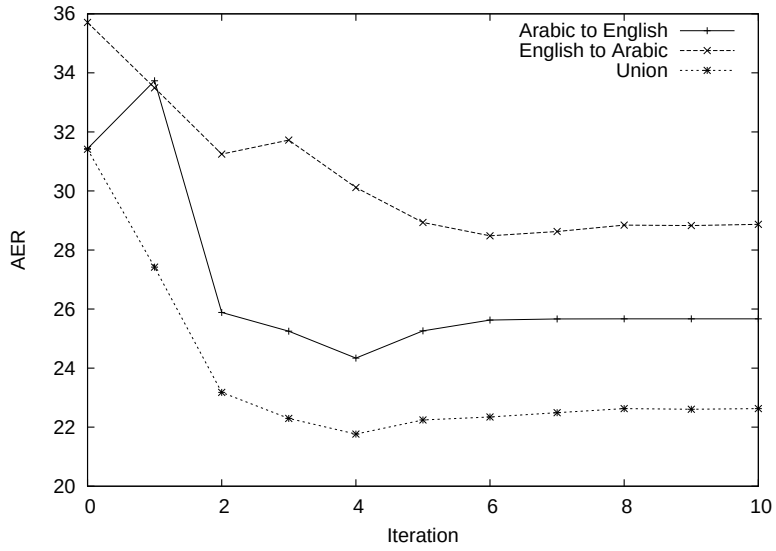


Figure 7.5: Graph showing how AER of models in each direction and their union varies as during MMI training (Arabic-English AGILE v4.1 data)

values of D result in prohibitively slow convergence which results in slow convergence. In these experiments, D was set so that many of the parameters (which are translation probabilities) were assigned negative values; we round these to a small positive value. In this way we achieve large gains in log probability and AER on the training data and test data, though this may harm the ability of the models to generalise to the parallel text corpus they are used to align.

The same training procedure was used to perform experiments for Chinese to English alignment. The results were similar to those obtained here and are not presented.

7.5 Word-dependent smoothing constant for HMM training

The NIST v1.1 Arabic-English data were used for ten iterations of Model 1 training, five iterations of Model 2 training and five iterations of HMM training, using the standard training procedure described in Section 3.12. MMI training was carried out on the HMM models using the manually aligned data, this time using a separate smoothing constant D_e specific to each source word. The model was trained on three-quarters of the manually aligned data and evaluated on the remaining quarter. The posterior probability of the correct alignment under the models was measured and the AER of alignments produced by the models was computed relative to the reference alignments.

Figures 7.6 and 7.7 show the performance of the models as training progresses. The log probability of the training data increases smoothly and the AER decreases during training. The same effects are seen on the held-out training data. Four-fold cross-validation was carried out to ensure the results the remained the same for the other three partitions of the data.

This method of setting the constant produces curves that are much smoother than those for the adaptive constant, but the speed of convergence is slower. The models have not converged

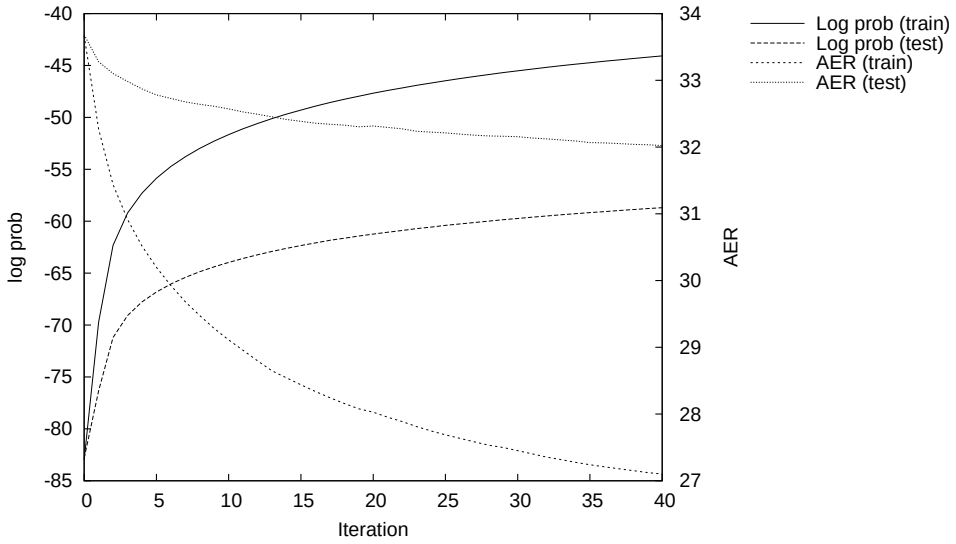


Figure 7.6: Variation of posterior log probability of the correct alignment and AER on training and test data sets as training progresses, for Arabic to English (NIST v1.1 data set)

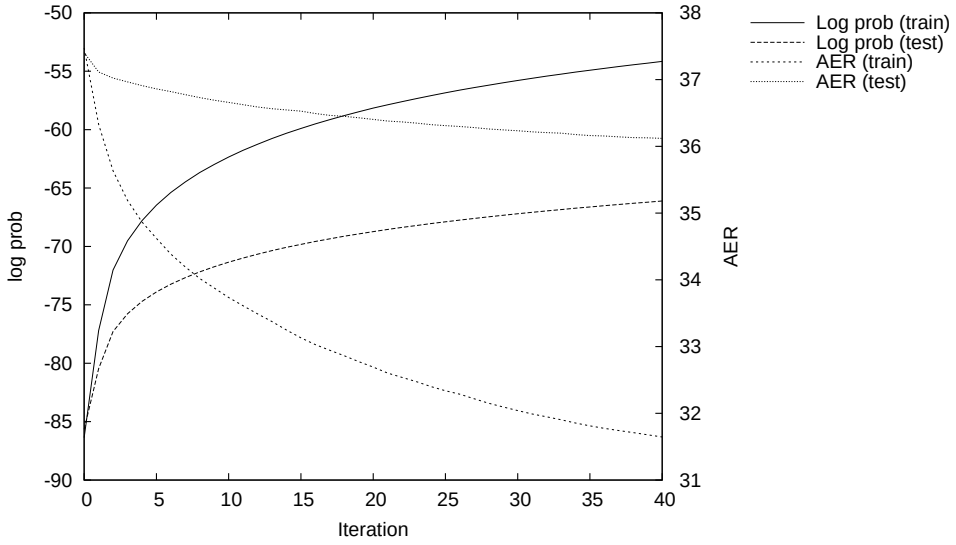


Figure 7.7: Variation of posterior log probability of the correct alignment and AER on training and test data sets as training progresses, for English to Arabic (NIST v1.1 data set)

after 40 iterations of training. They also do not yield the same improvement in AER over the baseline models. They do, however, avoid the estimation of negative probabilities and the erratic behaviour that occurs during the initial iterations of training, and it is hoped that these models generalise better to alignment of the full parallel text.

System	**Number of source phrases**	*Tune.text.nw* BLEU (TER)	*SysCombTune.text.nw* BLEU (TER)
Baseline HMM	61.7M	43.59 (**45.46**)	41.65 (46.65)
Iteration 5	62.7M	43.45 (45.57)	41.65 (**46.58**)
Iteration 10	63.5M	43.60 (45.50)	41.70 (**46.58**)
Iteration 15	63.9M	**43.64** (45.51)	**41.71** (**46.58**)
Iteration 20	64.2M	43.63 (45.53)	41.70 (46.59)

Table 7.1: BLEU *and TER scores of Arabic to English translation systems built using different iterations of MMI-trained HMM alignment models, on* Tune.text.nw *and* SysCombTune.text.nw *test sets*

7.6 Initialisation of translation systems from MMI-trained models

We have shown that MMI training of HMMs leads to improved alignment quality on the manually-aligned data. However, our main aim is to use the manual alignments as an additional source of information to improve the alignment models' performance on the entire parallel text. Machine translation systems built using alignments from the models should then produce higher quality translations. We now test the quality of Arabic to English translation systems built on the MMI-trained models.

A TTM system was trained on alignments output by iterations 5, 10, 15 and 20 of an HMM trained in the same way as those in Section 7.5, though this model was trained using the full manually-aligned data set. For comparison, a system was also trained on the alignments from the standard HMM from which the MMI models were initialised. Table 7.1 shows the BLEU score and TER of those systems on the *Tune.text.nw* and *SysCombTune.text.nw* test sets.

In each case, the performance of the system is almost identical to the baseline, indicating that the improved alignment quality on the manually-aligned data does not lead to an improvement in translation performance. It does however lead to a difference in the number of source phrases extracted by the translation system from the alignments: the number of alignment links output by the models decreases and the number of the source phrases extracted increases as MMI training progresses.

The use of MMI-trained HMMs does not appear to improve the translation quality over standard HMMs when the alignments from these models are used to initialise a machine translation system. There are several possible reasons for this. The first is that there is a mismatch between the manually-aligned data and the rest of the training data. Although the alignment quality for the manually aligned data is improved, the remainder of the data is sufficiently different that the improved model may not lead to better alignments on the whole data set.

The second is that the manually-aligned data set does not cover a large enough proportion of the words in the full parallel text. The translation probability distributions for source words that don't appear in the manually-aligned data are unchanged, so it is possible that the alignment model does not change much overall. There are 400k words in the Arabic vocabulary, of which 47k appear in the manually-aligned text. For English, there are 296k

words in the vocabulary, of which 25k appear in the manually-aligned text. This may not be enough to make a significant difference to the translation performance.

Finally, it may simply be that improved alignment quality does not lead to improved translation in this case. There has been other work which reports a similar outcome, i.e. that a reduction in AER of alignment does not lead to an increase in BLEU score of translations (Ittycheriah and Roukos, 2005; Vilar et al., 2006).

7.7 Conclusions and future work

We have shown that MMI training increases the quality of alignment for Model 1 and HMM alignment models for Arabic to English and English to Arabic generation. A held-out data set shows that the reduction in AER generalises to data other than the training data. We have investigated whether the improved alignment quality leads to an improvement in translation quality, using the MMI-trained HMMs to create alignments for initialisation of a machine translation system. It appears, however, that the improved quality of alignment does not lead to improved translations; possible reasons for this have been described.

7.7.1 Future work

While the translation system constructed from alignments of improved quality does not produce better translation hypotheses by itself, it may be useful in when combined with other systems, for example by MBR system combination; this approach is successful for the context-dependent models of Chapter 8 and may yield results here. Another potential avenue of research is to use the MMI-trained Model 1 to initialise Model 2 and allow training to continue as usual.

We have investigated the effect of carrying out discriminative training on a subset of manually-aligned data after the models have been trained on a larger set of parallel text. The EMD algorithm, which alternates EM training on a large parallel text with discriminative training on a manually aligned subset, has been shown to perform well (Fraser and Marcu, 2006b). A similar approach may bring improvements here. A discriminative step could tune the translation and alignment probabilities to encourage the correct alignments on the manually-aligned data. Links in the full parallel text assigned high probability by the discriminative model can then form a framework around which the remaining links in the sentences (which may be between words that were not seen in the manually aligned data) are arranged. The correct alignment of the word pair that has been indicated to be a match by the manual alignments would then cause the words around that alignment link to be aligned correctly as well.

CHAPTER 8

Context-Dependent Alignment Models

8.1 Introduction

Previous work on statistical alignment modelling has not taken into account the source word context when determining translations of that word, i.e. the translation probability $t(f|e)$ is independent of the words to the left and right of e and the other words that occur in the source sentence e_1^I. It is intuitive that a word in one context, with a particular part-of-speech and particular words surrounding it, may translate differently when in a different context. For example, the English noun *badger* translates to French as *blaireau*, while *badger* used as a transitive verb translates to *harceler*; knowledge of the part of speech of the word helps us to determine its translation. The English noun *bank* in a financial context would probably translate to French as *banque*, while it is likely to translate as *rive* when directly preceded by *river*. We aim to take advantage of the additional context information present in a sentence to provide a better estimate of the word's translation.

Due to the sheer number of possibilities for words surrounding a particular source word from which we can gather context information, one of the challenges of incorporating context information is maintaining computational tractability of estimation and alignment. We wish to avoid having to estimate too many different probability distributions, and we want enough training data for each distribution to estimate it accurately. Even the use of part-of-speech tags to define context rather than the surrounding words themselves is problematic, since this still leads to many possible contexts. We develop algorithms to overcome this difficulty.

For inspiration, we look to the field of Automatic Speech Recognition (ASR), where continuous-density HMMs are used for the acoustic model P($\mathbf{O}|\mathbf{e}$), where $\mathbf{e}$ is a sequence of words and $\mathbf{O}$ is the feature vector from speech associated with those words. The words are modelled as a sequence of phones that make up the word; the pronunciation of a given phone is affected by the preceding and subsequent phones (as well as other surrounding phones), so context-dependent phones are introduced to account for this effect.

The development of efficient estimation procedures for context-dependent acoustic models revolutionised ASR (Bahl et al., 1991; Young et al., 1994). Clustering is used extensively for improving parameter estimation of triphone (and higher order) acoustic models, enabling robust estimation of parameters and reducing the computation required for recognition. We adopt a similar approach to estimate context-dependent translation probabilities. We aim to improve the sentence-level alignments given by the alignment models by taking the source word context into account when estimating model parameters. We introduce binary decision trees (Breiman et al., 1984) for clustering contexts of source words to maintain computational tractability and avoid data sparsity.

This work builds on existing alignment models described in Chapter 3, from the original IBM models to the HMM alignment models used by Deng and Byrne (2005a). We focus on alignment with IBM Model 1 and HMMs. HMMs are commonly used to generate alignments from which state of the art SMT systems are built; improved quality of alignment should lead to improved phrases being extracted for use in translation and improved learning of rules for use in a hierarchical decoder. Model 1 is used as an intermediate step in the creation of more powerful alignment models, such as HMMs and further IBM models. In addition, it is used in SMT as a feature in Minimum Error Rate Training (Och et al., 2004) and for rescoring lattices of translation hypotheses (Blackwood et al., 2008a). It is also used for lexically-weighted phrase extraction (Costa-jussà and Fonollosa, 2005) and sentence-level segmentation of parallel text (Deng et al., 2007) prior to machine translation. The introduction of context-dependence to the model should improve its performance for all of these purposes.

8.2 Previous and related work

Bahl et al. (1991) introduce binary decision trees for clustering context-dependent phones for modelling by discrete HMMs. They construct a decision tree for each phone to specify the acoustic realisation of that phone in the context in which it appears. The tree is built top down using a greedy algorithm that splits each node using the best question at that node; we adopt a similar approach, constructing a decision tree for each word in the source language vocabulary to cluster the contexts of that word that occur. Bahl et al. (1989) use decision trees to generalise the n-gram language model used during natural language speech recognition.

Kannan et al. (1994) introduce a binary tree-growing procedure for clustering Gaussian models for triphone contexts based on the value of a likelihood ratio. For each allowable binary partition of the data, they choose between one of two hypotheses:

- H_0: the observations were generated from one distribution (that corresponds to the ML estimate for the parent node),
- H_1: the observations were generated from two different distributions (that correspond to the ML estimates for the child nodes).

The likelihood ratio is the ratio of the likelihood of the observations being generated from one distribution (H_0) to the likelihood of the observations in the partition being generated from two different distributions (H_1). The cluster is split if the likelihood of generating from two distributions is sufficiently greater than the likelihood of generating from one.

We now compare our model to existing work on the use of context-dependent models in machine translation. We divide the possible uses of context into themes corresponding to the stages of translation. The context information can be used during a *pre-processing* step prior to training and alignment by models: in this case, the alignment and translation models themselves do not need to be modified. The *alignment models* that produce links between parallel text sentences can also be improved by the addition of context. Finally, the *translation models* themselves can take contextual information into account.

8.2.1 Use of context in pre-processing

One approach to using linguistic knowledge to improve alignment and translation quality is to process the data to incorporate this knowledge prior to translation; an advantage of this approach is that the alignment and translation models themselves do not need to be modified and can be trained as usual.

Habash and Sadat (2006) perform morphological decomposition of Arabic words, such as splitting of prefixes and suffixes and the processing of the Arabic to make it more similar to English. This leads to gains in machine translation quality when systems are trained on parallel text containing the modified Arabic, and processing of Arabic text is carried out prior to translation into English. The impact of various pre-processing schemes is investigated, with the result that the choice of processing depends on the amount of training data. For small amounts of training data, more sophisticated morphological analysis helps to overcome data sparsity problems, while for large amounts of data less morphological analysis is needed.

Nießen and Ney (2001a) perform pre-processing of German and English parallel text prior to translation, transforming both the source and target sentences. They use the processing to address two specific issues in English to German and German to English translation. Firstly, some German verbs have the main part of the word and a detachable prefix that can be shifted to the end of the sentence; the prefixes are prepended to the main verb to enable easier correspondence with the single English verb, which is a single word. Secondly, the word order changes between declarative statements and questions so that translations of words in questions cannot be learnt from declarative sentences; the word order in questions is modified to harmonise it with declarative statements. In both cases, further processing is carried out after translation so hypothesis sentences have the correct verb form and word order.

8.2.2 Use of context in translation models

While we concentrate on the use of context for the alignment models that are used to produce word-aligned parallel text, others have integrated context information into the generation of hypotheses and decoding. The most obvious use of context in translation is the phrase-based translation models that segment the sentence into a sequence of phrases before translating the phrases individually. This makes use of the surrounding words to determine the translation of a particular word. In terms of word-based translation models, Berger et al. (1996) introduce maximum entropy models for machine translation, showing that the exponential model with the greatest entropy while remaining consistent with the constraints on it is the same as the

model which maximises the likelihood of the training data. They iteratively add features to the translation model, adding at each stage the one which gives the largest gain in the log likelihood of the training data. Context-based features are used for input language words, though these are simple: each one is an indicator function that checks for the presence of another word in particular positions, within a window of 3 words to the left and 3 words to the right of the current word.

Stroppa et al. (2007) consider context-informed features of phrases as components of the log-linear model during phrase-based translation. Features are split into word-based features that include the words to the left and right of the focus phrase, class-based features which consider the part-of-speech of the focus phrase and context words. Context information $\mathrm{CI}(\tilde{f}_k)$ and source phrase $\tilde{f}_k$, represented as vectors, are the input to the IGTree classifier (Daelemans et al., 1997) and the output phrases $\tilde{e}_k$ as classes to which the input must be assigned. This is a memory-based classifier that uses information gain to form a decision tree using training examples. Weighted class labels output by the classifier are normalised to form probability distribution $\mathrm{P}(\tilde{e}_k|\tilde{f}_k, \mathrm{CI}(\tilde{f}_k))$; this estimate is used as a feature for the log-linear model. The feature weights are optimised using MERT (Och, 2003). Stroppa et al. (2007) do not, however, address alignment.

Sarikaya et al. (2007) define a *context-dependent word* (CDW), which adds context information to a word, and use these to replace words as the basic units of translation, similar to the way in which context-dependent phones are used in Automated Speech Recognition (ASR). They investigate a number of ways of defining the context, which incorporate the part-of-speech of the current word, previous word and next word to varying degrees. The context-dependent words are applied in two ways. Firstly, they are used to build a language model for the output language that uses context-dependent words, which is then used for rescoring of n-best lists. Secondly, they are used as the units of translation so that the entire translation process is completed using CDWs in the output language (they translate from Iraqi Arabic to English and do not have a part-of-speech tagger for Iraqi Arabic). The rescoring produces gains in Bleu score using a phrase-based system, but the use of CDWs throughout translation fails to improve output quality.

Superficially, the work in this paper may appear similar to the work presented here, with the same type of context information being used, namely a combination of the parts-of-speech of the current, previous and next words. However, our work differs in that we use context in both input and output languages and use context information for improving alignments, whereas they find that building alignment models using CDWs does not help. A possible reason for this is that they use a stoplist of the 150 most common words to control model complexity, thereby preventing CDWs from being used for the most common words where they may be most useful for discriminating between different contexts. We also develop decision tree algorithms to cluster similar contexts so we do not need to use a stoplist. Sarikaya et al. (2007) apply context-dependent models during and post translation, whereas we apply only during alignment.

Other researchers have investigated the application of word sense disambiguation (WSD) to machine translation, the idea being that knowing the sense (i.e. meaning) of a word provides information about its translation. Results presented are mixed: Carpuat and Wu (2005) report that WSD does not improve translation quality for a Chinese to English system, while Vickrey et al. (2005) find WSD is able to fill blanks where words have been removed in translated sentences but do not incorporate their work into a decoder. Chan et al. (2007)

perform disambiguation using context information similar to that described here and report a statistically significant increase in BLEU score from incorporating WSD features into Hiero.

8.2.3 Use of context in alignment models

The HMM allows the translation model to be effectively split into two components: the *alignment component* models the transitions between the states of the HMM and the *translation component* controls the emission of target words from those states. Either or both of these can be modified to consider the context.

8.2.3.1 Target context

As discussed above, Deng and Byrne (2005a) allow the use of target language context by the use of bigram translation tables within the word-to-phrase HMM.

