Statistical Machine Translation and Automatic Speech Recognition under Uncertainty

Lambert Mathias

A dissertation submitted to the Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland
December 2007

Abstract

Statistical modeling techniques have been applied successfully to natural language processing tasks such as automatic speech recognition (ASR) and statistical machine translation (SMT). Since most statistical approaches rely heavily on availability of data and the underlying model assumptions, reduction in *uncertainty* is critical to their optimal performance.

In speech translation, the uncertainty is due to the speech input to the SMT system whose elements are represented as distributions over sequences. A novel approach to statistical phrase-based speech translation is proposed. This approach is based on a generative, source-channel model of translation, similar in spirit to the modeling approaches that underly hidden Markov model(HMM)-based ASR systems: in fact, our model of speech-to-text translation contains the acoustic models of a large vocabulary ASR system as one of its components. This model of speech-to-text translation is developed as a direct extension of the phrase-based models used in text translation systems. Speech is translated by mapping ASR word lattices to lattices of phrase sequences which are then translated using operations developed for text translation. Efficient phrase extraction from ASR lattices and word and phrase level pruning strategies for speech translation are investigated to reduce uncertainty in translation of speech.

In order to achieve good translation performance it is necessary to find optimal parameters under a particular training objective. Two different discriminative training objective functions are investigated: Maximum Mutual Information (MMI) and Expected BLEU. A novel iterative optimization procedure, using growth transformations is proposed as a parameter