Nießen and Ney (2001b) use morphological information of the target word to estimate *hierarchical translation probabilities*, by considering derivatives of the base form of the word at different levels of detail. Starting with the base form in equivalence class c_0, they form a chain of equivalence classes $c_0, c_1, \ldots, c_n$, with each class becoming increasingly specialised, specifying more information about the part-of-speech, number etc; c_n is the complete morphological information about the word as it occurs in the sentence. For example, the hierarchy of equivalence classes for the German present tense first-person singular verb *ankomme*, derived from the base form *ankommen*[1] and translated into English as *arrive*, is as follows:

$$\begin{aligned} c_n &= \texttt{ankommen-V-IND-PRÄS-SG1} \\ c_{n-1} &= \texttt{ankommen-V-IND-PRÄS-SG} \\ c_{n-2} &= \texttt{ankommen-V-IND-PRÄS} \\ &\vdots \\ c_0 &= \texttt{ankommen} \end{aligned}$$

The probabilities for each class in the hierarchy are estimated separately, and the probabilities for all classes, $t_i(f|e)$ for $0 \leq i \leq n$, are interpolated to give a probability distribution:

$$t(f|e) = \lambda_0 t_0(f|e) + \ldots + \lambda_n t_n(f|e) \tag{8.1}$$

8.2.3.2 Source context

Och and Ney (2000a) reason that estimating the transition probabilities using only the size of the jump works well for a small corpus of training data; however a more refined model, where the transition probability is dependent on the source word, may be possible with a larger corpus. Using the words themselves would result in a very large number of alignment parameters, hence words are mapped to equivalence classes determined by the statistical clustering algorithm presented by Kneser and Ney (1991). This is similar to the classes used for the Model 4 distortion parameter (Brown et al., 1993), which controls the positioning of target phrases generated from the source sentence. Och and Ney (2000b) allow the HMM

[1] This has been pre-processed to combine the separable prefix *an* with the main part of the verb (Nießen and Ney, 2001a), as described in Section 8.2.1

transition probabilities away from a state i to depend on the word class $C(e_i)$ of the word in that state, so the probability becomes $a(a_j|a_{j-1}, C(e_{a_{j-1}}))$.

Varea et al. (2002) use a maximum entropy model of alignment, and incorporate source context information by testing for the presence of specific words within a window of the current source word, using similar features to Berger et al. (1996).

Popović and Ney (2004) use the base form of a source word and its part-of-speech tag during the estimation of word-to-word translation probabilities for IBM models and HMMs. They do not examine the surrounding words to give context-dependent estimates of translation probabilities.

8.2.3.3 Source and target context

Toutanova et al. (2002) use part-of-speech information in both the source and target languages with the aim of improving alignment models. A tag translation probability distribution $\mathrm{P}(d_j|c_i)$ is introduced into the model, for part-of-speech d_j of word f_j and part-of-speech c_i of e_i, so that prior knowledge of the translation of a word based on its part of speech can be incorporated. The translation model is re-written as:

$$\begin{aligned}
& \mathrm{P}(f_1^J, d_1^J, a_1^J|e_1^I, c_1^I) \\
= \; & \mathrm{P}(f_1^J, d_1^J, a_1^J|e_1^I, c_1^I) \\
= \; & \prod_{j=1}^{J} \mathrm{P}(f_j, d_j, a_j|f_1^{j-1}, d_1^{j-1}, a_1^{j-1}, e_1^I, c_1^I) \\
= \; & \prod_{j=1}^{J} \mathrm{P}(a_j|f_1^{j-1}, d_1^{j-1}, a_1^{j-1}, e_1^I, c_1^I)\mathrm{P}(d_j|a_1^j, d_1^{j-1}, f_1^{j-1}, e_1^I, c_1^I)\mathrm{P}(f_j|d_1^j, a_1^j, f_1^{j-1}, e_1^I, c_1^I),
\end{aligned}$$

where c_1^I and d_1^J are part-of-speech tag sequences for the source sentence e_1^I and target sentence f_1^J respectively, and it is assumed that the part-of-speech tagger in each language generates a single fixed sequence of tags for each sentence. The alignment is assumed to depend only on the previous alignment and source sentence length. Further, components of the distribution are replaced by approximations: the probability of a target word's tag depends only on the tag of the source word to which it is aligned, i.e. $\mathrm{P}(d_j|a_1^j, d_1^{j-1}, f_1^{j-1}, e_1^I, c_1^I) = \mathrm{P}(d_j|c_{a_j})$; the translation probability depends only on the two words being aligned, i.e. $\mathrm{P}(f_j|d_1^j, a_1^j, f_1^{j-1}, e_1^I, c_1^I) = t(f_j|e_{a_j})$. Overall, we have

$$\mathrm{P}(f_1^J, a_1^J|e_1^I, c_1^I) = \prod_{j=1}^{J} a(a_j|a_{j-1}, I)\mathrm{P}(d_j|c_{a_j})t(f_j|e_{a_j}) \tag{8.2}$$

The effect of this is to add an additional factor to the standard model that encourages words to align if their part-of-speech tags are compatible. Toutanova et al. (2002) also examine the use of part-of-speech information in the alignment component of the model, for better estimation of the HMM transition probabilities.

This approach differs from ours in several ways. Firstly, their tag translation probabilities do not depend on the surrounding words, whereas we find that many of the most commonly used questions for clustering depend on the following words. Secondly, their word-to-word translation probabilities are not changed by the context. There is also no consideration of any form of context other than part-of-speech.

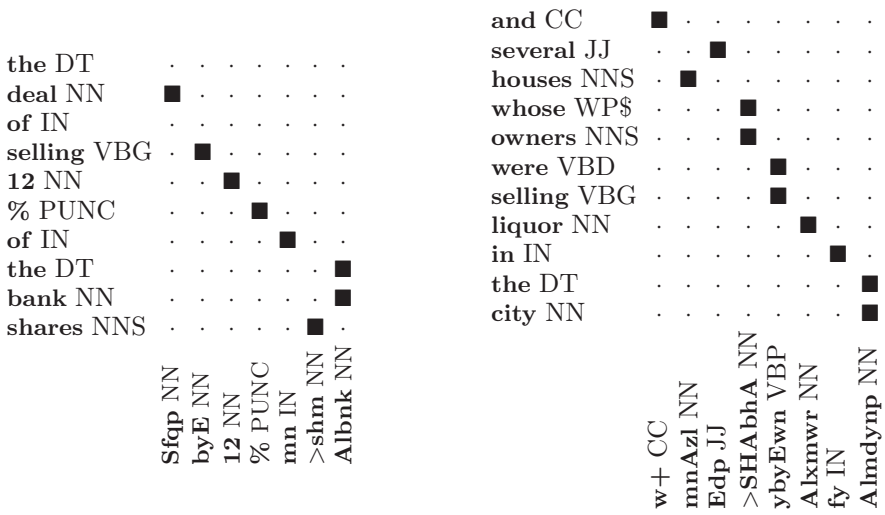


Figure 8.1: Alignment of the English selling *in different contexts. In the first, it is preceded by* of *and links to the infinitive of the Arabic verb* byE*; in the second, it is preceded by* were *and links to an inflected form of the same Arabic verb,* ybyEwn*.*

Both Toutanova et al. (2002) and Popović and Ney (2004) report gains in AER, but the effect of the models on translation quality is not investigated. Like these, we consider part-of-speech tags and morphological information about the words; however we also look at the surrounding words and evaluate the models using translation systems based on alignments generated by these models.

8.3 Use of source language context in alignment modelling

Consider the alignment of the target sentence $\mathbf{f} = f_1^J$ with the source sentence $\mathbf{e} = e_1^I$. Let $\mathbf{a} = a_1^J$ be the alignments of the target words to the source words. Let c_i be the context information of e_i for $i = 1, \ldots, I$. This context information can be any information about the word. We can consider part-of-speech, previous and next words, part-of-speech of previous and next words, and longer range context information. We can also consider the output of syntax parsers, morphological analysers and dependency parsers.

As an example of a situation where part-of-speech context is useful, Figure 8.1 shows the links of the English source word *selling* in two different contexts where it links to different words in Arabic, which are both forms of the same verb. The part-of-speech of the previous word is useful for discriminating between the two cases, whereas a context-independent model would assign the same probability to both Arabic words.

8.3.1 Addition of context-dependence to models

We follow Brown et al. (1993), but extend their modelling framework to include information about the source word from which a target word is emitted. We model the alignment process,

for a single sentence, as:

$$\mathrm{P}(f_1^J, a_1^J | e_1^I, c_1^I) = \mathrm{P}(J | e_1^I, c_1^I) \prod_{j=1}^{J} \mathrm{P}(a_j | f_1^{j-1}, a_1^{j-1}, e_1^I, c_1^I) \mathrm{P}(f_j | a_1^j, f_1^{j-1}, e_1^I, c_1^I) \tag{8.3}$$

For simplicity, we assume that the target sentence length, J, depends only on the length of the source sentence, I, i.e.

$$\mathrm{P}(J | e_1^I, c_1^I) = \epsilon(J|I). \tag{8.4}$$

This sentence length distribution is often ignored for convenience, since we align sentences of fixed length. We also assume that the target word depends only on the source word from which it is emitted, and the context of that source word, i.e.

$$\mathrm{P}(f_j | a_1^j, f_1^{j-1}, e_1^I, c_1^I, I) = t(f_j | e_{a_j}, c_{a_j}). \tag{8.5}$$

We introduce word-to-word translation tables that depend on the source language context for each word, i.e. the probability that e translates to f given e has context c is $t(f|e,c)$. This approach allows for multiple context sequences by viewing the context sequence as a hidden variable and marginalising over it to give $\mathrm{P}(f_1^J, a_1^J | e_1^I)$:

$$\begin{aligned} \mathrm{P}(f_1^J, a_1^J | e_1^I) &= \sum_{\tilde{c}_1^I} \mathrm{P}(f_1^J, a_1^J, \tilde{c}_1^I | e_1^I) \\ &= \sum_{\tilde{c}_1^I} \mathrm{P}(f_1^J, a_1^J | \tilde{c}_1^I, e_1^I) \mathrm{P}(\tilde{c}_1^I | e_1^I). \end{aligned} \tag{8.6}$$

However, we assume here that the context sequence c_1^I is given for a word sequence e_1^I, i.e. for an arbitrary context sequence $\tilde{c}_1^I$,

$$\mathrm{P}(\tilde{c}_1^I | e_1^I) = \begin{cases} 1, & \text{if } \tilde{c}_1^I = c_1^I \\ 0, & \text{otherwise} \end{cases} \tag{8.7}$$

Introducing this context-dependent translation into the standard models, we obtain the following sentence-to-sentence translation probabilities. For Model 1, ignoring the sentence length distribution,

$$\mathrm{P}_{\mathrm{M1}}(f_1^J, a_1^J | e_1^I, c_1^I) = \prod_{j=1}^{J} t(f_j | e_{a_j}, c_{a_j}). \tag{8.8}$$

For Model 2,

$$\mathrm{P}_{\mathrm{M2}}(f_1^J, a_1^J | e_1^I, c_1^I) = \prod_{j=1}^{J} a(a_j | j, I, J) t(f_j | e_{a_j}, c_{a_j}). \tag{8.9}$$

For the HMM model,

$$\mathrm{P}_{\mathrm{H}}(f_1^J, a_1^J | e_1^I, c_1^I) = \prod_{j=1}^{J} a(a_j | a_{j-1}, I) t(f_j | e_{a_j}, c_{a_j}). \tag{8.10}$$

8.3.2 Clustering of source contexts

Estimating translation probabilities separately for every possible context of a source word individually leads to problems with data sparsity and rapid growth of the translation table. We therefore wish to cluster source contexts which lead to similar probability distributions. Let C_e denote the set of all observed contexts of a particular source word e. A particular clustering is denoted

$$\mathcal{K}_e = \{K_{e,1}, \ldots, K_{e,N_e}\},$$

where $\mathcal{K}_e$ is a partition of C_e into N_e disjoint sets. We define a class membership function μ_e such that for any context c, $\mu_e(c)$ is the cluster containing c. We assume that all contexts in a cluster give rise to the same translation probability distribution for that source word, i.e. for a cluster K, $t(f|e,c) = t(f|e,c')$ for all contexts $c, c' \in K$ and all target words f; we write this shared translation probability as $t(f|e,K)$.

The Model 1 sentence translation probability for a given alignment (Equation 8.8) becomes

$$\mathrm{P}_{\mathrm{M1}}(f_1^J, a_1^J | e_1^I, c_1^I) = \frac{1}{(I+1)^J} \prod_{j=1}^{J} t(f_j | e_{a_j}, \mu_e(c_{a_j})). \tag{8.11}$$

For HMM alignment, we assume for now that the transition probabilities $a(a_j|a_{j-1}, I)$ are independent of the word contexts and the sentence translation probability is

$$\mathrm{P}_{\mathrm{H}}(f_1^J, a_1^J | e_1^I, c_1^I) = \prod_{j=1}^{J} a(a_j|a_{j-1}, I) t(f_j | e_{a_j}, \mu_e(c_{a_j})). \tag{8.12}$$

Section 8.3.3.1 describes how the context classes are determined by optimisation of the EM auxiliary function. Although the translation model is significantly more complex than that of context-independent models, once class membership is fixed, alignment and parameter estimation use the standard algorithms.

The original context-independent model comprises $|\mathcal{V}_{\mathrm{src}}|$ translation tables, each with up to $|\mathcal{V}_{\mathrm{tgt}}|$ entries (where $\mathcal{V}_{\mathrm{src}}$ and $\mathcal{V}_{\mathrm{tgt}}$ are the source and target language vocabularies), but the translation of $e \in \mathcal{V}_{\mathrm{src}}$ cannot vary depending on its context. Consideration of each context individually would need $\sum_{e \in \mathcal{V}_{\mathrm{src}}} |C_e|$ translation tables; the models would suffer from data sparsity and would grow too large. With clustering, we must estimate $\sum_{e \in \mathcal{V}_{\mathrm{src}}} N_e$ translation tables. During clustering, we try to find a balance between large model size and data sparsity problems, and inability to discriminate between translations of a word when it occurs in different contexts.

8.3.3 Parameter estimation for context-dependent models

We train the models using Expectation Maximisation (EM), using a parallel text training corpus comprising S sentences, $\{\mathbf{e}^{(s)}, \mathbf{f}^{(s)}\}_{s=1}^{S}$. We assume that the sentences of the training corpus are independent and calculate the likelihood of the the training data as

$$\begin{aligned} & \prod_{s=1}^{S} \sum_{\mathbf{a}} \mathrm{P}_\theta(\mathbf{f}^{(s)}, \mathbf{a}, J^{(s)} | \mathbf{e}^{(s)}, \mathbf{c}^{(s)}) \\ = & \prod_{s=1}^{S} \sum_{\mathbf{a}_1^J} \epsilon(J^{(s)}|I^{(s)}) \prod_{j=1}^{J^{(s)}} a(a_j|j, a_{j-1}, I^{(s)}, J^{(s)}) t(f_j^{(s)} | e_{a_j}^{(s)}, c_{a_j}^{(s)}) \end{aligned} \tag{8.13}$$

Here, $a(a_j|j, a_{j-1}, I^{(s)}, J^{(s)})$ is a general expression for the alignment component of the model that is valid for Model 1, Model 2 and the HMM alignment model, though each model makes additional conditional independence assumptions that simplify this expression; the derivation holds for all these models. We wish to estimate model parameters θ using the training data. Given model parameters θ', we estimate new parameters θ by maximisation of the EM auxiliary function

$$\begin{aligned} & \mathrm{E}\left[\log \prod_{s=1}^{S} \mathrm{P}(\mathbf{f}^{(s)}, \mathbf{a}, J^{(s)}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}; \theta) \middle| \mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}; \theta'\right] \\ = & \sum_s \sum_{\mathbf{a}} \mathrm{P}_{\theta'}(\mathbf{a}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \log \mathrm{P}_{\theta}(\mathbf{f}^{(s)}, \mathbf{a}, J^{(s)}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}). \end{aligned} \tag{8.14}$$

The log probability for a given parallel text sentence pair $\mathbf{e}^{(s)}, \mathbf{f}^{(s)}$ and alignment $\mathbf{a}$ is given by

$$\log \mathrm{P}_{\theta}(\mathbf{e}^{(s)}, \mathbf{a}, I^{(s)}|\mathbf{f}^{(s)}, \mathbf{c}^{(s)}) = \sum_{j=1}^{J^{(s)}} \log a(a_j|a_{j-1}, j, I^{(s)}, J^{(s)}) + \sum_{j=1}^{J^{(s)}} \log t(f_j^{(s)}|e_{a_j}^{(s)}, c_{a_j}^{(s)}). \tag{8.15}$$

Therefore the auxiliary function is given by

$$\sum_s \sum_{\mathbf{a}} \mathrm{P}_{\theta'}(\mathbf{a}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \left\{ \sum_{j=1}^{J^{(s)}} \log a(a_j|a_{j-1}, j, I^{(s)}, J^{(s)}) + \sum_{j=1}^{J^{(s)}} \log t(f_j^{(s)}|e_{a_j}^{(s)}, c_{a_j}^{(s)}) \right\} \tag{8.16}$$

We assume the sentence length distribution and alignment probabilities do not depend on the contexts of the source words; the alignment probabilities are estimated as in the context-independent case. Hence the relevant part of the auxiliary function is

$$\begin{aligned} & \sum_s \sum_{\mathbf{a}} \mathrm{P}_{\theta'}(\mathbf{a}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \sum_{j=1}^{J^{(s)}} \log t(f_j^{(s)}|e_{a_j}^{(s)}, c_{a_j}^{(s)}) \\ = & \sum_s \sum_{\mathbf{a}} \mathrm{P}_{\theta'}(\mathbf{a}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \sum_{j=1}^{J^{(s)}} \sum_{i=1}^{I^{(s)}} \delta_i(a_j) \log t(f_j^{(s)}|e_i^{(s)}, c_i^{(s)}) \\ & \quad \text{(introducing the sum over source sentence indices } i \text{ and an indicator function } \delta_i) \\ = & \sum_s \sum_{i=1}^{I^{(s)}} \sum_{j=1}^{J^{(s)}} \log t(f_j^{(s)}|e_i^{(s)}, c_i^{(s)}) \sum_{\mathbf{a}} \delta_i(a_j) \mathrm{P}_{\theta'}(\mathbf{a}|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \\ & \quad \text{(reordering the sums)} \\ = & \sum_s \sum_{i=1}^{I^{(s)}} \sum_{j=1}^{J^{(s)}} \log t(f_j^{(s)}|e_i^{(s)}, c_i^{(s)}) \mathrm{P}_{\theta'}(a_j = i|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \\ & \quad \text{(substituting the expression for the posterior probability that } f_j \text{ aligns to } i) \\ = & \sum_e \sum_f \sum_{c \in C_e} \sum_s \sum_{i=1}^{I^{(s)}} \sum_{j=1}^{J^{(s)}} \delta_e(e_i^{(s)}) \delta_f(f_j^{(s)}) \delta_c(c_i^{(s)}) \log t(f|e, c) \mathrm{P}_{\theta'}(a_j = i|\mathbf{e}^{(s)}, \mathbf{c}^{(s)}, \mathbf{f}^{(s)}) \\ & \quad \text{(introducing sum over } e, f, c \text{ and appropriate indicator functions)} \end{aligned} \tag{8.17}$$

$$= \sum_e \sum_f \sum_{c \in C_e} \left\{ \sum_s \sum_{i=1}^{I^{(s)}} \sum_{j=1}^{J^{(s)}} \delta_c(c_i^{(s)}) \delta_e(e_i^{(s)}) \delta_f(f_j^{(s)}) \mathrm{P}_{\theta'}(a_j = i | \mathbf{e}^{(s)}, \mathbf{f}^{(s)}, \mathbf{c}^{(s)}) \right\} \log t(f|e,c)$$

$$= \sum_e \sum_f \sum_{c \in C_e} \gamma'(f|e,c) \log t(f|e,c), \tag{8.18}$$

where

$$\gamma'(f|e,c) = \sum_s \sum_{i=1}^{I^{(s)}} \sum_{j=1}^{J^{(s)}} \delta_c(c_i^{(s)}) \delta_e(e_i^{(s)}) \delta_f(f_j^{(s)}) \mathrm{P}_{\theta'}(a_j = i | \mathbf{e}^{(s)}, \mathbf{f}^{(s)}, \mathbf{c}^{(s)}) \tag{8.19}$$

Here γ' can be computed under Model 1, Model 2 or the HMM, and is calculated using the forward-backward algorithm for the HMM.

8.3.3.1 Parameter estimation with clustered contexts

Once the contexts have been clustered, parameter estimation is little more complex than the estimation of parameters for context-independent models. We can re-write the EM auxiliary function (Equation 8.18) in terms of the cluster-specific translation probabilities:

$$\begin{aligned} \sum_e \sum_f \sum_{c \in C_e} \gamma'(f|e,c) \log t(f|e,c) &= \sum_e \sum_f \sum_{l=1}^{|\mathcal{K}_e|} \sum_{c \in K_{e,l}} \gamma'(f|e,c) \log t(f|e,\mu_e(c)) \\ &= \sum_e \sum_f \sum_{K \in \mathcal{K}_e} \gamma'(f|e,K) \log t(f|e,K) \end{aligned} \tag{8.20}$$

where

$$\gamma'(f|e,K) = \sum_{c \in K} \gamma'(f|e,c) \tag{8.21}$$

Following the usual derivation, the EM update for the class-specific translation probabilities becomes

$$\hat{t}(f|e,K) = \frac{\gamma'(f|e,K)}{\sum_{f'} \gamma'(f'|e,K)}. \tag{8.22}$$

Standard EM training of context-independent models can be viewed as a special case of this, with every context of a source word grouped into a single cluster, i.e. $\mathcal{K}_e = \{C_e\}$. Another way to view these clustered context-dependent models is that contexts belonging to the same cluster are tied and share a common translation probability distribution, which is estimated from all training examples in which any of the contexts occur.

8.4 Decision trees for context clustering

We have so far looked at the benefits of using context-dependent translation tables and of clustering those contexts, but have not considered the best way to partition contexts.

The objective for each source word is to split the contexts into classes to maximise the likelihood of the training data. Since it is not feasible to maximise the likelihood of the observations directly, we maximise the expected log likelihood by considering the EM auxiliary function, in a similar manner to that used for modelling contextual variations of phones for ASR (Bahl et al., 1991; Singer and Ostendorf, 1996; Young et al., 1994). We perform divisive

clustering independently for each source word e, by building a binary decision tree which forms classes of contexts which maximise the EM auxiliary function. Questions for the tree are drawn from a set of questions concerning the context information of e:

$$\mathcal{Q}_e = \{q_1, q_2, \ldots\}. \tag{8.23}$$

We start with all the contexts grouped together at one root node. We then evaluate each question in turn, select the one that gives the largest gain in objective function when used to split the node, and split the node using that question. The process is then repeated for each leaf node of the tree until the tree has been grown sufficiently.

Let K be any set of contexts of e, and define

$$\begin{aligned} L(K) &= \sum_f \sum_{c \in K} \gamma'(f|e,c) \log \hat{t}(f|e,K) \\ &= \sum_f \sum_{c \in K} \gamma'(f|e,c) \log \frac{\sum_{c \in K} \gamma'(f|e,c)}{\sum_{f'} \sum_{c \in K} \gamma'(f'|e,c)}. \end{aligned} \tag{8.24}$$

This is the contribution to the EM auxiliary function of source word e occurring in the contexts of K. Let q be a binary question about the context of e, and consider the effect on the partial auxiliary function (Equation 8.20) of splitting K into two clusters using question q. Define K_q to be the set of contexts in K which answer 'yes' to q and $K_{\bar{q}}$ to be the contexts which answer 'no'. Define the *quality function* of a cluster K to be

$$\begin{aligned} Q_{e,q}(K) &= \sum_f \sum_{c \in K_q} \gamma'(f|e,c) \log \hat{t}(f|e,K_q) + \sum_f \sum_{c \in K_{\bar{q}}} \gamma'(f|e,c) \log \hat{t}(f|e,K_{\bar{q}}) \\ &= L(K_q) + L(K_{\bar{q}}) \end{aligned}$$

This quantity allows us to determine the change in our objective function associated with splitting K using question q. When the node is split using question q, the increase in objective function is given by

$$Q_{e,q}(K) - L(K) = L(K_{\bar{q}}) + L(K_q) - L(K); \tag{8.25}$$

we choose q to maximise this.

8.4.1 Growing the decision tree

In order to build the decision tree for e, we take the set of all contexts C_e as the initial cluster at the root node. We then find the question $\hat{q}$ such that $Q_{e,q}(C_e)$ is maximal, i.e.

$$\hat{q} = \operatorname*{argmax}_{q \in \mathcal{Q}} Q_{e,q}(C_e) \tag{8.26}$$

This splits C_e, so our decision tree now has two nodes. We iterate this process, at each iteration splitting (into two further nodes) the leaf node that leads to the greatest increase in objective function. This leads to a greedy search to optimise the EM auxiliary function over possible state clusterings.

8.4.2 Complexity controls

In order to control the growth of the tree, we put in place stopping criteria that determine when we should stop growing the tree. We define two thresholds, similar to those used by HTK (Odell et al., 1995):

- *Improvement threshold*: T_{imp} is the minimum improvement in objective function required for a node to be split; without it, we would continue splitting nodes until each contained only one context, even though doing so would cause data sparsity problems.
- *Occupancy threshold*: T_{occ} is the minimum occupancy of a node, based on how often the contexts at that node occur in the training data; we want to ensure that there are enough examples of a context in the training data to estimate accurately the translation probability distribution for that cluster.

For each leaf node l, we consider the set of contexts K_l at that node and the effect of splitting them using each $q \in \mathcal{Q}_e$, to form $K_{l,q}$ and $K_{l,\bar{q}}$. We find the question q_l that, when used to split K_l, produces the largest gain in objective function:

$$\begin{aligned} q_l &= \operatorname*{argmax}_{q \in \mathcal{Q}_e} [L(K_{l,q}) + L(K_{l,\bar{q}}) - L(K_l)] \\ &= \operatorname*{argmax}_{q \in \mathcal{Q}_e} [L(K_{l,q}) + L(K_{l,\bar{q}})] \\ &= \operatorname*{argmax}_{q \in \mathcal{Q}_e} Q_{e,q}(K) \end{aligned} \tag{8.27}$$

We then find the leaf node among all leaf nodes of the tree for which splitting gives the largest improvement:

$$\hat{l} = \operatorname*{argmax}_{l} [L(K_{l,q_l}) + L(K_{l,\bar{q}_l}) - L(K_l)] \tag{8.28}$$

We split the contexts at node into two parts, creating two new leaf nodes as child nodes of $\hat{l}$, if the following criteria are both satisfied at that node:

- The objective function increases sufficiently:

$$L(K_{l,q_l}) + L(K_{l,\bar{q}_l}) - L(K_l) > T_{\text{imp}}$$

- The occupancy threshold is exceeded for both child nodes:

$$\sum_f \sum_{c \in K_{l,q}} \gamma'(f|e,c) > T_{\text{occ}},$$
$$\text{and} \quad \sum_f \sum_{c \in K_{l,\bar{q}}} \gamma'(f|e,c) > T_{\text{occ}}$$

The algorithm is illustrated in Figure 8.2. We perform such clustering for every source word in the parallel text.

The naive implementation of the algorithm is computationally expensive and is time-consuming to use on large data sets, but this can be overcome. We do this by a number of different methods:

- **Parallelisation:** since the clustering is performed for each source word separately, the process can be parallelised so different source words are processed on different machines. The words are partitioned, the partitions being determined by the frequency of the words' occurrence in the training data so that clustering takes a similar time for each partition.
- **Question elimination:** if a question fails to satisfy the occupancy criteria when splitting a node, it will also fail for all descendant nodes, so it need not be considered. The question can then be eliminated from consideration for splitting child nodes.
- **Node occupancy constraints:** if a node's occupancy is less than twice the occupancy threshold T_{occ}, it will not be able to be split into two sufficiently large nodes by any question. Thus, setting a high occupancy threshold can reduce training time as well as prevent data sparsity.

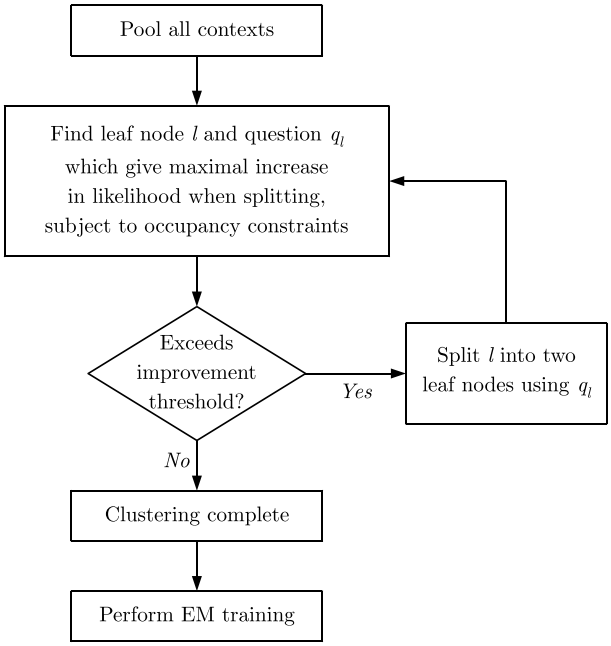
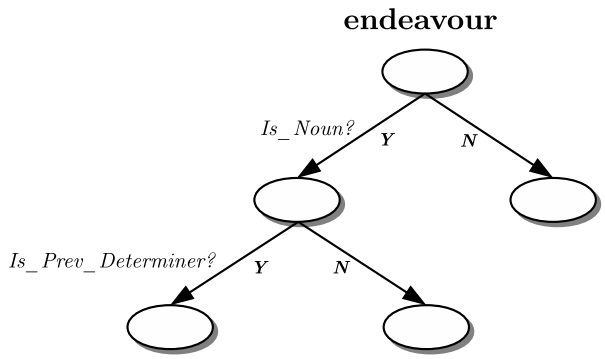


Figure 8.2: Decision tree clustering algorithm for word contexts.

An example decision tree for the English word *endeavour* is shown in Figure 8.3, along with some sentences in which it appears in the Arabic-English parallel text. In Sentence 1, the word occurs as a verb; in Sentences 2 and 3, it occurs as a noun, though in the first sentence the preceding word is a determiner. We can see that these words may translate differently in these different contexts. The decision tree supports this: the question with the greatest discriminatory effect asks if it is a noun, and asking whether the previous word is a determiner splits the uses of the noun.



1. `he assured that he would endeavour to provide an answer`
2. `the participation of the people is an integral part in this endeavour`
3. `they assign high priority to their cooperation in these areas of endeavour`

Figure 8.3: Decision tree for the English word endeavour*, and example sentences where* endeavour *occurs in the parallel text*

8.5 Initialisation of context-dependent models

The translation tables of the context-dependent models are initialised from those of the context-independent models. We begin by initialising the translation probability distribution of source word e with context c to be identical to that of the context-dependent source word e, i.e. we clone the context-independent distribution for each context so that

$$t(\cdot|e,c) = t(\cdot|e).$$

We then accumulate counts $\gamma'(f|e,c)$ over the training data using standard algorithms for EM, and perform clustering using these counts. The fact that we are directly optimising the EM auxiliary function allows us to perform the clustering and the first iteration of EM training simultaneously. We use the mapping μ_e to map each context to its cluster: initially, the distribution for each cluster is the same as the context-independent distribution, i.e. $\mu_e(c) = C_e$. This mapping enables us to avoid storing multiple copies of the same translation table.

The transition probabilities of the context-dependent HMM are not dependent on context and are initialised directly from those of the context-independent HMM.

8.6 Questions for decision tree clustering

The algorithms presented in Section 8.3 allow for any information about the context of the source word to be considered. We have so far been deliberately vague as to the questions that may be asked, in order to emphasise the generality of the approach. Any question about the source sentence is permissible, provided we have a source of information that can answer it. We could consider general questions of the form *'Is the previous word x?'*, or *'Does word y occur within n words of this one?'*, or even *'Is w a first-person singular verb in the past tense, followed by a proper noun?'*. The aim is to use the training data to choose the questions

carefully, so that they split the contexts into clusters whose members all translate according to a common probability distribution.

In order to perform clustering, we must accumulate a count for each context possible given the set of questions. Adding a binary question independent of all the other questions doubles the number of counts we must accumulate, so questions must be chosen carefully to avoid making the model too complex. We must also ensure there remain enough contexts in each cluster to estimate the translation probability distribution for that context accurately.

One method of maintaining computational tractability is to restrict the questions to those concerning the part-of-speech tag assigned to the current, previous and next words. We do not ask questions about the identities of the words themselves. This part-of-speech context approach is discussed in Section 8.7. Provided attention is paid to the complexity of the models and the number of contexts considered, the surrounding words themselves can be used as context, rather than just their parts of speech. We investigate this lexical context in Section 8.8. The two approaches, as well as other context information, can be combined using multiple stages of clustering. This is discussed further in Section 8.9.

8.7 Part-of-speech context

Part-of-speech is a context feature that can be readily found with many languages due to the availability of part-of-speech taggers. Part-of-speech is also informative of how a word translates, so it is a natural feature for use in this framework. For each part-of-speech tag T, we ask the question *'Does w have tag T?'*. We also ask the same questions of the previous and next words in the source sentence. In addition, we manually group part-of-speech tags to ask more general questions. For example, the set of contexts which satisfies *'Is w a noun?'* contains those that satisfy *'Is w a proper noun?'* and *'Is w a singular or mass noun?'*.

This addition of context information to a word can be viewed in a similar way to a triphone acoustic model in ASR. For a source word x in the parallel text, we form the word-with-context

$$\textsc{pos}(x_{-1})-x\#\textsc{pos}(x)+\textsc{pos}(x_{+1}), \tag{8.29}$$

where POS denotes part-of-speech and x_{-1} and x_{+1} are the source words appearing before and after x respectively. Words at the beginning of a sentence have no previous word and are marked with a sentence start token rather than $\textsc{pos}(x_{-1})$; similarly, the final word of a sentence is marked with a sentence end token rather than $\textsc{pos}(x_{+1})$. The decision tree clustering for x begins with all contexts $w-x+y$ occurring in the parallel text pooled together, which are then successively divided into clusters using the question that maximises the objective function at each step.

8.8 Lexical context

So far, we have considered part-of-speech context only. The derivation presented allows for arbitrary questions about the source word, so there is the possibility of including a far greater variety of questions. We now look at using lexical context, i.e. using the surrounding words themselves rather than just their parts of speech.

Write $\mathcal{V}_{\text{src}}$ for the source language vocabulary. For a word x, we augment the word with the previous word w and next word y to give *word-with-context* $w-x+y$. The questions that can then be asked of each word are

- *Is the previous word w?* (for all words $w \in \mathcal{V}_{\text{src}}$)
- *Is the next word y?* (for all words $y \in \mathcal{V}_{\text{src}}$)

Naively implemented, this gives $2|\mathcal{V}_{\text{src}}|$ possible questions, each of which must be tested to determine its effect on splitting the contexts of x. Since there are hundreds of thousands of words in the vocabulary, this is clearly intractable. A first step to resolving this is to restrict the questions for each word x to only those involving contexts in which x actually occurs in training data. The questions can be further reduced by removing those questions that concern contexts occurring infrequently in the training data, since there is unlikely to be enough data to estimate parameters for these contexts accurately.

Write $\mathcal{Q}_{\text{prev}}$ for the set of questions asking about the previous word and $\mathcal{Q}_{\text{next}}$ for those asking about the next word, i.e.

$$\begin{aligned} \mathcal{Q}_{\text{prev}} &= \{\text{'Is the previous word } w\text{?'} : w \in \mathcal{V}_{\text{src}}\} \\ \mathcal{Q}_{\text{next}} &= \{\text{'Is the next word } y\text{?'} : y \in \mathcal{V}_{\text{src}}\} \end{aligned}$$

We can then use subsets of $\mathcal{Q}_{\text{prev}} \cup \mathcal{Q}_{\text{next}}$, the set of all questions about immediate lexical context, to incorporate lexical context into our models. For a question q asking about the presence of a word, write $w(q)$ for the word to which it refers: for example, if the question is $q =$ *'Is the previous word the?'*, then $w(q) =$ 'the'. For a source word x, we can find the questions concerning that word, and those that occur more than a given number of times:

$$\mathcal{Q}_{\text{prev}}(x) = \{q \in \mathcal{Q}_{\text{prev}} : w(q){-}x \text{ occurs in training data}\} \quad (8.30)$$

$$\mathcal{Q}^F_{\text{prev}}(x) = \{q \in \mathcal{Q}_{\text{prev}} : w(q){-}x \text{ occurs} \geq F \text{ times in training data}\} \quad (8.31)$$

$$\mathcal{Q}_{\text{next}}(x) = \{q \in \mathcal{Q}_{\text{next}} : x{+}w(q) \text{ occurs in training data}\} \quad (8.32)$$

$$\mathcal{Q}^F_{\text{next}}(x) = \{q \in \mathcal{Q}_{\text{next}} : x{+}w(q) \text{ occurs} \geq F \text{ times in training data}\} \quad (8.33)$$

We set a threshold T_{freq} and perform the decision tree clustering algorithm of Section 8.3 using question set $\mathcal{Q}^{T_{\text{freq}}}_{\text{prev}}(x) \cup \mathcal{Q}^{T_{\text{freq}}}_{\text{next}}(x)$. For common words that occur in many contexts, this can result in too many questions for the computation to be tractable. We apply a further threshold T_{maxq} on the maximum number of questions asked of x to restrict questions to those concerning the contexts that occur most frequently.

In summary, we can reduce the number of questions to a manageable level by allowing questions only for contexts that occur sufficiently often in the data, and limiting the number of questions that can be applied to any given source word.

8.8.1 Lexical context with stemming

It is possible that using individual words as lexical context may cause data sparsity problems. One method of reducing the number of possible contexts is to stem the words that are used to provide context. A simple method of stemming is to reduce the surrounding words to their first L letters and use those letters as context. This reduces the number of possible contexts, and has the effect of ensuring that all contexts defined by words with the same first L letters are assigned to the same cluster. More sophisticated methods of stemming and morphological analysis, such as those used by Nießen and Ney (2001b) can also be used.

8.9 Further applications

Ideally we would like to incorporate many different forms of source context information, selecting those which lead to the greatest increase in translation quality. However, it is infeasible to consider performing clustering for too many types of context at once. In order to perform clustering, counts $\gamma'(f|e,c)$ are accumulated for each target word f, source word e and source word context c that occurs in the training data. If there are too many possible contexts, it is impossible to accumulate all these counts. If we choose to include lexical information, for example, or increase the size of the context window, the number of possible contexts becomes too large, since each additional piece of context information is independent of all the others and we must keep a count for every possible value of every context feature.

One way of overcoming this is to perform the clustering for different context features independently and sequentially. For example, we can use part-of-speech information initially to split the contexts, group those contexts which have the same probability distribution and run clustering again using lexical information. At each iteration of clustering, we introduce new context information; but those contexts that have already been grouped in the previous stage can be counted together, significantly reducing the complexity. If we wish to use more context information, we can perform further cloning of models and clustering incorporating this information. We can then build a series of decision trees $T^{(1)}, T^{(2)}, \ldots, T^{(N)}$, where $T^{(n)}$ is an extension of $T^{(n-1)}$ for each n.

With this framework, it is possible to include dependencies beyond the neighbouring words: this section describes possible methods of doing this. These approaches can be combined to further improve the alignment models. We now discuss additional sources of context information that may be used:

- **Dependency parsing:** Rather than simply using the immediate window surrounding the word as context, we can use a dependency parser such as Malt (Nivre et al., 2007) to determine the sentence structure and add information about inter-word dependencies to the context information. Rather than conditioning the translation distribution on the previous word, we can condition on the head word of the dependency. We can then run a clustering algorithm on this as before.

- **Extension of context using existing clusters:** It seems logical that contexts that translate in similar ways will affect the translation of their surrounding words in similar ways. We can incorporate this information by performing clustering and then annotating the data for further clustering using the context clusters from the first iteration.

  We can perform initial clustering using word-with-context $x_{-1} - x + x_{+1}$ to produce decision tree $T^{(1)}$. The data can be re-annotated, replacing the original contexts by the clustered contexts, then we perform further clustering with contexts $T^{(1)}(x_{-2} - x_{-1} + x) - T^{(1)}(x_{-1} - x + x_{+1}) + T^{(1)}(x - x_{+1} + x_{+2})$.

- **Incremental addition of context:** We can perform initial clustering using words-with-context of the form $x_{-1} - x + x_{+1}$, We can then view the cluster of $x_{-1} - x + x_{+1}$ as a single entity and split these contexts further by creating a decision tree to cluster

$$x_{-2} - \bar{x} + x_{+2} \tag{8.34}$$

  where $\bar{x} = x_{-1} - x + x_{+1}$. This cannot be accomplished in one step due to the space needed to accumulate counts for all possible 5-tuples of words, since this would require counts

to be accumulated for each possible context. In the case of lexical context, this would be $|\mathcal{V}_{\text{src}}|^5$ separate contexts, each of which must have a count accumulated for each target word it could possibly be aligned to. If we work with the immediately preceding and following words then introduce additional tags to the context, we accumulate $|\mathcal{V}_{\text{src}}|^3$ counts for the first training and $|V_{S'}||\mathcal{V}_{\text{src}}|^2$ for the second, where $V_{S'}$ is the set of clusters from the first training.

- **Part-of-speech within a wider window:** First, we can build decision trees to cluster contexts with a window size of 3, i.e. just involving the parts of speech of the previous, current and next source words. Once these clusters have been determined, the data can be annotated again with the surrounding parts of speech and these new contexts clustered.

  We first find context classes for $\text{POS}(x_{-1})-x\#\text{POS}(x)+\text{POS}(x_{+1})$, then split these classes further using context

  $$\text{POS}x_{-2}-\bar{x}+\text{POS}(x_{+2}), \tag{8.35}$$

  where $\bar{x} = \text{POS}(x_{-1})-x\#\text{POS}(x)+\text{POS}(x_{+1})$.

- **Testing for existence of a given word within the sentence:** If we suspect that the presence of a given word y in a sentence may affect the translation of our source word e, we can introduce the question *'Does the sentence contain y?'* and accumulate counts for both the case where the sentence does contain y and where it doesn't. The decision tree clustering algorithm can then be used to determine whether this splitting increases the EM auxiliary function sufficiently to split the cluster, and whether this split is more beneficial than conditioning on the presence of any other word.

8.10 Summary

This chapter has introduced context-dependent Model 1 and HMM alignment models for alignment of parallel text, which use context information in the source language to improve estimates of word-to-word translation probabilities. Decision tree clustering methods have been introduced for grouping the contexts to avoid the models becoming too complex and prevent difficulties with infrequently occurring contexts in the training data. We build binary decision trees, directly optimising the EM auxiliary function. Once clustering has been completed, the well-known EM algorithm is used for parameter estimation.

Context information in both languages of the parallel is incorporated by using alignment models trained in opposite directions. The algorithms derived are general and can be applied to any type of source language context, including part of speech, surrounding words and intra-sentence dependencies. We have described the application of the algorithms to integrating part-of-speech and lexical context, and briefly discussed how multiple sources of context information may be combined. Finally, we examined previous work on the integration of context information into alignment and translation models. Part of this work was published in a paper at NAACL-HLT 2009 (Brunning et al., 2009).

Chapter 9
Analysis of Context-Dependent Alignment Models

9.1 Introduction

The previous chapter introduces the context-dependent modelling framework and discusses training of those models. In this chapter, we carry out experiments using different types of context for Arabic to English and Chinese to English translation. We evaluate the effectiveness of the models by measurement of alignment quality on a held-out manually-aligned data set, and by measurement of translation quality when the alignments are used to initialise a machine translation system.

Our primary aim is that our alignment models improve the performance of translation systems built using those models. However, it is computationally expensive and time-consuming to evaluate every possible change to the model by building a translation system, which must be tuned, and computing the Bleu score of a test set. Alignment quality is simple to evaluate by comparison with a corpus of manually aligned sentence pairs and has been shown to be a reasonable indicator of translation quality; hence we first evaluate our context-dependent alignment models using AER.

9.2 Experimental method and data sets

Our models were built using the MTTK toolkit (Deng and Byrne, 2005b) and decision tree clustering was implemented. Clustering was carried out on each word separately; therefore the process can be parallelised to enable thousands of decision trees to be built in a reasonable time. Our context-dependent (CD) Model 1 models trained on context-annotated data were compared to the baseline context-independent (CI) models trained on untagged data.

The standard training scheme was used for the context-independent models: ten iterations of Model 1 training starting from a uniform translation table, five iterations of Model 2 training initialised from the final iteration of Model 1, and five iterations of word-to-word HMM training initialised from the final iteration of Model 2 (Brown et al., 1993; Deng and Byrne, 2005b). Context-dependent models were either initialised from the tenth iteration of Model 1 or from the fifth iteration of the word-to-word HMM.

Alignment quality was evaluated by computing AER relative to the manual alignments. Although there is not always a strict correlation between translation quality measured by BLEU score and 1 - AER, as discussed in Chapter 3, we require a method of evaluating our new alignment models that does not involve the expense of building a translation system trained on alignments generated by those models. AER is usually a reasonably good indicator of translation quality, and we use it for this reason. Since the links supplied contain only 'sure' links and no 'possible' links, we use the following formula for computing AER given reference alignment links S and hypothesised alignment links A: $\text{AER} = 1 - \frac{2|S \cap A|}{|S|+|A|}$.

In our translation experiments, translation was carried out using the HiFST decoder (Iglesias et al., 2009a) and we measure the quality of the translation hypotheses using BLEU score against a corpus of reference translations.

9.2.1 Data sets

The TnT tagger (Brants, 2000) was used as distributed, with its model trained on the Wall Street Journal portion of the English Penn treebank, to obtain part-of-speech tags for the English side of the parallel text. Marcus et al. (1993) gives a complete list of part-of-speech tags produced. No morphological analysis is performed for English. See Table A.3 for a summary of the parallel text used for model training.

9.2.1.1 Arabic

The Arabic data were pre-processed using the MADA toolkit (Habash and Sadat, 2006) to perform morphological word decomposition and part-of-speech tagging, as described in Section 2.9.1. The alignment models are trained on this processed data, and the prefixes and suffixes are treated as words in their own right; in particular their contexts are examined and clustered.

The following data sets were used for training Arabic to English alignment models:

- NIST 08 non-UN: the data allowed for the NIST 08 Arabic-English evaluation[1], excluding the UN collections, comprising 300k parallel sentence pairs, a total of 8.4M words of Arabic and 9.5M words of English;

[1] `http://nist.gov/speech/tests/mt/2008`

- NIST 08: the complete data allowed for the NIST 08 machine translation evaluation.

Automatic word alignments were compared to a manually-aligned corpus made up of the IBM Arabic-English Word Alignment Corpus (Ittycheriah et al., 2006) and the word alignment corpora LDC2006E86 and LDC2006E93. This contains 28k parallel text sentences pairs: 667k words of Arabic and 790k words of English. The alignment links were modified to reflect the MADA tokenisation; after modification, there were 946k word-to-word alignment links.

9.2.1.2 Chinese

The following data sets were used for training models for experiments with Chinese to English translation:

- $\frac{1}{10}$NIST 08: 600k random parallel text sentences were taken from the newswire LDC collection allowed for NIST MT08, a total of 15.2M words of Chinese and 16.6M words of English;
- NIST 09: data allowed for the NIST 09 Chinese-English evaluation;
- AGILE P4: the complete Phase 4 Chinese-English data set for the AGILE project.

The Chinese text was tagged[1] using the MXPOST maximum-entropy part-of-speech tagging tool (Ratnaparkhi, 1996) trained on the Penn Chinese Treebank 5.1 (Xia, 2000); the English text was tagged using the TnT part-of-speech tagger (Brants, 2000) trained on the Wall Street Journal portion of the English Penn treebank.

Automatic word alignments were compared to a manually-aligned corpus of 11.6k sentences, provided with the AGILE v8.0.3 data. This is a total of 336k words of English and 277k words of Chinese.

9.2.1.3 French

Experiments were carried out with parallel text allowed for the ACL Workshop on Statistical Machine Translation (WMT) 10. The models were trained on a set made up of the Europarl, news commentary and UN corpora; we refer to this data set as WMT 10. No manual alignments were available, so evaluation of the context-dependent models was carried out by using them to produce aligned parallel text and training a translation system.

9.2.2 Part-of-speech tagging

To look at the effect of part-of-speech context, we need the source text to be annotated with part-of-speech tags. Due to differences between the languages and the taggers used, the tags vary between languages. For each part-of-speech tag T, we ask the question *'Does w have tag T?'*. We also ask the same questions of the previous and next source words in the source sentence, with additional questions to determine if the word occurs at the start or end of the sentence. We manually group part-of-speech tags to ask more general questions; the questions asked for English are shown in Table 9.1.

[1]Thanks to Yanjun Ma for his help in training the part-of-speech tagger

Question	Pattern
Is_Punctuation	‘‘|,|:|.|’’|(|)
Is_Noun	NN|NNP|NNPS|NNS
Is_Adverb	RB|RBR|RBS
Is_Verb	VB|VBD|VBG|VBN|VBP|VBZ
Is_Adjective	JJ|JJR|JJS
Is_Pronoun	PRP|PRP$

Table 9.1: General context questions for English.

In English, this gives a total of 152 distinct questions, each of which is considered when splitting a leaf node. The MADA part-of-speech tagger uses a reduced tag set, which produces a total of 68 distinct questions for Arabic; the Penn Chinese treebank uses a total of 32 tags, resulting in 98 questions. See C for lists of the part-of-speech tags output by the various taggers. The tags themselves are not highly important: we mainly need a method of distinguishing between contexts, and a set of questions to enable the clustering of contexts that translate in the same way.

In English, this gives a total of 152 distinct questions, each of which is considered when splitting a leaf node. The MADA part-of-speech tagger uses a reduced tag set, which produces a total of 68 distinct questions for Arabic; the Penn Chinese treebank uses a total of 32 tags, resulting in 98 questions.

9.2.3 Summary of experiments

In Chapter 8, we discussed a number of ways of integrating context information into Model 1 and HMM alignment. We can evaluate alignment models by looking at the quality of the alignments they produce or at the translation quality of the machine translation systems initialised from those alignments. In order to permit a thorough investigation into the properties of the context-dependent models, we split our experiments into the following parts:

- Evaluation of alignment quality for Arabic-English translation with Model 1 with part-of-speech context (Section 9.3)
- Evaluation of alignment quality for Arabic-English translation with word-to-word HMMs with part-of-speech context (Section 9.4)
- Evaluation of alignment quality for Chinese-English translation with word-to-word HMMs with part-of-speech context (Section 9.5)
- Full translation of Arabic, Chinese and French into English for a number of data sets (Section 9.6)

9.3 Alignment quality of Arabic-English part-of-speech context-dependent Model 1

We first investigate the effect of adding part-of-speech context information to Model 1. Context-dependent training was carried out in both translation directions. For Arabic to

English generation, the Arabic side of the parallel text is tagged and the English side remains untagged; we view the English words as being generated from the Arabic words and questions are asked about the context of the Arabic words to determine clusters for the translation table. For English to Arabic, the situation is reversed: we used tagged English text as the source language and untagged Arabic text, with morphological decomposition, as the target language.

The context-dependent models were initialised from the final iteration of CI Model 1. A decision tree was built to cluster the contexts and a further 10 iterations of training were carried out using the tagged words-with-context to produce context-dependent models (CD Model 1). The models were then evaluated using AER at each iteration. A number of improvement thresholds T_{imp} were tested, and performance compared to that of models obtained from further iterations of CI Model 1 training on the untagged data. An occupancy threshold of $T_{\text{occ}} = 0$ was used for all models.

In both alignment directions, the log probability of the training data increases during training. As we would expect, the training set likelihood increases as the threshold T_{imp} is reduced, allowing more clusters and closer fitting to the data. Figure 9.1 shows how the log probability of the training data increases as training progresses.

9.3.1 Variation of improvement threshold T_{imp}

We now look at the alignment quality of the models for varying improvement threshold. There are a number of methods for symmetrising alignments for use in machine translation (Koehn et al., 2003), but our MT system is built on the union of the two sets of alignments (Iglesias et al., 2009a). Therefore, we consider the AER of union of alignments in the two directions as well as those in each direction. Figure 9.2 shows the change in AER of the alignments in each direction, as well as the alignment formed by taking their union at corresponding thresholds and training iterations. In both directions, the context-dependent models outperform the context-independent models at all thresholds.

There is a trade-off between modelling the data accurately, which requires more clusters, and eliminating data sparsity problems, which generally requires fewer clusters. With a smaller number of large clusters, each cluster should contain contexts that occur frequently enough in the training data to estimate the translation probabilities accurately. Use of a smaller threshold T_{imp} leads to more clusters per word and an improved ability to fit to the data, but this can lead to reduced alignment quality if there is insufficient data to estimate the translation probability distribution accurately for each cluster.

For lower thresholds, we observe over-fitting and the AER rises after the second iteration of context-dependent training, similar to the behaviour seen by Och (2002). The effect of this over-fitting is not as evident in the union alignment, since the loss in recall for one alignment is compensated by the additional links from the other alignment. Setting the improvement threshold $T_{\text{imp}} = 0$ results in each context of a word having its own cluster, which leads to data sparsity problems.

The threshold also has an effect on the number of words for which the contexts are split into multiple clusters. Table 9.2 shows the percentage of words for which the contexts are split for CD Model 1 with varying improvement thresholds. For words where the contexts are not split, all the contexts remain in the same cluster and parameter estimation is exactly the same as for the unclustered context-independent models.

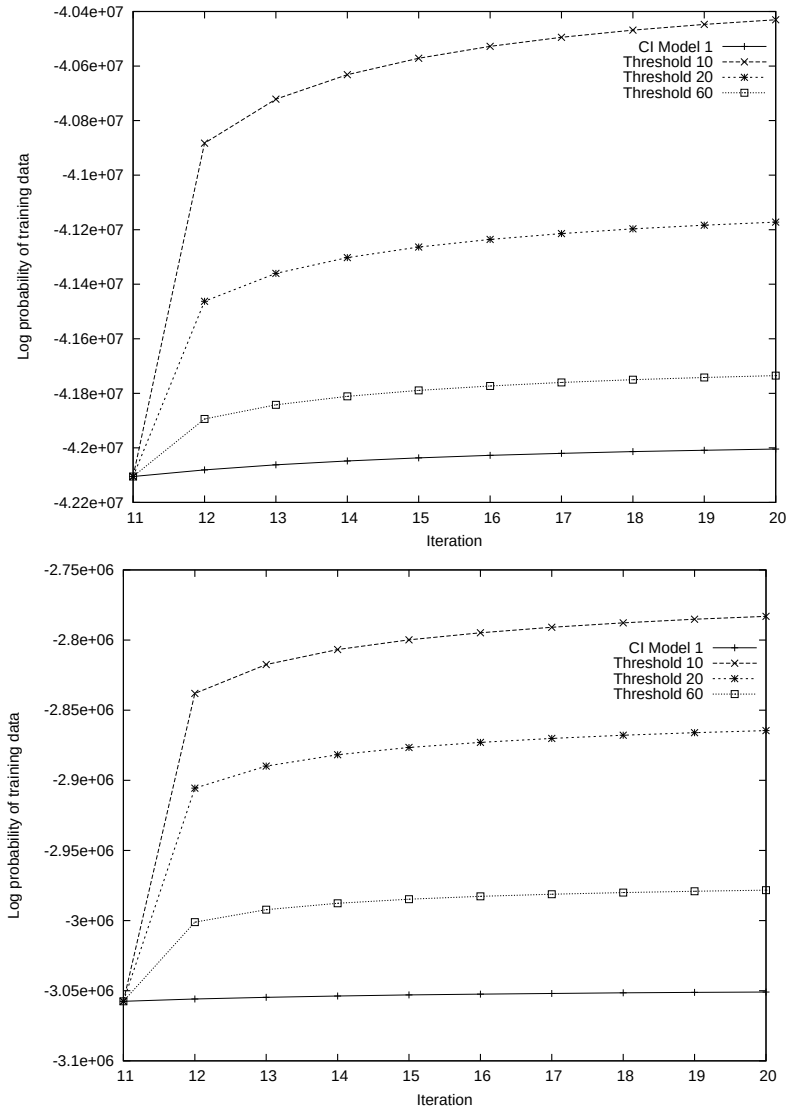
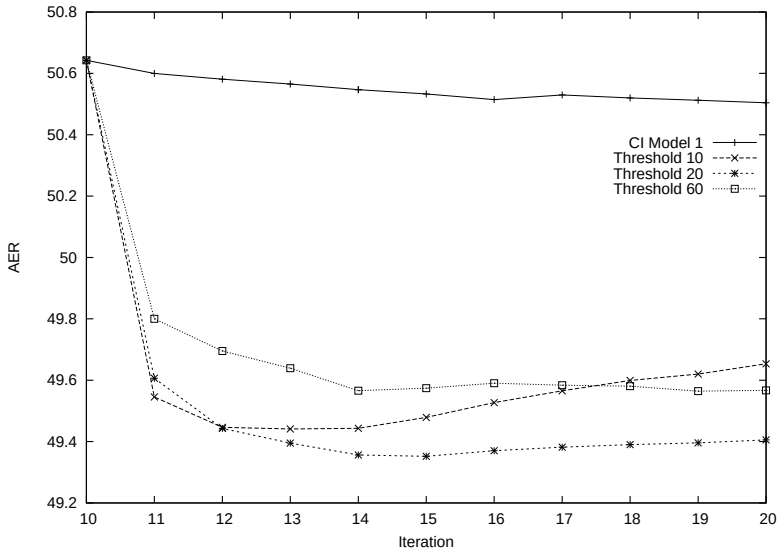


Figure 9.1: Increase in log probability of training data during training for varying T_{imp}, with Model 1, for Arabic to English (top) and English to Arabic (bottom)

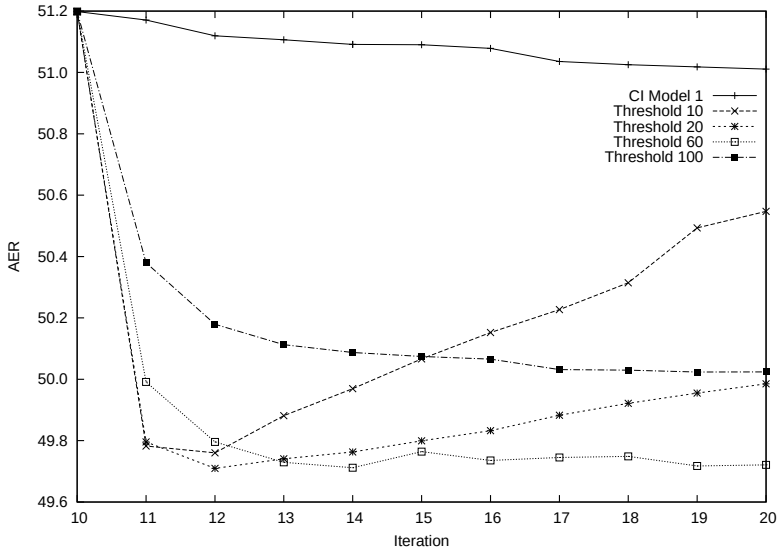
T_{imp}	Arabic-English (%)	English-Arabic (%)
10	30601 (25.33)	26011 (39.87)
20	11193 (9.27)	18365 (28.15)
40	1874 (1.55)	9104 (13.96)
60	724 (0.60)	4126 (6.33)
100	307 (0.25)	1128 (1.73)
140	204 (0.17)	629 (0.97)

Table 9.2: Words [number (percentage)] whose contexts are split by decision tree clustering, for varying improvement threshold T_{imp}

Arabic to English



English to Arabic



Union

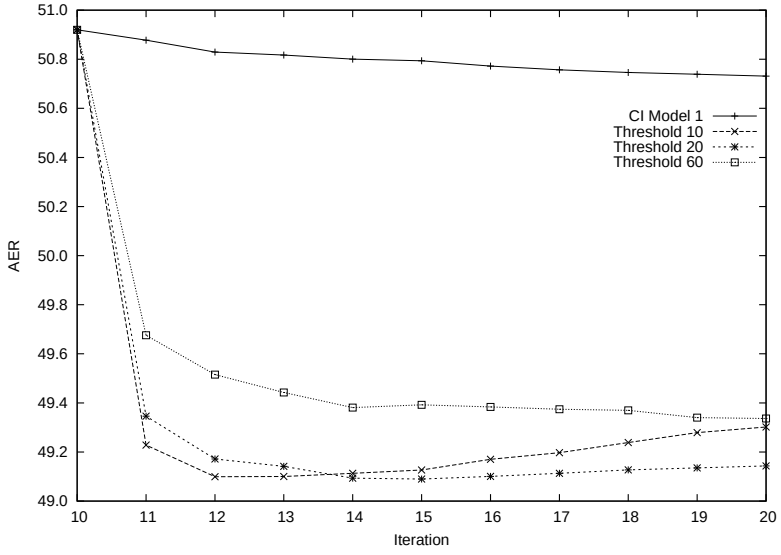


Figure 9.2: Variation of AER during Model 1 training for varying T_{imp}, for Arabic to English (top), English to Arabic (middle) and their union (bottom)

9.3.2 Analysis of frequently used questions

We would like to know the questions that are used most often by the decision tree clustering algorithm, as this gives an indication of the questions that have the greatest ability to discriminate between contexts which cause a word to translate differently. Table 9.3 shows the questions used most frequently at the root node of the decision tree when clustering contexts in English and Arabic, for occupancy threshold 10. The list shows the importance of the left and right contexts of the word in predicting its translation: almost all of the questions shown are not about the word itself, but concern the left or right context. Of the most common 50 questions used for the English context, 25 concern the previous word, 19 concern the next word, and only 6 concern the part-of-speech of the current word. For Arabic, of the most frequent 50 questions, 21 concern the previous word, 20 concern the next word and 9 the current word.

This does not mean that the part-of-speech of the current word is not important, rather that questions about part-of-speech are only needed for words that occur with multiple parts-of-speech. 72% of the English words have a single part-of-speech, so questions concerning part-of-speech are not needed for the majority of words. However, where a word does frequently occur with more than one part-of-speech, the part-of-speech plays a part in determining its translation.

9.4 Alignment quality of Arabic-English part-of-speech CD-HMMs

The addition of source word context to the alignment models has so far led to large reductions in AER for Model 1. However, the HMM alignment model is able to better represent the data due to its ability to allow alignments to depend on the previous alignment rather than just the word-to-word translation probabilities; hence performance of context-dependent Model 1 does not match that of context-independent HMMs trained on untagged data. We wish to incorporate context into the HMM; therefore we train context-dependent HMMs on tagged data.

We proceeded by training Model 1 and Model 2 in the usual way. From Model 2, context-independent HMMs (CI-HMMs) were initialised and trained for 5 iterations on the untagged data. Statistics were then gathered for clustering at various thresholds, after which 5 further EM iterations were performed with tagged data to produce context-dependent HMMs (CD-HMMs). The HMMs were trained in both the Arabic to English and the English to Arabic directions.

The log likelihood of the training set varies with T_{imp} in much the same way as for Model 1, increasing at each iteration, with greater likelihood at lower thresholds. The occupancy threshold has a similar effect, with the log probability increasing more for lower thresholds. Figure 9.3(a) shows the increase in log probability during training for different values of T_{occ} with T_{imp} fixed, for alignment with English as the source language and Arabic as the target language; Figure 9.3(b) shows the log probability for different values of T_{imp} with T_{occ} fixed. We see the same patterns in the English to Arabic translation direction.

The context-dependent models again lead to increased alignment quality, with all alignment models producing alignments with a lower AER than the context-independent HMM. The AER generally decreases for the first two iterations of training in both Arabic-to-English

English question	Frequency
Is_Prev_Determiner	2208
Is_Prev_Preposition	1912
Is_Next_Preposition	1908
Is_Prev_Adjective	1457
Is_Next_Noun_Singular_Mass	1110
Is_Prev_Noun_Singular_Mass	1066
Is_Prev_Coordinating_Conjunction	978
Is_Prev_Verb	966
Is_Next_Comma	791
Is_Next_Noun_Plural	714
Is_Next_Noun	657
Is_Prev_Comma	635
Is_Adjective	568
Is_Next_Sentence_Terminator	554
Is_Next_Punctuation	550
Is_Next_Determiner	542

Arabic question	Frequency
Is_Prev_Noun	3759
Is_Prev_Preposition	3002
Is_Next_Preposition	2869
Is_Next_Noun	2472
Is_Prev_Coordinating_Conjunction	1877
Is_Next_Punctuation	1672
Is_Prev_Noun_SingularMass	1583
Is_Next_Adjective_Adverb	1575
Is_Prev_Adjective_Adverb	1242
Is_Prev_Verb	1171
Is_Next_Coordinating_Conjunction	1072
Is_Noun	720
Is_Prev_Wh-Pronoun	685
Is_Prev_Punctuation	631
Is_Prev_Particle	546
Is_Prev_Noun_Proper	525

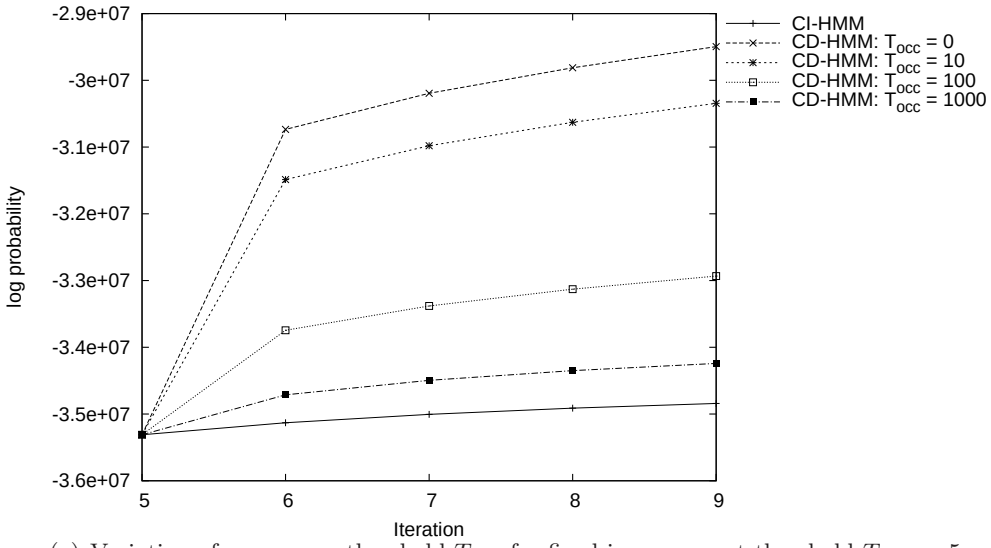
Table 9.3: Most frequent root node context questions for English (top) and Arabic (bottom)

and English-to-Arabic alignment; after this it starts to rise, a sign that the models are becoming overtrained. We aim to maximise the log probability of the training data during training; while we succeed in increasing the log probability of the training data at each iteration, this does not necessarily lead to an increase in alignment quality.

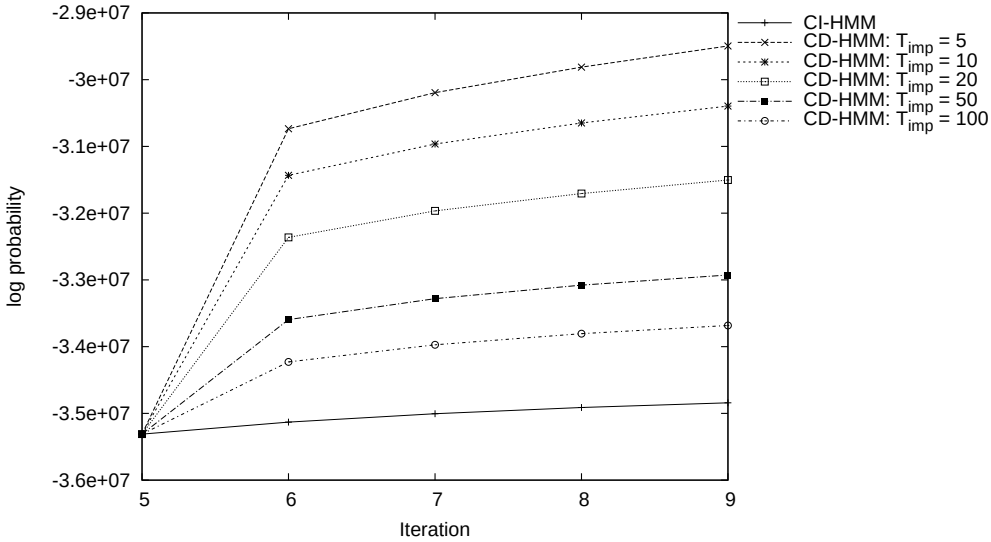
Analysis of the decision trees shows that the same questions are asked at the root nodes as for Model 1. This suggests that the context of a word affects its translation in the same way even with improved modelling of the alignment. Again the most commonly used questions concern the part-of-speech of the next and previous words.

9.4.1 Effect of varying improvement and occupancy thresholds

Figure 9.4 shows how the AER varies throughout training for different values of T_{occ}, with



(a) Variation of occupancy threshold T_{occ} for fixed improvement threshold $T_{\text{imp}} = 5$



(b) Variation of improvement threshold T_{imp} for fixed occupancy threshold $T_{\text{occ}} = 0$

Figure 9.3: Increase in log probability of training data for context-dependent HMM, for English to Arabic direction

T_{imp} fixed at 5, for both translation directions between English and Arabic. With $T_{\text{occ}} = 0$, there is no lower limit on the size of the clusters so a cluster may contain a context that appears only once in the training data, causing overtraining to occur and the AER to rise rapidly after the first iteration of context-dependent training. The effect of increasing T_{occ} is to prevent such unbalanced splits occurring, so that each context either has sufficient training data to estimate its probability distribution accurately or has its distribution smoothed by being clustered with similar contexts. A higher value of T_{occ} prevents overtraining occurring.

However, large values of T_{occ} can also prevent the model from fitting the data: if it is too high, it may not be possible to split the contexts at all, since insufficient examples are

present in the training data. This causes the benefits of context-dependence to be lost. We therefore aim to choose T_{occ} so that it is high enough to prevent data sparsity when estimating probabilities yet low enough that contexts can be split to model the data accurately. We can see the same trends with larger values of the improvement threshold, though they are less pronounced since it reduces the amount of splitting that can take place during clustering.

Figure 9.5 shows the variation in AER throughout training for different values of the improvement threshold T_{imp}, with T_{occ} fixed at zero. For small values of the threshold, the contexts can be split more easily during decision tree growth, but this can lead to overtraining. Setting the threshold too large results in all contexts of a word being assigned to the same cluster. We see the same trends with a non-zero occupancy threshold but the effects are less pronounced, as the occupancy threshold prevents leaf nodes being split when there are few training examples.

Like the union alignments from Model 1, the union alignments have a lower AER than the alignments in either direction. The improvement and occupancy thresholds have similar but complementary effects. Increasing either of them will generally lead to fewer clusters with more contexts in each cluster. The improvement threshold ensures that splitting a cluster results in sufficient gain in objective function, while the occupancy threshold ensures there are enough contexts that occur sufficiently often in the training data. In the latter case, we may have a small number of frequently occurring contexts or a larger number of relatively infrequent contexts. We aim to choose a combination of T_{occ} and T_{imp} so that the training data are modelled best.

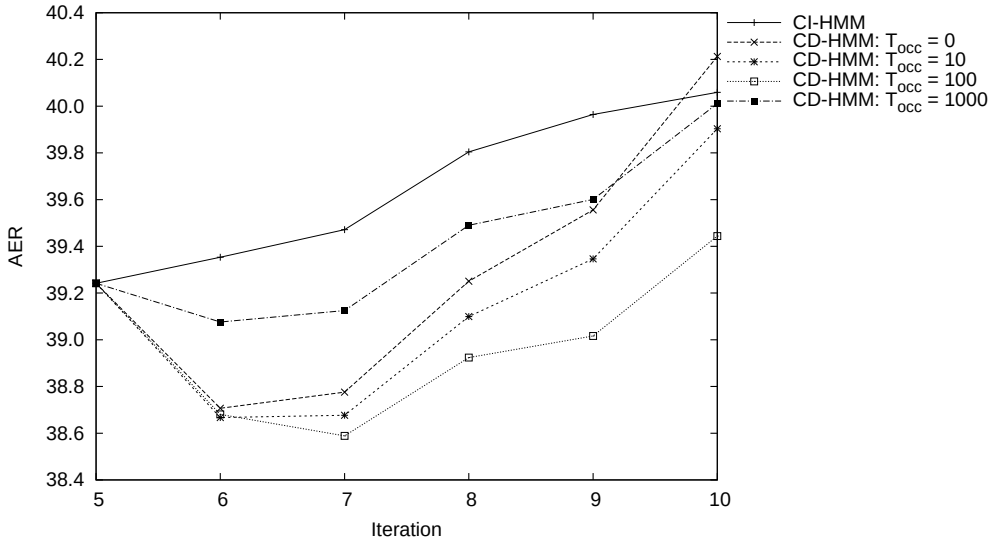
All of the context-dependent models examined here produce alignments with a lower AER than the context-independent HMM models, whose alignments have increasing AER as more iterations of training are carried out.

9.4.2 Alignment precision and recall

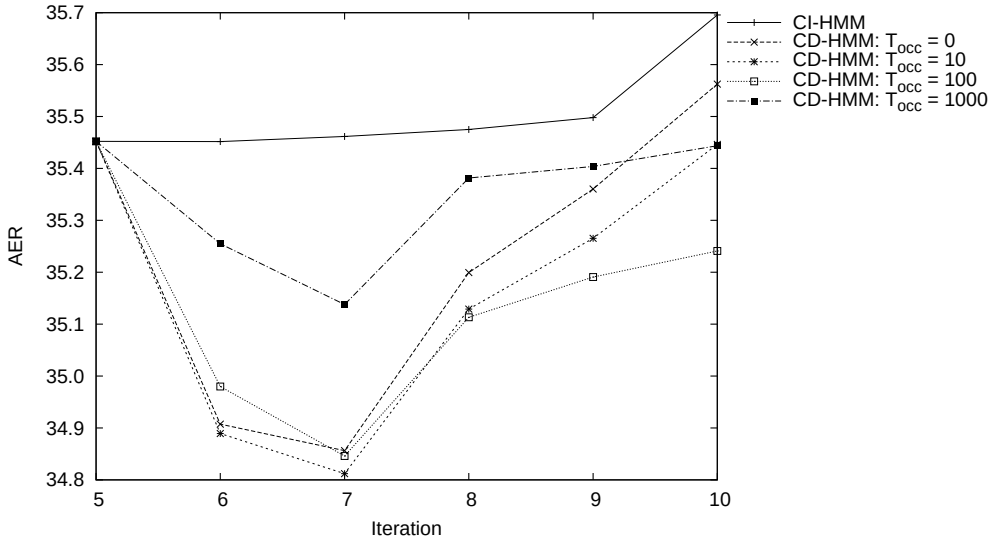
We can modify the null transition probability of the HMM models, p_0, to adjust the number of alignments to the null token, as described in Section 3.5. Where a target word is emitted from null, it is not included in the alignment links, so this target word is viewed as not being aligned to any source word. This affects the precision and recall: if a target word that was previously aligned correctly becomes aligned to the null token, precision and recall decrease, whereas if a target word that was previously incorrectly aligned becomes aligned to null, precision increases. The results reported above use $p_0 = 0.2$ for English-Arabic and $p_0 = 0.4$ for Arabic-English; we can tune these values to produce alignments with the lowest AER. Figure 9.6 shows precision-recall curves for the CD-HMMs compared to the CI-HMMs for both translation directions. For a given value of precision, the CD-HMM has higher recall; for a given value of recall, the CD-HMM has higher precision. This indicates that the context-dependent models are better at performing alignment, regardless of the p_0 parameters used.

An alternative method of evaluating the models using precision and recall is to extract alignment links by their posterior probability. By varying the probability threshold above which potential alignment links are included, we can produce precision/recall curves for our models. Figure 9.7 shows precision/recall curves for $T_{\text{occ}} = 100, T_{\text{imp}} = 10$ for both translation directions, with the posterior link probability threshold λ varying between 0.001 and 0.99; a small change in λ has a larger effect at the extremes of this range so more data points were taken in those areas. Again we can see that the curves of the context-dependent models lie entirely outside those of the context-independent models, indicating they produce better

English to Arabic



Arabic to English



Union

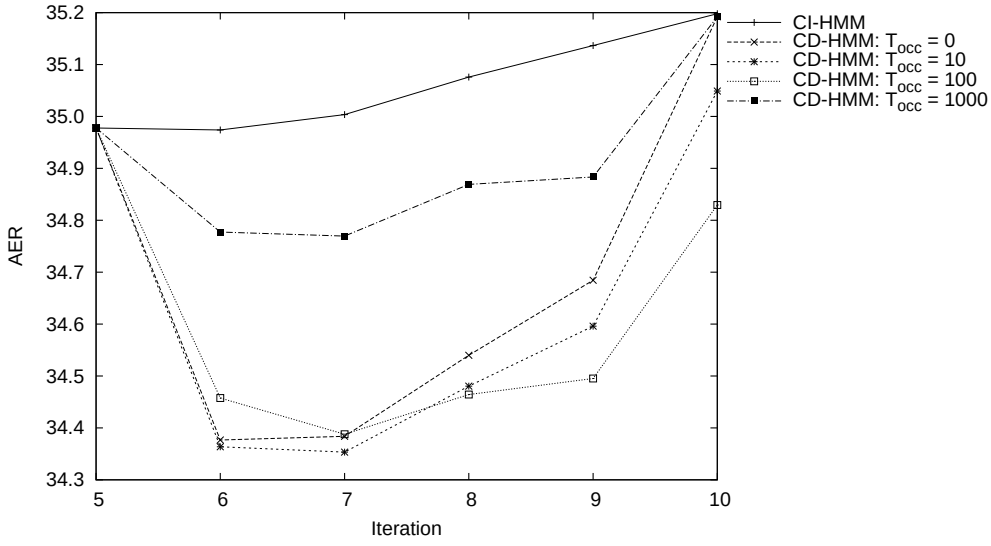
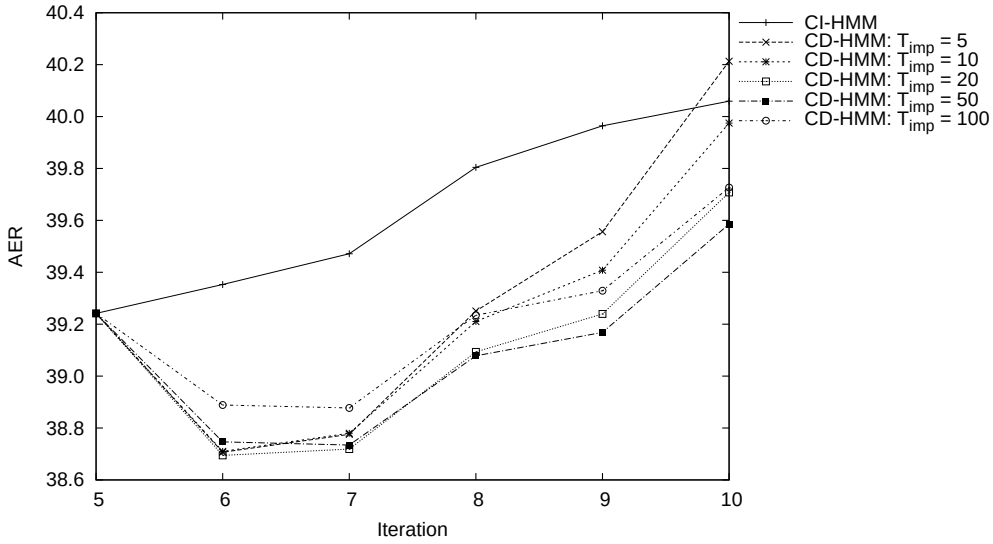
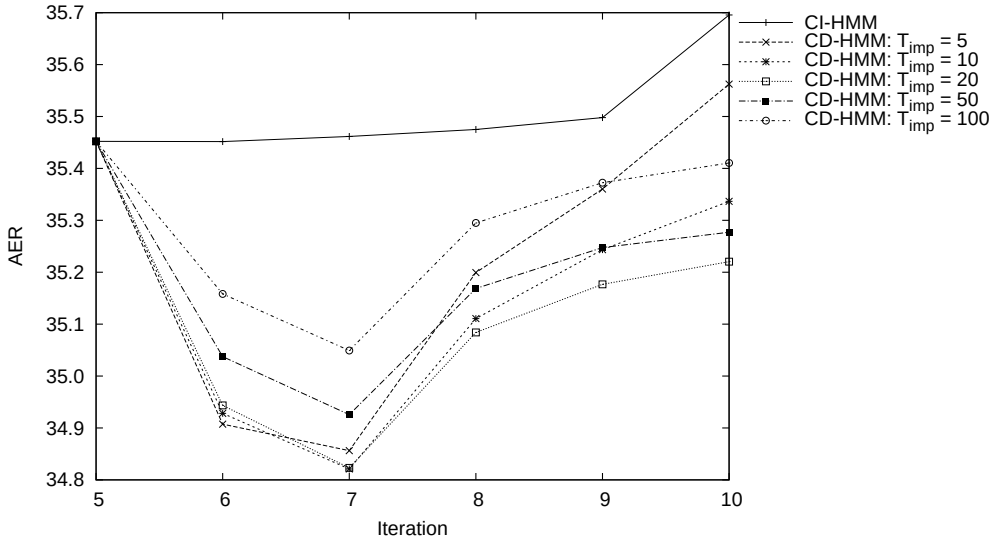


Figure 9.4: AER of CD-HMM alignment for English to Arabic (top), Arabic to English (middle) and their union (bottom) for varying T_{occ}

English to Arabic



Arabic to English



Union

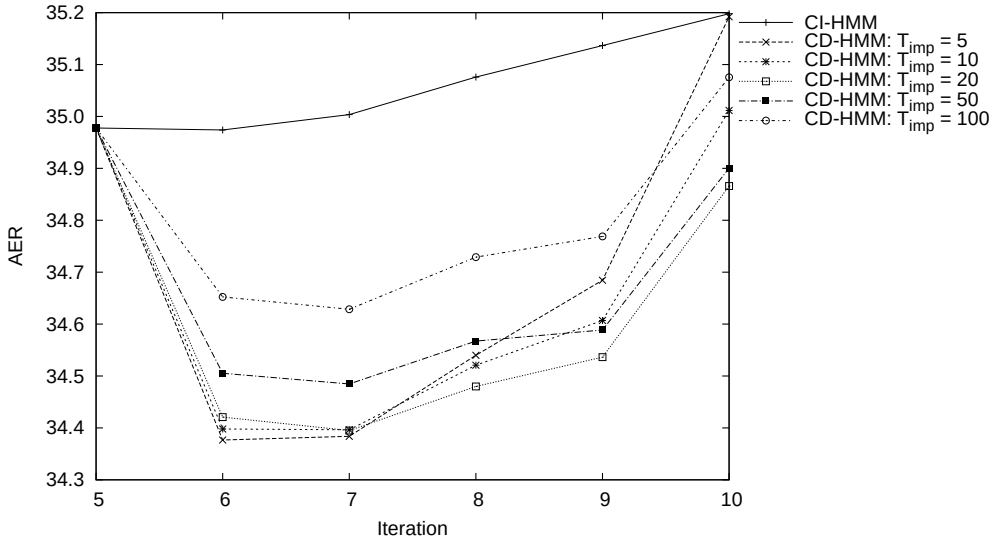


Figure 9.5: AER of CD-HMM alignment for English to Arabic (top), Arabic to English (middle) and their union (bottom) for varying T_{imp}

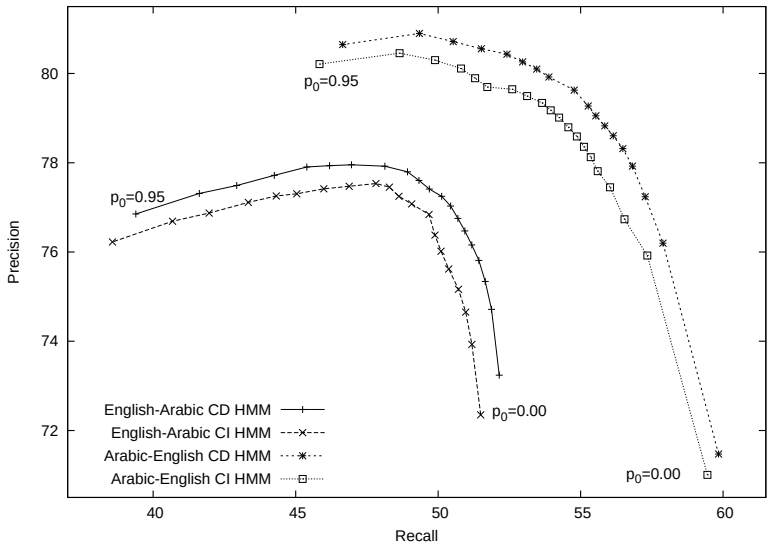


Figure 9.6: Precision/recall curves for the context-dependent HMM and the baseline context-independent HMM, for Arabic to English and English to Arabic. p_0 varies from 0.00 to 0.95 in steps of 0.05.

alignments for every probability threshold. The difference between the models is greatest in the region where precision is approximately the same as recall, i.e. the context-dependent model shows the greatest improvement over the context-independent model in the region in which they are normally used.

9.5 Evaluation of alignment quality for Chinese

Context-dependent HMM models were trained on the $\frac{1}{10}$NIST 08 data set, initialised from CI-HMMs trained on the same data. The variation of alignment quality as training progresses is shown in Figure 9.8. We see that the context-dependent models again produce improved quality of alignment for the first two iterations of training. For Chinese to English translation, the AER of the CI-HMM alignments also decreases as additional iterations are performed, though the AER of the CD-HMM alignments is lower so the context-dependent models again produce alignments of a higher quality in both directions.

Like the source contexts for Arabic-to-English alignment, the parts-of-speech of the next and previous words are more informative in determining the cluster than the part-of-speech of the current word. Of the 50 questions occurring most commonly at the root node of the decision tree for Chinese to English translation, 20 concern the previous word, 21 concern the next word and 9 concern the current word. For English to Chinese translation, only 5 of the top questions depend on the current word, with 24 depending on the previous word and 21 on the next. Table 9.4 shows the most commonly used questions in each translation direction for $T_{\text{imp}} = 10$ and $T_{\text{occ}} = 100$.

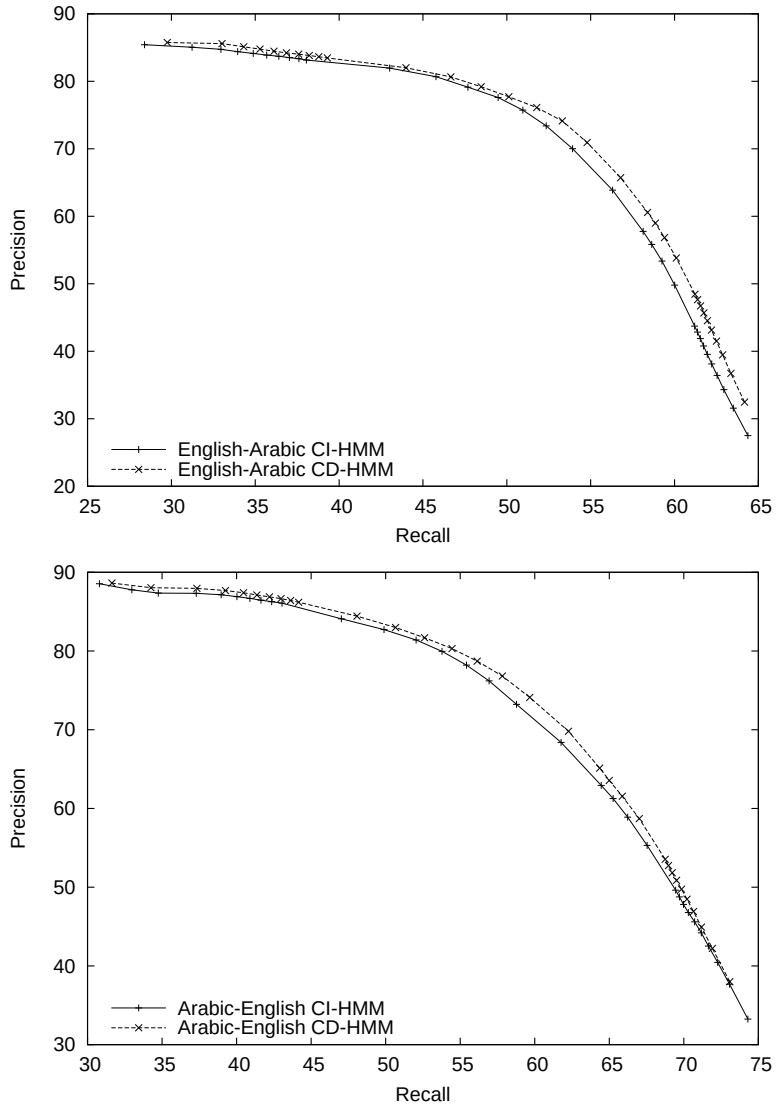


Figure 9.7: Precision/recall curves generated by variation of alignment link posterior probability threshold λ for the context-dependent HMM and the baseline context-independent HMM, for Arabic to English (top) and English to Arabic (bottom). λ varies from 0.001 *to* 0.99*, with more data points at the ends of this range.*

9.6 Evaluation of translation quality for context-dependent HMMs

We have shown that the context-dependent models produce a decrease in AER measured on manually-aligned data; we wish to show this improved model performance leads to an increase in translation quality, measured by BLEU score (Papineni et al., 2002). We report results for Arabic to English and Chinese to English translation using a variety of data sets.

Alignment models were used to align the training data using the models in each translation

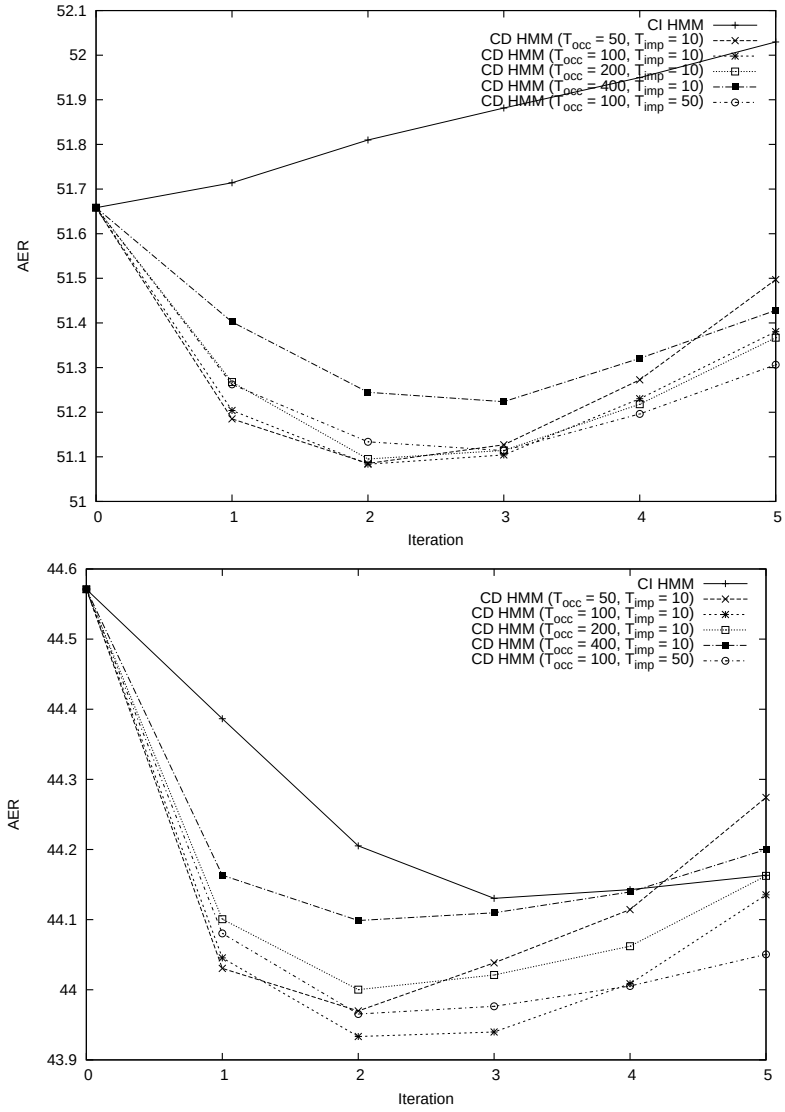


Figure 9.8: Graph showing AER for Chinese-English HMMs on $\frac{1}{10}$NIST 08 data set, for English to Chinese (top) and Chinese to English (bottom)

direction. HiFST, a WFST-based hierarchical translation system described by Iglesias et al. (2009a), was trained on the union of these alignments. Note that the words-with-context are not used during the translation process; they are used just to obtain the alignments, and translation continues with standard words with no additional context information. The features used for HiFST were:

- Output language model (described in Section 2.4),
- Input-to-output and output-to-input phrase translation models (described in Section 3.10),
- Word insertion penalty,
- Rule penalty that assigns a cost to each rule used to favour simpler derivations,

English question	Frequency
Is_Next_Noun	646
Is_Next_Preposition	613
Is_Prev_Adjective	527
Is_Prev_Determiner	500
Is_Next_Noun_Singular_Mass	382
Is_Prev_Preposition	379
Is_Prev_Noun	288
Is_Next_Punctuation	277
Is_Prev_Noun_Singular_Mass	263
Is_Prev_Verb	249

Chinese question	Frequency
Is_Next_NN	1522
Is_Prev_NN	902
Is_Next_PU	481
Is_Prev_VV	365
Is_NN	339
Is_Next_VV	323
Is_Prev_AD	187
Is_Prev_NR	151
Is_Next_CD	107
Is_Prev_P	107

Table 9.4: Most frequent root node context questions for English (top) and Chinese (bottom)

- Number of usages of the glue rule (Chiang, 2005),
- Input-to-output and output-to-input lexical models (Koehn et al., 2003),
- Three rule count features inspired by Bender et al. (2007), which condition on whether the rule occurred once, twice or more times in the training data.

MERT (Och, 2003) was carried out using a development set, and the BLEU score was evaluated on a number of test sets. Decoding used a 4-gram language model, though the full lattice of translation hypotheses was rescored with a 5-gram language model in some cases (this is detailed in the individual experiment descriptions).

For both Arabic and Chinese to English translation, the CD-HMM models were evaluated as follows. Iteration 5 of the CI-HMMs were used to produce alignments for the parallel text training data, whose union was used to train the baseline system. The CI-HMM was then used to initialise a CD-HMM, which was trained on the tagged data until the AER of a development set increased. The same data were aligned using the CD-HMMs and a second WFST-based translation system built from these alignments. The models were evaluated by comparing the BLEU scores obtained by translating the test set with those of the baseline model.

9.6.1 Arabic to English translation

Alignment models were trained on the NIST MT08 Arabic-English parallel text, excluding the UN portion. The HMM null alignment probability p_0 was chosen based on the AER,

Experiment	Model	mt02_05_tune	mt02_05_test	MT08-nw
NIST non-UN	CI-HMM	**50.0**	49.4	46.3
	CD-HMM	**50.0**	**49.7**	**46.9**
NIST	CI-HMM	**52.7**	**52.0**	–
	CD-HMM	52.1	51.8	–

Table 9.5: Comparison of the CD-HMM with the baseline CI-HMM for Arabic, using Bleu *score*

resulting in values of $p_0 = 0.05$ for Arabic to English and $p_0 = 0.10$ for English to Arabic. We performed experiments on the NIST Arabic-English translation task. The *mt02_05_tune* and *mt02_05_test* data sets are formed from the odd and even numbered sentences of the NIST MT02 to MT05 evaluation sets respectively; each contains 2k sentences and 60k words. We use *mt02_05_tune* as a development set and evaluate the system on *mt02_05_test* and the newswire portion of the MT08 set, *MT08-nw*. The English language model was a 4-gram model estimated from the English side of the entire MT08 parallel text, and a 965M word subset of monolingual data from the English Gigaword Third Edition.

CD-HMMs were trained using occupancy threshold 100 and improvement threshold 10. Table 9.5 shows a comparison of the system trained using CD-HMMs with the baseline system, which was trained using CI-HMM models on untagged data. The context-dependent models result in a gain in Bleu score of 0.3 for *mt02_05_test* and 0.6 for *MT08-nw*.

Further testing with a larger training set was carried out on the full NIST Arabic-English parallel text, including an additional 5.6M words of UN data. The same occupancy and improvement thresholds were used. The algorithms scale well for use on large data sets, particularly with caching of answers to questions.

However, the translation system initialised from the CD-HMMs was unable to improve on one initialised from the context-independent HMMs in this case. A possible reason for this is that the occupancy threshold T_{occ} and improvement threshold T_{imp} were not optimally set: with no automatic method of setting these parameters, they are set by trial and error. However, it is time-consuming to determine the effect of many different parameter combinations on translation quality since clustering must be performed to produce an alignment model for each parameter setting, and a separate translation system must be initialised from each alignment model. It is likely that the context-dependent models will perform better if better thresholds can be found.

9.6.2 Chinese to English translation

For experiments in Chinese-to-English translation, we train on the parallel text and MERT is performed to optimise feature weights on a development set *Tune*. Translation quality is then evaluated on a validation set and a test set. Results for these experiments are shown in Table 9.6.

For initial investigation, models were trained on the $\frac{1}{10}$NIST MT08 data set. The development set *tune-nw* and validation set *test-nw* contain a mix of the newswire portions of MT02 through MT05 and additional developments sets created by translation within the GALE program. We also report results on the newswire portion of the MT08 set, *MT08-nw*. Again we see an increase in Bleu score for both test sets: 0.5 for *test-nw* and 0.8 for *MT08-nw*.

Experiment	Model	Development	Validation	Test
$\frac{1}{10}$NIST non-UN	CI-HMM	28.1	28.5	26.9
	CD-HMM	**28.5**	**29.0**	**27.7**
NIST 09	CI-HMM	31.7	32.2	30.3
	CD-HMM	**31.8**	**32.5**	**30.7**
AGILE P4 newswire	CI-HMM	28.4	28.7	21.0
	CD-HMM	28.4	29.0	21.0
	CD-HMM + CI-HMM	**29.3**	**29.6**	**22.0**
AGILE P4 web	CI-HMM	16.1	15.8	14.4
	CD-HMM	16.3	15.9	14.4
	CD-HMM + CI-HMM	**16.8**	**16.5**	**15.1**

Table 9.6: Comparison of the CD-HMM with the baseline CI-HMM for Chinese, using BLEU *score*

We also see improvements for large data sets. For the NIST 09 and AGILE phase 4 data, separate HiFST translation systems were built using the CD-HMM alignments and the CI-HMM alignments. Each system was discriminatively trained using MERT on the *Tune* set for those evaluations, before the system was evaluated on the *SysCombTune* and *Test* sets. During decoding, 4-gram language models were used to produce a lattice. The lattice was then rescored using a stupid backoff 5-gram language model (Brants et al., 2007) trained on over six billion words of English text, including the Gigaword corpus and the English side of the parallel text. We see modest gains from using the CD-HMM alignments compared to the CI-HMM alignments.

The translation quality is further enhanced by the use of system combination. We use the MBR system combination described in Chapter 2 to combine the output of the two systems for the AGILE P4 data. For the newswire data, we achieve gains of 0.6 BLEU points on the validation set and a full BLEU point gain over either system on the test set. We also report good results on the web data, with gains of 0.6 and 0.7 BLEU points on the validation and test set respectively.

9.6.3 French to English translation

Context-independent HMM alignment models were trained[1] using the Europarl, News Commentary and United Nations parallel text corpora allowed for the WMT 2010 evaluation (Pino et al., 2010). Context-dependent HMMs were initialised from these and further iterations of context-dependent training carried out using the same parallel text. A 4-gram language model was used during decoding and lattices were rescored using a 5-gram language model before MBR rescoring was applied to each system individually. Experiments were carried out on the *Tune* and *Test* sets for this evaluation; these contain 2051 and 2525 sentences respectively. Table 9.7 shows the context-dependent models lead to an increase in BLEU score over the context-independent HMMs of 0.4 on both the tuning and test sets.

[1]Thanks to Juan Pino for his help in running these experiments

Experiment	Model	Tune	Test
WMT 10	CI-HMM	24.1	27.9
	CD-HMM	**24.5**	**28.3**

Table 9.7: Comparison of the CD-HMM with the baseline CI-HMM for French, using BLEU *score*

We do not have manually-aligned sentences with which to compare the alignments we produce for French-English alignment, but examination of some of the alignments indicates the context-dependent models generally lead to improved alignment quality.

9.6.4 Translation rules extracted from context-dependent models

The French-English context-dependent alignment models were further analysed by considering the hierarchical translation rules extracted from their alignments. Rules for each sentence pair were extracted as described in Section 3.11 on a sentence-by-sentence basis, using glue rules and rules with one non-terminal of the form $X \rightarrow \langle \alpha_1 X_{\boxed{1}} \alpha_2, \gamma_1 X_{\boxed{1}} \gamma_2 \rangle$, where $\alpha_1, \alpha_2, \gamma_1, \gamma_2$ are allowed to be empty; this family of translation rules was found to be effective for French-English translation (Pino et al., 2010). These rules were then used to attempt to find a synchronous derivation for 1281 sentence pairs. If such a derivation can be found, the system is capable of generating the output sentence as a translation hypothesis; if not, the reference output cannot even be hypothesised by the system (though a very similar one may be). We therefore desire that we can find as many sentence pairs which have derivations as possible. With the context-independent models, derivations could be found for 81.7% of sentences, compared to 84.5% for the context-dependent models, indicating that the context-dependent models lead to rules with a better coverage of the sentences.

The presence of spurious links in a sentence prevents rules being extracted since a phrase pair cannot contain words that are linked to words beyond the phrase. The context-dependent models appear to prevent spurious long-distance alignment links, particularly those involving function words. For example, the context-dependent model can distinguish between *to* as part of a verb infinitive and its use as a preposition.

Figure 9.9 shows the alignments produced for a sentence pair by the CI and CD alignment models. The context-dependent model learns that *of* followed by a plural noun translates to *des* rather than *de* and prevents a spurious long-distance link, which results in rules being extracted that enable synchronous derivations of the sentence pair (the derivation of lowest cost is shown). No derivation was possible using rules extracted from the context-independent alignment. Another sentence for which there was no derivation using the context-independent alignment is shown in Figure 9.9.

9.7 Conclusions

We have shown that the context-dependent alignment models are capable of producing better quality alignments than their context-independent equivalents. We have evaluated the

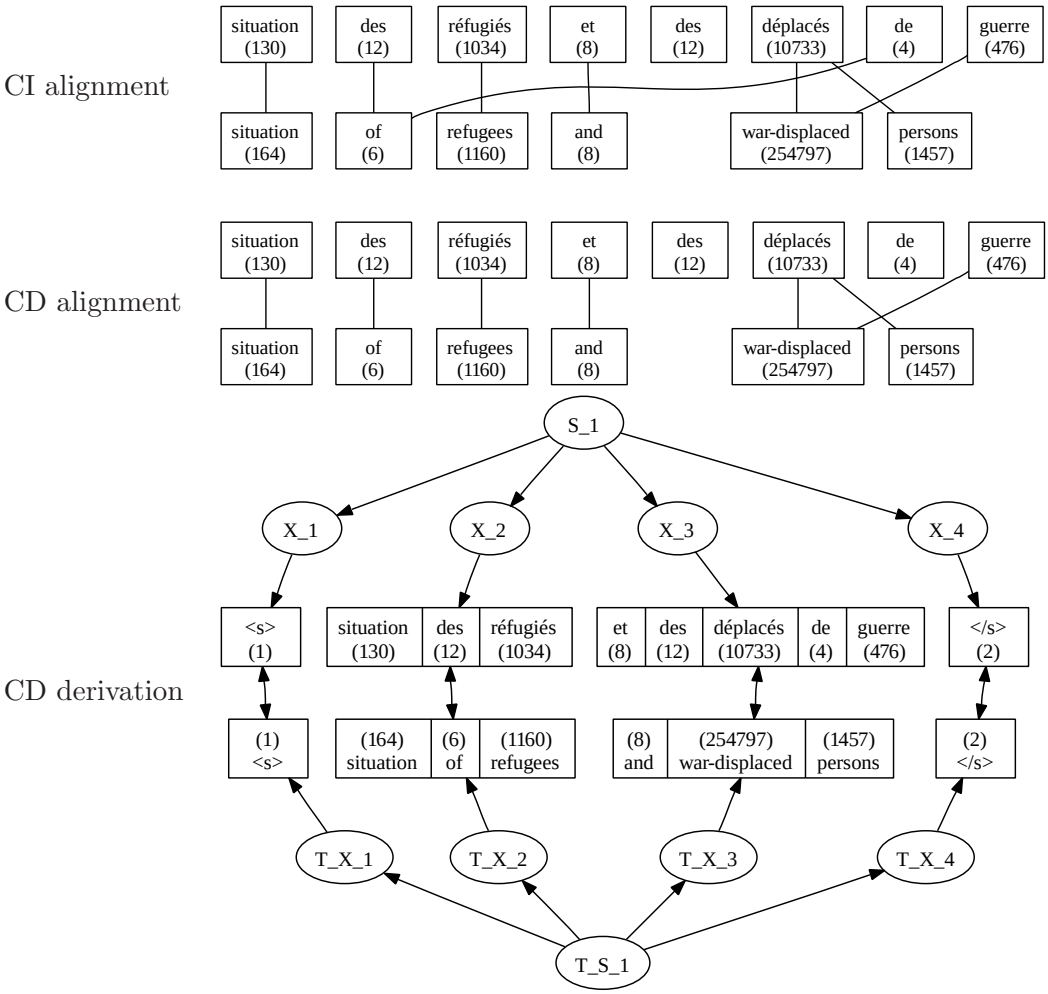


Figure 9.9: Improved alignment quality of English-French alignment with context-dependent models, and the resulting derivation of the sentence pair using rules extracted from the context-dependent alignment.

models by comparing the AER of alignments output by the models against a set of reference alignments.

It is not always the case that improved alignment quality leads to improved translation but we have shown that the context-dependent models introduced here lead to improved quality of translation. Systems initialised using alignments from these models have performed better than those initialised from context-independent alignments. For Arabic to English, Chinese to English and French to English translation, we report an increase in translation quality measured by BLEU score compared to a system built using context-independent alignments. These gains are compatible with MERT, rescoring with phrasal segmentation models and rescoring with 5-gram language models.

The decision tree clustering algorithm is efficient, especially with optimisation such as storing the answers to questions and careful selection of questions during clustering. Once

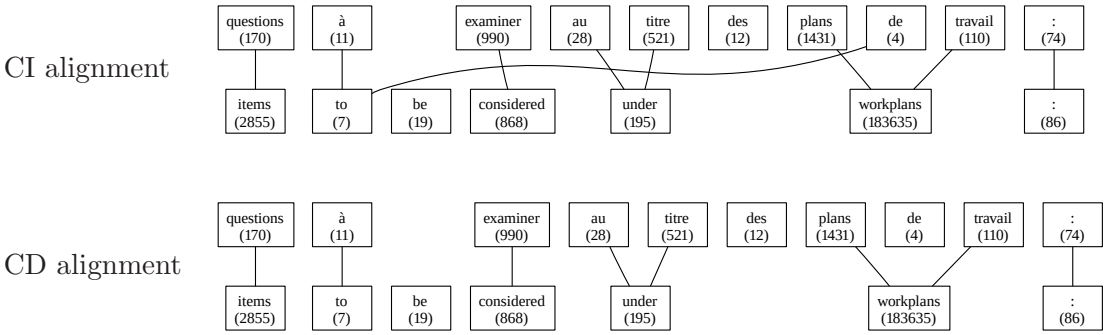


Figure 9.10: A further example of improved alignment quality of English-French alignment with context-dependent models

clustering has been performed, we use the EM algorithm for training, which is also runs efficiently. We have shown that clustering and model training can be used on large amounts of parallel text, up to 11 million sentences in length, the largest parallel text we use for any experiments.

Chapter 10
Conclusions and Future Work

This dissertation has addressed the use of alignment models in statistical machine translation. It has investigated algorithms for the training of alignment models and for the use of those alignment models for improving the quality of machine translation systems.

10.1 Review of the work

The first part of this work introduces a method of representing every possible alignment to a source sentence of each sentence in a lattice of target language translations, for particular generative models of alignment. A lattice encodes the joint probability of each target sentence and alignment given the source sentence, for Model 1 and the word-to-word HMM alignment model; each path through the lattice corresponds to a particular target sentence and alignment.

These lattices form the basis of a method for rescoring lattices using the alignment models, where a lattice of output language translation hypotheses is aligned to a sentence in the input language. This is much more efficient than rescoring N-best lists and results in no search errors. Experiments using the TTM decoder showed that rescoring using alignment model scores in addition to the score assigned by the translation system increased Bleu score of the resulting translations by up to 3.0 points. Rescoring using Model 1 was found to perform better than rescoring using the HMM and the algorithm is more efficient due to reduced alignment dependencies.

Chapter 6 introduces discriminative training for the IBM Model 1 and word-to-word HMM alignment models. Previous uses of discriminative training in machine translation have been restricted to estimating a small number of feature weights for log-linear models of translation and alignments; we adopt an approach that has more in common with discriminative training techniques for speech recognition and re-estimate many parameters within the model. We derive a method for using manually-aligned reference alignments to estimate parameters to increase the number of correct alignment links and reduce the number of incorrect alignment links produced by the model. We update word-to-word translation probabilities for all words that appear in the manually-aligned training data, as well as re-estimating parameters for the alignment component of the word-to-word HMM model.

Using discriminatively trained alignment models, we achieve improved quality of alignment as measured by AER. However, it is not always the case that improved quality of alignment leads to improved translation quality. Despite the increase in alignment quality, translation systems initialised from discriminatively trained HMMs did not yield a significant increase in translation quality compared to systems initialised from the standard HMMs.

In Chapter 8, we motivate the addition of source context information into interlingual word-to-word translation probabilities and extend IBM Model 1 and word-to-word HMM alignment models to include context information. We develop decision tree clustering algorithms, based on direct optimisation of the EM auxiliary function, to form classes of contexts with common probability distributions. The use of clustering significantly reduces the size of the model and prevents over-fitting to the training data. We describe a number of methods of introducing context information, including use of the part-of-speech of the source word and the previous and next source words. We also discuss lexical context, i.e. the identities of the previous and next words, stemming and dependency parsing, concluding with a discussion of methods for incremental inclusion of context from multiple sources.

We evaluate context-dependent alignment models in Chapter 9, testing their performance for Chinese to English, Arabic to English and French to English translation. The context-dependent models are efficient enough to be trained on the largest parallel text corpora available to us, and have been included in the Cambridge University AGILE system. We find that context-dependent Model 1 and HMM alignment models consistently provide higher quality alignments than their context-independent equivalents, with lower AER relative to reference alignments. Evaluation by comparison of machine translation systems constructed using the models is similarly successful. In all languages, we obtain a higher quality of translation measured by Bleu score over a variety of data sets when the translation system is initialised from alignments produced by context-dependent models. Chinese to English translation systems trained from context-dependent alignment models and context-independent models can be combined using MBR system combination to give improved translation quality over either system individually, and we believe system combination will also work for other language pairs.

10.2 Suggestions for future work

The use of alignment models to rescore lattices of translation hypotheses proved initially effective, though its effect was reduced when the baseline was improved by the use of 5-gram language models and phrasal segmentation models for lattice rescoring. Advances in error rate training for optimisation of feature weights in a translation system have led to the ability

to include a large number of features in the system. With these advances, it may be possible to incorporate the alignment model scores as a feature and estimate the feature weight during discriminative training; this may lead to larger gains in combination with the other rescoring techniques than we have achieved here. The algorithms presented allow a lattice of translation hypotheses to be rescored, allowing relatively simple integration into the overall MT system.

The thread of this research that shows most promise is the addition of context-dependence to alignment models. With increased amounts of data available for training models, we will in future have more data available to estimate their parameters accurately. Directions for future research include:

- **Extension to other models:** We have concentrated on applying context-dependence to Model 1 and the word-to-word HMM model only. While the word-to-phrase HMM models and the word-to-phrase model with bigram translation table have not shown consistent improvement over the word-to-word model for large data sets, extension of these more sophisticated models to include dependence on source word context may yield good results.

- **Automatic determination of thresholds in parameter estimation:** The parameters for decision tree growth have been manually tuned for all experiments, with parameters that provide the largest increase in alignment quality measured by AER. Some manual judgement has also been used to choose between systems with very similar alignment quality: for Chinese to English translation, the CD-HMM with the larger occupancy threshold was chosen over a similar model with a smaller occupancy threshold as its should be more accurately estimated. This is time-consuming since clustering has to be performed for every model. Automatic methods for determining the occupancy and improvement thresholds should be investigated. Techniques such as cross validation and maximisation of the Bayesian Information Criterion have been used for growing decision trees for clustering in ASR (Chen and Gopalakrishnan, 1998).

- **Further translation experiments:** As discussed in Chapter 3, it is difficult to find a metric for alignment quality that correlates well with translation quality. The choice of metric is dependent on the type of machine translation system used and the data used in training and evaluation. It is therefore still not clear how best to produce alignments that will enable HiFST to learn useful rules. We symmetrise the alignments by taking their union. It is possible that other methods of symmetrisation and alignment link selection may perform better with the models we have developed. Further translation experiments should be carried out to determine properties of alignment models that lead to improved translation quality.

- **Clustering of similar words:** In this work, we only split the contexts of words that occur in the training data. For words that occur infrequently, there may still be data sparsity issues, even without clustering, as a source word may occur only a few times in the training data. Clustering the probability distributions of these infrequently occurring words together may produce improvements in alignment quality; for example, words derived from the same root may translate in a similar way.

  This approach could even be used for spelling mistakes or typographical errors that occur in the source language: incorrect words are likely to be similar to the intended words and translate in the same way, however they are currently treated as a separate

word in their own right. This should improve the system's robustness to errors in the text.

With some method of determining the words to be considered for clustering, a slight modification to the clustering algorithm and the use of appropriate questions, it would be possible to form clusters of words.

- **Merging of similar decision tree nodes:** The algorithm we use for growing decision trees is greedy and is not guaranteed to find the optimal tree; it simply makes decisions locally as the tree is built. Some of the leaf nodes of the tree may have similar probability distributions. A common practice in decision tree clustering is to merge similar leaf nodes once the tree has been built (Breiman et al., 1984), which can decrease the model size and reduce data sparsity problems. The leaf nodes whose merging causes the minimal reduction in objective function can be merged.

  With the exception of the more general questions described in Section 9.2.2, the questions about a particular position in the context are mutually exclusive, i.e. if a context answers *yes* to one question about (say) the previous word, it will not satisfy any other question about the previous word. This means we cannot ask questions of the form *Is the previous word x or y?*, which may well be desirable, without explicitly specifying the question in advance. Merging leaf nodes of the tree can alleviate this situation, by combining those contexts that satisfy *Is the previous word x?* and *Is the previous word y?*.

10.3 Conclusion

This thesis covers three aspects of alignment modelling for machine translation. The first is the use of the alignment models during translation to rescore lattices of translation hypotheses. The second is a novel use of discriminative training to estimate word-to-word translation probabilities and alignment probabilities within the alignment model. Finally, we present context-dependent alignment models and algorithms for their training, showing that they lead to improved quality of alignment in multiple languages. We report improved translation quality over a good baseline system when the models are used to produce the word-aligned parallel text from which the machine translation system is built.

Appendix A
Data sets

Language	Parallel text	Sentences	English words	Foreign words
Arabic-English	NIST 05	3.8M	104M	100M
	NIST 05 news subset	137k	3.8M	3.6M
Chinese-English	NIST 05	9.8M	226M	212M
French-English	WMT 06	688k	15.2M	16.7M
	WMT 10	8.8M	278M	241M
Spanish-English	TC-STAR 06	1.5M	42.3M	43.8M

Table A.1: Summary of the data sets used for training alignment models for rescoring experiments

Language	Parallel text	Sentences	English words	Foreign words
Arabic-English	AGILE v4.1	8.7M	232M	232M
	NIST v1.1	5.8M	155M	156M
	Manually aligned	27.9k	784k	610k
Chinese-English	AGILE v3.2	10.2M	241M	225M
	Manually aligned	11.6k	336k	277k

Table A.2: Summary of the data sets used for discriminative training of alignment models

Language	Parallel text	Sentences	English words	Foreign words
Arabic-English	NIST 08 non-UN	300k	9.5M	8.4M
	NIST 08	5.9M	154M	153M
	Manually aligned	28k	790k	667k
Chinese-English	$\frac{1}{10}$NIST 08	600k	16.6M	15.2M
	NIST 09	8.3M	203M	190M
	AGILE P4	11.0M	264M	242M
	Manually aligned	11.6k	336k	277k
French-English	WMT 10	8.8M	278M	241M

Table A.3: Summary of the data sets used for context-dependent model training and evaluation

APPENDIX B

Alternative optimisation criteria for MMI

For sentences e_1^I and f_1^J and alignment links A between the sentences, define

$$A = \{(i,j) : 0 \leq i \leq I, 0 \leq j \leq J, e_i \text{ is linked to } f_j\}, \tag{B.1}$$

and define the likelihood

$$l(A) = \prod_{(i,j)\in A} \mathrm{P}((i,j)|f_1^J, e_1^I), \tag{B.2}$$

where $\mathrm{P}((i,j)|f_1^J, e_1^I)$ is the posterior probability of e_i being aligned to f_j under an alignment model. This is defined for

We now show that for Models 1 and 2 and an alignment A valid under the model, the likelihood $l(A)$ is equivalent to the posterior probability of A under the model, i.e. the posterior probability of the alignment is the product of the posterior probabilities of the individual alignment links. The probability of f_1^J given e_1^I is

$$\begin{aligned}
\mathrm{P}(f_1^J|e_1^I) &= \sum_{a_1^J} \mathrm{P}(f_1^J, a_1^J|e_1^I) \\
&= \sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \mathrm{P}(f_1^J, a_1^J|e_1^I) \\
&= \sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} a(a_j|j,I,J)t(f_j|e_{a_j}).
\end{aligned} \tag{B.3}$$

Following Deng (2005b), we can simplify this as follows:

$$\begin{aligned}
&\sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \prod_{j=1}^{J} a(a_j|j,I,J)t(f_j|e_{a_j}) \\
= &\sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \left[\prod_{j=1}^{J-1} a(a_j|j,I,J)t(f_j|e_{a_j})\right] a(a_J|J,I,J)t(f_J|e_{a_J}) \\
= &\sum_{a_1=0}^{I} \cdots \sum_{a_{J-1}=0}^{I} \left[\prod_{j=1}^{J-1} a(a_j|j,I,J)t(f_j|e_{a_j})\right] \sum_{a_J=0}^{I} a(a_J|J,I,J)t(f_J|e_{a_J}) \qquad \text{(B.4)}
\end{aligned}$$

Inductively

$$\begin{aligned}
\mathrm{P}(f_1^J|e_1^I) &= \sum_{a_1=0}^{I} a(a_1|1,I,J)t(f_1|e_{a_1}) \times \cdots \\
&\quad \times \sum_{a_j=0}^{I} a(a_j|j,I,J)t(f_j|e_{a_j}) \times \cdots \\
&\quad \times \sum_{a_J=0}^{I} a(a_J|J,I,J)t(f_J|e_{a_J}) \\
&= \prod_{j=1}^{J} \sum_{i=0}^{I} a(i|j,I,J)t(f_j|e_i). \qquad \text{(B.5)}
\end{aligned}$$

Therefore the dependencies of the model are such that we can swap the sum over all alignments of a product over target positions, to be the product over target positions of a sum over alignment positions. We now find the posterior probability of an alignment given the source and target sentences.

$$\begin{aligned}
\mathrm{P}(a_1^J|f_1^J,e_1^I) &= \frac{\mathrm{P}(f_1^J,a_1^J|e_1^I)}{\mathrm{P}(f_1^J|e_1^I)} \\
&= \frac{\prod_{j=1}^{J} a(a_j|j,I,J)t(f_j|e_{a_j})}{\prod_{j=1}^{J}\sum_{i=0}^{I} a(i|j,I,J)t(f_j|e_i)} \\
&\quad \text{(substituting from Equation B.5)} \\
&= \prod_{j=1}^{J} \frac{a(a_j|j,I,J)t(f_j|e_{a_j})}{\sum_{i=0}^{I} a(i|j,I,J)t(f_j|e_i)} \\
&= \prod_{j=1}^{J} f(a_j|j,f_1^J,e_1^I) \qquad \text{(B.6)}
\end{aligned}$$

where we define

$$f(a_j|j,f_1^J,e_1^I) = \frac{a(a_j|j,I,J)t(f_j|e_{a_j})}{\sum_{i=0}^{I} a(i|j,I,J)t(f_j|e_i)}. \qquad \text{(B.7)}$$

We now show that $f(a_j|j, f_1^J, e_1^I)$ is in fact the posterior probability of the target word at position j linking to the source word at position a_j. The posterior probability of an alignment link between the source word in position i and the target word in position j is

$$\begin{aligned}
\mathrm{P}(a_j = i|f_1^J, e_1^I) &= \sum_{a_1^J} \delta_i(a_j)\mathrm{P}(a_1^J|f_1^J, e_1^I) \\
&= \sum_{a_1=0}^{I} \cdots \sum_{a_J=0}^{I} \delta_i(a_j)p(a_1|1, f_1^J, e_1^I) \cdots f(a_J|J, f_1^J, e_1^I) \\
&\quad \text{(substituting from Equation B.6)} \\
&= \sum_{a_1=0}^{I} f(a_1|1, f_1^J, e_1^I) \times \cdots \\
&\quad \times \sum_{a_j=0}^{I} \delta_i(a_j) f(a_j|j, f_1^J, e_1^I) \times \cdots \\
&\quad \times \sum_{a_J=0}^{I} f(a_J|J, f_1^J, e_1^I). \qquad \text{(B.8)}
\end{aligned}$$

Now for $1 \leq j \leq J$,

$$\sum_{a_j=0}^{I} f(a_j|j, f_1^J, e_1^I) = \frac{\sum_{a_j=0}^{I} a(a_j|j, I, J)t(f_j|e_{a_j})}{\sum_{i=0}^{I} a(i|j, I, J)t(f_j|e_i)} = 1. \qquad \text{(B.9)}$$

Therefore

$$\mathrm{P}(a_j = i|f_1^J, e_1^I) = f(i|j, f_1^J, e_1^I) \qquad \text{(B.10)}$$

We now return to the quantity we hope to maximise. For a valid Model 1 or Model 2 alignment sequence a_1^J, the set of alignment links is given by

$$A = \{(a_j, j) : 1 \leq j \leq J\} \qquad \text{(B.11)}$$

and its likelihood is given by

$$\begin{aligned}
l(A) &= \prod_{j=1}^{J} \mathrm{P}(a_j|j, f_1^J, e_1^I) \\
&= \prod_{j=1}^{J} p(a_j|j, f_1^J, e_1^I) \\
&= \mathrm{P}(a_1^J|f_1^J, e_1^I), \qquad \text{(B.12)}
\end{aligned}$$

i.e. the likelihood we maximise is exactly the same as the posterior probability of the sentence-level alignment A. Therefore, the alternative criterion is consistent with the criteria used during MMI training for valid Model 1 and Model 2 alignments. Due to the dependencies of the model, there is no such convenient formula for the HMM model; however we can still seek to maximise the likelihood as defined in Equation (6.64).

APPENDIX C

Part-of-speech tags for context-dependent alignment models

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
FW	Foreign word
IN	Preposition/subordinating conjunction
JJ	Adjective
NN	Noun, singular or mass
NNP	Proper noun, singular
NNPS	Proper noun, plural
NNS	Noun, plural
PRP	Personal pronoun
PUNC	Punctuation
RB	Adverb
RP	Particle
UH	Interjection
VBD	Verb, past tense
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
WP	Wh-pronoun

Table C.1: Part-of-speech tags output by the MADA Arabic part-of-speech tagger (Habash and Sadat, 2006)

AD	adverb
AS	aspect marker
BA	把 in ba-construction
CC	coordinating conjunction
CD	cardinal number
CS	subordinating conjunction
DEC	的 in a relative-clause
DEG	associative 的
DER	得 in V-de const. and V-de-R
DEV	地 before VP
DT	determiner
ETC	for words 等, 等等
FW	foreign words
IJ	interjection
JJ	other noun-modifier
LB	被 in long bei-const
LC	localizer
M	measure word
MSP	other particle
NN	common noun
NR	proper noun
NT	temporal noun
OD	ordinal number
ON	onomatopoeia
P	preposition excl. 被 and 把
PN	pronoun
PU	punctuation
SB	被 in short bei-const
SP	sentence-final particle
VA	predicative adjective
VC	是
VE	有 as the main verb
VV	other verb

Table C.2: Part-of-speech tags output by MXPOST Chinese part-of-speech tagger (from Xia (2000))

Tag	Description	Tag	Description
$	Dollar	NNS	Noun, plural
‘‘	Opening quotation mark	PDT	Pre-determiner
’’	Closing quotation mark	POS	Genitive marker
(	Opening parenthesis	PRP	Personal pronoun
)	Closing parenthesis	PRP$	Possessive pronoun
,	Comma	RB	Adverb
--	Dash	RBR	Comparative adverb
.	Sentence terminator	RBS	Superlative adverb
:	Colon or ellipsis	RP	Particle
CC	Coordinating conjunction	SYM	Symbol
CD	Cardinal number	TO	“To” as preposition or infinitive marker
EX	Existential there	UH	Interjection
DT	Determiner	VB	Verb, base form
FW	Foreign word	VBD	Verb, past tense
IN	Preposition or subordinating conjunction	VBG	Verb, present participle or gerund
JJ	Adjective	VBN	Verb, past participle
JJR	Comparative adjective	VBP	Verb, non-3rd person singular present
JJS	Superlative adjective	VBZ	Verb, 3rd person singular present
LS	List item marker	WDT	Wh-determiner
MD	Modal auxiliary	WP	Wh-pronoun
NN	Noun, singular or mass	WP$	Wh-pronoun, possessive
NNP	Proper noun, singular	WRB	Wh-adverb
NNPS	Proper noun, plural		

Table C.3: Part-of-speech tags output by the TnT English part-of-speech tagger (Brants, 2000)

References

Allauzen, C., Mohri, M., and Roark, B. (2003). Generalized algorithms for constructing statistical language models. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 40–47, Morristown, NJ, USA. Association for Computational Linguistics.

Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., and Mohri, M. (2007). OpenFst: A general and efficient weighted finite-state transducer library. In *Proceedings of the Ninth International Conference on Implementation and Application of Automata, (CIAA 2007)*, volume 4783 of *Lecture Notes in Computer Science*, pages 11–23. Springer. `http://www.openfst.org`.

Alshawi, H., Douglas, S., and Bangalore, S. (2000). Learning dependency translation models as collections of finite-state head transducers. *Comput. Linguist.*, 26(1):45–60.

Arun, A. and Koehn, P. (2007). Online learning methods for discriminative training of phrase based statistical machine translation. In *MT Summit XI*.

Aubert, X. L. (2002). An overview of decoding techniques for large vocabulary continuous speech recognition. *Computer Speech and Language*, 16(1):89 – 114.

Ayan, N. F. and Dorr, B. J. (2006). A maximum entropy approach to combining word alignments. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 96–103, Morristown, NJ, USA. Association for Computational Linguistics.

Bahl, L. R., Brown, P. F., DeSouza, P. V., and Mercer, R. L. (1989). A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(7):1001–1008.

Bahl, L. R., de Soutza, P. V., Gopalakrishnan, P. S., Nahamoo, D., and Picheny, M. A. (1991). Context dependent modeling of phones in continuous speech using decision trees. In *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 264–269, Morristown, NJ, USA. Association for Computational Linguistics.

Banchs, R. E., Crego, J. M., de Gispert, A., Lambert, P., and Mari no, J. B. (2005). Statistical machine translation of euparl data by using bilingual n-grams. In *ParaText '05: Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 133–136, Morristown, NJ, USA. Association for Computational Linguistics.

Bangalore, S., Bordel, G., and Riccardi, G. (2001). Computing consensus translation from multiple machine translation systems. In *Proceedings of ASRU*, pages 351–354.

Baum, L. E. and Eagon, J. A. (1967). An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73(3):360–363.

Bender, O., Matusov, E., Hahn, S., Hasan, S., Khadivi, S., and Ney, H. (2007). The RWTH Arabic-to-English spoken language translation system. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU) 2007*, page in press, Kyoto, Japan.

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

Bertoldi, N., Zens, R., and Federico, M. (2007). Speech translation by confusion network decoding. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1297–1300, Honululu, HI, USA.

Bishop, C. M. (1996). *Neural networks for pattern recognition*. Oxford University Press, Oxford, UK.

Blackwood, G., de Gispert, A., Brunning, J., and Byrne, W. (2008a). European language translation with weighted finite state transducers: The CUED MT system for the 2008 ACL workshop on SMT. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 131–134, Columbus, Ohio. Association for Computational Linguistics.

Blackwood, G., de Gispert, A., Brunning, J., and Byrne, W. (2009). Large-scale statistical machine translation with weighted finite state transducers. In *Post Proceedings of the 7th International Workshop on Finite-State Methods and Natural Language Processing, FSMNLP 2008*, pages 39–49, Amsterdam, The Netherlands. IOS Press.

Blackwood, G., de Gispert, A., and Byrne, W. (2008b). Phrasal segmentation models for statistical machine translation. In *Coling 2008: Companion volume: Posters and Demonstrations*, pages 17–20, Manchester, UK. Coling 2008 Organizing Committee.

Blunsom, P. and Cohn, T. (2006). Discriminative word alignment with conditional random fields. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 65–72, Morristown, NJ, USA. Association for Computational Linguistics.

Blunsom, P., Cohn, T., and Osborne, M. (2008). A discriminative latent variable model for statistical machine translation. In *Proceedings of ACL*.

Brants, T. (2000). TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference: ANLP-2000*, Seattle, USA.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA.

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

Brunning, J., de Gispert, A., and Byrne, W. (2009). Context-dependent alignment models for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 110–118, Boulder, Colorado. Association for Computational Linguistics.

Buckwalter, T. (2002). Buckwalter Arabic morphological analyzer.

Carpuat, M. and Wu, D. (2005). Word sense disambiguation vs. statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 387–394, Morristown, NJ, USA. Association for Computational Linguistics.

Cer, D., Jurafsky, D., and Manning, C. D. (2008). Regularization and search for minimum error rate training. In *StatMT '08: Proceedings of the Third Workshop on Statistical Machine Translation*, pages 26–34, Morristown, NJ, USA. Association for Computational Linguistics.

Chan, Y. S. and Ng, H. T. (2008). MAXSIM: A maximum similarity metric for machine translation evaluation. In *Proceedings of ACL-08: HLT*, pages 55–62, Columbus, Ohio. Association for Computational Linguistics.

Chan, Y. S., Ng, H. T., and Chiang, D. (2007). Word sense disambiguation improves statistical machine translation. In *ACL*. The Association for Computer Linguistics.

Chen, B. and Federico, M. (2006). Improving phrase-based statistical translation through combination of word alignment. In *5th International Conference. FinTAL 2006*, pages 356–367. Springer Verlag.

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

Chen, S. S. and Gopalakrishnan, P. S. (1998). Clustering via the Bayesian information criterion with applications in speech recognition. *Acoustics, Speech, and Signal Processing, 1998. ICASSP '98. Proceedings of the 1998 IEEE International Conference on*, 2:645–648 vol.2.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, Morristown, NJ, USA. Association for Computational Linguistics.

Chiang, D. (2007). Hierarchical phrase-based translation. *Comput. Linguist.*, 33(2):201–228.

Chiang, D., DeNeefe, S., Chan, Y. S., and Ng, H. T. (2008a). Decomposability of translation metrics for improved evaluation and efficient algorithms. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 610–619, Morristown, NJ, USA. Association for Computational Linguistics.

Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics on ZZZ*, pages 218–226, Morristown, NJ, USA. Association for Computational Linguistics.

Chiang, D., Marton, Y., and Resnik, P. (2008b). Online large-margin training of syntactic and structural translation features. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Morristown, NJ, USA. Association for Computational Linguistics.

Costa-jussà, M. R. and Fonollosa, J. A. R. (2005). Improving phrase-based statistical translation by modifying phrase extraction and including several features. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 149–154.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585.

Daelemans, W., Bosch, A. V. D., and Weijters, T. (1997). IGTree: Using trees for compression and classification in lazy learning algorithms. *Artificial Intelligence Review*, 11(1-5):407–423.

Darroch, J. N. and Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480.

de Gispert, A., Virpioja, S., Kurimo, M., and Byrne, W. (2009). Minimum Bayes risk combination of translation hypotheses from alternative morphological decompositions. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 73–76, Morristown, NJ, USA. Association for Computational Linguistics.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Deng, Y. (2005a). *Bitext Alignment for Statistical Machine Translation*. PhD thesis, Johns Hopkins University.

Deng, Y. (2005b). MTTK technical report.

Deng, Y. and Byrne, W. (2005a). HMM word and phrase alignment for statistical machine translation. In *Proc. of HLT-EMNLP*.

Deng, Y. and Byrne, W. (2005b). JHU-Cambridge statistical machine translation toolkit (MTTK) user manual.

Deng, Y. and Byrne, W. (2006). HMM word and phrase alignment for statistical machine translation. *IEEE*.

Deng, Y., Kumar, S., and Byrne, W. (2007). Segmentation and alignment of parallel text for statistical machine translation. *Journal of Natural Language Engineering*, 13:3:235–260.

Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

Duh, K. and Kirchhoff, K. (2008). Beyond log-linear models: Boosted minimum error rate training for n-best re-ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 37–40, Columbus, Ohio. Association for Computational Linguistics.

Ehling, N., Zens, R., and Ney, H. (2007). Minimum bayes risk decoding for bleu. In *ACL '07: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 101–104, Morristown, NJ, USA. Association for Computational Linguistics.

Feng, Y., Liu, Y., Mi, H., Liu, Q., and Lü, Y. (2009). Lattice-based system combination for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1105–1113, Singapore. Association for Computational Linguistics.

Fiscus, J. (1997). A post-processing system to yield reduced word error rates: Recogniser Output Voting Error Reduction (ROVER). In *Proceedings 1997 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 347–352, Santa Barbara, CA.

Fossum, V., Knight, K., and Abney, S. (2008). Using syntax to improve word alignment precision for syntax-based machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 44–52, Columbus, Ohio. Association for Computational Linguistics.

Fraser, A. and Marcu, D. (2006a). Measuring word alignment quality for statistical machine translation. Technical Report ISI-TR-616, ISI/University of Southern California.

Fraser, A. and Marcu, D. (2006b). Semi-supervised training for statistical word alignment. In *Proceedings of ACL-2006*, pages 769–776.

Fraser, A. and Marcu, D. (2007). Getting the structure right for word alignment: LEAF. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 51–60.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 961–968, Morristown, NJ, USA. Association for Computational Linguistics.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule? In Susan Dumais, D. M. and Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA. Association for Computational Linguistics.

Ganchev, K., Graça, J. a. V., and Taskar, B. (2008). Better alignments = better translations? In *Proceedings of ACL-08: HLT*, pages 986–993, Columbus, Ohio. Association for Computational Linguistics.

Goel, V. and Byrne, W. J. (2000). Minimum Bayes-risk automatic speech recognition. *Computer Speech and Language*.

Gopalakrishnan, P. S., Kanevsky, D., Nadas, A., and Nahamoo, D. (1989). A generalization of the Baum algorithm to rational objective functions. In *ICASSP-89*.

Gopalakrishnan, P. S., Kanevsky, D., Nádas, A., and Nahamoo, D. (1991). An inequality for rational functions with applications to some statistical estimation problems. *IEEE Transactions on Information Theory*, 37(1):107–113.

Graca, J., Ganchev, K., and Taskar, B. (2008). Expectation maximization and posterior constraints. In Platt, J., Koller, D., Singer, Y., and Roweis, S., editors, *Advances in Neural Information Processing Systems 20*, pages 569–576. MIT Press, Cambridge, MA.

Graff, D., Kong, J., e Chen, and Maeda, K. (2005). English Gigaword second edition. Technical report, Linguistic Data Consortium, Philadelphia.

Habash, N. and Rambow, O. (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 573–580, Morristown, NJ, USA. Association for Computational Linguistics.

Habash, N. and Sadat, F. (2006). Arabic preprocessing schemes for statistical machine translation. In *HLT-NAACL*.

Huang, F. (2009). Confidence measure for word alignment. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 932–940, Suntec, Singapore. Association for Computational Linguistics.

Iglesias, G., de Gispert, A., Banga, E. R., and Byrne, W. (2009a). Hierarchical phrase-based translation with weighted finite state transducers. In *Procedings of NAACL-HLT, 2009*, Boulder, Colorado.

Iglesias, G., de Gispert, A., Banga, E. R., and Byrne, W. (2009b). Rule filtering by pattern for efficient hierarchical translation. In *EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 380–388, Morristown, NJ, USA. Association for Computational Linguistics.

Ittycheriah, A., Al-Onaizan, Y., and Roukos, S. (2006). The IBM Arabic-English word alignment corpus.

Ittycheriah, A. and Roukos, S. (2005). A maximum entropy word aligner for Arabic-English machine translation. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 89–96, Morristown, NJ, USA. Association for Computational Linguistics.

Jurafsky, D. and Martin, J. H. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Kannan, A., Ostendorf, M., and Rohlicek, J. R. (1994). Maximum likelihood clustering of Gaussians for speech recognition. *Speech and Audio Processing, IEEE Transactions on*, 2(3):453–455.

Kneser, R. and Ney, H. (1991). Forming word classes by statistical clustering for statistical language modelling. In *1. Quantitative Linguistics Conference*.

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *International Conference on Acoustic, Speech and Signal Processing*, pages 181–184.

Knight, K. (1999). Decoding complexity in word-replacement translation models. *Comput. Linguist.*, 25(4):607–615.

Knight, K. and Graehl, J. (2005). An overview of probabilistic tree transducers for natural language processing. In *Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics (CICLing), Lecture Notes in Computer Science*.

Koehn, P. (2004). Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In Frederking, R. E. and Taylor, K., editors, *AMTA*, volume 3265 of *Lecture Notes in Computer Science*, pages 115–124. Springer.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation.

Koehn, P. (2009). *Statistical Machine Translation*. Cambridge University Press.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *ACL '07: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180, Morristown, NJ, USA. Association for Computational Linguistics.

Koehn, P. and Monz, C. (2006). Manual and automatic evaluation of machine translation between european languages. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 102–121, New York City. Association for Computational Linguistics.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54.

Kumar, S. and Byrne, W. (2003). A weighted finite state transducer implementation of the alignment template model for statistical machine translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 63–70, Morristown, NJ, USA. Association for Computational Linguistics.

Kumar, S. and Byrne, W. (2004). Minimum Bayes-risk decoding for statistical machine translation. In *Proc. HLT-NAACL, 2004.*

Kumar, S. and Byrne, W. (2005). Local phrase reordering models for statistical machine translation. In *Proceedings of HLT-EMNLP*.

Kumar, S., Deng, Y., and Byrne, W. (2006). A weighted finite state transducer translation template model for statistical machine translation. *Journal of Natural Language Engineering*, 12(1):35–75.

Lambert, P., Banchs, R. E., and Crego, J. M. (2007). Discriminative alignment training without annotated data for machine translation. In *North American Chapter of the Association for Computational Linguistics, Human Language Technologies Conference (NAACL-HLT07)*.

LDC (2005). Linguistic data annotation specification: assessment of fluency and adequacy of translations, revision 1.5.

Li, Z., Callison-Burch, C., Dyer, C., Ganitkevitch, J., Khudanpur, S., Schwartz, L., Thornton, W. N. G., Weese, J., and Zaidan, O. F. (2009). Joshua: an open source toolkit for parsing-based machine translation. In *StatMT '09: Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 135–139, Morristown, NJ, USA. Association for Computational Linguistics.

Liu, X. A., Byrne, W. J., Gales, M. J. F., Gispert, A. D., Tomalin, M., Woodl, P. C., and Yu, K. (2007). Discriminative language model adaptation for Mandarin broadcast speech transcription and translation. In *Proceedings of IEEE Automatic Speech Recognition and Understanding (ASRU)*, Kyoto, Japan.

Lopez, A. (2008). Statistical machine translation. *ACM Comput. Surv.*, 40(3):1–49.

Macherey, W. and Och, F. J. (2007). An empirical study on computing consensus translations from multiple machine translation systems. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 986–995, Prague, Czech Republic. Association for Computational Linguistics.

Macherey, W., Och, F. J., Thayer, I., and Uszkoreit, J. (2008). Lattice-based minimum error rate training for statistical machine translation. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 725–734, Morristown, NJ, USA. Association for Computational Linguistics.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Mariño, J. B., Banchs, R. E., Crego, J. M., de Gispert, A., Lambert, P., Fonollosa, J. A. R., and Costa-jussà, M. R. (2006). N-gram-based machine translation. *Comput. Linguist.*, 32(4):527–549.

Marton, Y. and Resnik, P. (2008). Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of ACL-08: HLT*, pages 1003–1011, Columbus, Ohio. Association for Computational Linguistics.

Mathias, L. (2007). *Statistical Machine Translation and Automatic Speech Recognition under Uncertainty*. PhD thesis, Johns Hopkins University.

Matusov, E., Ueffing, N., and Ney, H. (2006). Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment. In *Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–40, Trento, Italy.

Melamed, I. (1998). Manual annotation of translational equivalence: The Blinker project.

Melamed, I. D. (2004). Statistical machine translation by parsing. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 653, Morristown, NJ, USA. Association for Computational Linguistics.

Merialdo, B. (1988). Phonetic recognition using hidden Markov models and maximum mutual information training. In *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*, pages 111–114 vol.1.

Mohri, M. (2009). *Weighted automata algorithms*, pages 213–254. Springer.

Mohri, M., Pereira, F., and Riley, M. (1997). *AT&T General-purpose finite-state machine software tools*. http://www.research.att.com/sw/tools/fsm/.

Moore, R. C. (2004). Improving IBM word-alignment model 1. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 518, Morristown, NJ, USA. Association for Computational Linguistics.

Moore, R. C. (2005a). Association-based bilingual word alignment. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 1–8, Ann Arbor, Michigan. Association for Computational Linguistics.

Moore, R. C. (2005b). A discriminative framework for bilingual word alignment. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 81–88.

Moore, R. C. and Quirk, C. (2008). Random restarts in minimum error rate training for statistical machine translation. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 585–592, Manchester, UK. Coling 2008 Organizing Committee.

Nadas, A. (1983). A decision theoretic formulation of a training problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood. *IEEE Trans. Acoust., Speech, Signal Process.*, ASSP-31(4):814–817.

Neal, R. M. and Hinton, G. E. (1999). A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368, Cambridge, MA, USA. MIT Press.

Ng, H. T. and Low, J. K. (2004). Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 277–284, Barcelona, Spain. Association for Computational Linguistics.

Nießen, S. and Ney, H. (2001a). Morpho-syntactic analysis for reordering in statistical machine translation. In *Proceedings of MT Summit VIII*, pages 247–252.

Nießen, S. and Ney, H. (2001b). Toward hierarchical models for statistical machine translation of inflected languages. In *Proceedings of the workshop on Data-driven methods in machine translation*, pages 1–8, Morristown, NJ, USA. Association for Computational Linguistics.

Nießen, S., Och, F. J., Leusch, G., and Ney, H. (2000). An evaluation tool for machine translation: Fast evaluation for MT research. In *Second International Conference on Language Resources and Evaluation (LREC)*.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Normandin, Y. (1991). *Hidden Markov Models, Maximum Mutual Information Estimation and the Speech Recognition Problem*. PhD thesis, McGill University, Montreal.

Normandin, Y. (1996). Maximum mutual information estimation of hidden markov models. In Lee, C., Soong, F., and Paliwal, K., editors, *Automatic Speech And Speaker Recognition — Advanced Topics*, chapter 3, pages 57–82. Kluwer Academic Publishers.

Och, F., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., Jain, V., Jin, Z., and Radev, D. (2004). A smorgasbord of features for statistical machine translation. In *Proceedings of NAACL*.

Och, F., Tillmann, C., and Ney, H. (1999). Improved alignment models for statistical machine translation.

Och, F. J. (1999). An efficient method for determining bilingual word classes. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 71–76, Morristown, NJ, USA. Association for Computational Linguistics.

Och, F. J. (2002). *Statistical Machine Translation: From Single Word Models to Alignment Templates*. PhD thesis, Franz Josef Och.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167.

Och, F. J. and Ney, H. (2000a). A comparison of alignment models for statistical machine translation. In *Proceedings of the 18th conference on Computational Linguistics*, pages 1086–1090.

Och, F. J. and Ney, H. (2000b). Improved statistical alignment models. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 440–447, Morristown, NJ, USA. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 295–302, Morristown, NJ, USA. Association for Computational Linguistics.

Och, F. J. and Ney, H. (2004). The alignment template approach to statistical machine translation. *Comput. Linguist.*, 30(4):417–449.

Odell, J., Ollason, D., Woodland, P., Young, S., and Jansen, J. (1995). *The HTK Book for HTK V2.0*. Cambridge University Press, Cambridge, UK.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Pino, J., Iglesias, G., de Gispert, A., Blackwood, G., Brunning, J., and Byrne, W. (2010). The cued hifst system for the wmt10 translation shared task. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 155–160, Uppsala, Sweden. Association for Computational Linguistics.

Popović, M. and Ney, H. (2004). Improving word alignment quality using morpho-syntactic information. In *In Proceedings of COLING*, page 310.

Przybocki, M., Peterson, K., and Bronsart, S. (2008). Official results of the NIST 2008 "Metrics for MAchine TRanslation" challenge (MetricsMATR08), http://nist.gov/speech/tests/metricsmatr/2008/results/.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142.

Sarikaya, R., Deng, Y., and Gao, Y. (2007). Context-dependent word modelling for statistical machine translation using part-of-speech tags. In *Proceedings of Interspeech 2007*.

Shi, Y. and Wang, M. (2007). A dual-layer CRF based joint decoding method for cascaded segmentation and labeling tasks. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1707–1712, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Sim, K. C., Byrne, W. J., Gales, M. J. F., Sahbi, H., and Woodland, P. C. (2007). Consensus network decoding for statistical machine translation system combination. In *Proceedings of ICASSP-2007*.

Singer, H. and Ostendorf, M. (1996). Maximum likelihood successive state splitting. *Proceedings of ICASSP*, 2:601–604.

Smith, D. A. and Eisner, J. (2006). Minimum-risk annealing for training log-linear models. In *Proceedings of the International Conference on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL), Companion Volume*, pages 787–794, Sydney.

Snover, M., Dorr, B., Schwartz, R., Makhoul, J., Micciula, L., and Weischedel, R. (2005). A study of translation error rate with targeted human annotation. Technical Report LAMP-TR-126,CS-TR-4755,UMIACS-TR-2005-58, University of Maryland, College Park and BBN Technologies*.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings AMTA*, pages 223–231.

Snover, M., Madnani, N., Dorr, B. J., and Schwartz, R. (2009). Fluency, adequacy, or HTER?: exploring different human judgments with a tunable MT metric. In *StatMT '09: Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 259–268, Morristown, NJ, USA. Association for Computational Linguistics.

Stroppa, N., van den Bosch, A., and Way, A. (2007). Exploiting source similarity for SMT using context-informed features. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2007)*, pages 231 – 240.

Taskar, B., Lacoste-Julien, S., and Klein, D. (2005). A discriminative matching approach to word alignment. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 73–80, Morristown, NJ, USA. Association for Computational Linguistics.

Toutanova, K., Ilhan, H. T., and Manning, C. D. (2002). Extensions to HMM-based statistical word alignment models. In *Proceedings of EMNLP*, pages 87–94.

Tromble, R., Kumar, S., Och, F., and Macherey, W. (2008). Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 620–629, Honolulu, Hawaii. Association for Computational Linguistics.

Turian, J., Wellington, B., and Melamed, I. D. (2006). Scalable discriminative learning for natural language parsing and translation. In *In Proceedings of the 2006 Neural Information Processing Systems (NIPS*.

Valtchev, V., Odell, J. J., Woodland, P. C., and Young, S. J. (1997). MMIE training of large vocabulary recognition systems. *Speech Commun.*, 22(4):303–314.

Varea, I. G., Och, F. J., Ney, H., and Casacuberta, F. (2002). Improving alignment quality in statistical machine translation using context-dependent maximum entropy models. In *Proceedings of COLING*, pages 1–7.

Vickrey, D., Biewald, L., Teyssier, M., and Koller, D. (2005). Word-sense disambiguation for machine translation. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 771–778, Morristown, NJ, USA. Association for Computational Linguistics.

Vilar, D., Popović, M., and Ney, H. (2006). AER: Do we need to "improve" our alignments? In *International Workshop on Spoken Language Translation*, pages 205–212, Kyoto, Japan.

Vogel, S., Ney, H., and Tillmann, C. (1996). HMM-based word alignment in statistical translation. In *Proceedings of COLING*, pages 836–841.

Warshall, S. (1962). A theorem on Boolean matrices. *J. ACM*, 9(1):11–12.

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

Weaver, W. (1949/1955). Translation. In Locke, W. N. and Boothe, A. D., editors, *Machine Translation of Languages*, pages 15–23. MIT Press, Cambridge, MA. Reprinted from a memorandum written by Weaver in 1949.

White, J. S., O'Connell, T. A., and Carlson, L. M. (1993). Evaluation of machine translation. In *HLT '93: Proceedings of the workshop on Human Language Technology*, pages 206–210, Morristown, NJ, USA. Association for Computational Linguistics.

Woodland, P. C. and Povey, D. (2002). Large scale discriminative training of hidden Markov models for speech recognition. *Computer Speech and Language*, 16(1):25–47.

Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Comput. Linguist.*, 23(3):377–403.

Xia, F. (2000). The part-of-speech tagging guidelines for the Penn Chinese Treebank (3.0).

Xue, N., Xia, F., Chiou, F.-d., and Palmer, M. (2005). The penn chinese treebank: Phrase structure annotation of a large corpus. *Nat. Lang. Eng.*, 11(2):207–238.

Yamada, K. and Knight, K. (2001). A syntax-based statistical translation model. In *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 523–530, Morristown, NJ, USA. Association for Computational Linguistics.

Yamada, K. and Knight, K. (2002). A decoder for syntax-based statistical MT. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 303–310, Morristown, NJ, USA. Association for Computational Linguistics.

Young, S. J., Odell, J. J., and Woodland, P. C. (1994). Tree-based state tying for high accuracy acoustic modelling. In *HLT '94: Proceedings of the workshop on Human Language Technology*, pages 307–312.

Zens, R. and Ney, H. (2004). Improvements in phrase-based statistical machine translation. In Susan Dumais, D. M. and Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 257–264, Boston, Massachusetts, USA. Association for Computational Linguistics.

Zhang, Y. and Clark, S. (2008). Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pages 888–896, Columbus, Ohio. Association for Computational Linguistics.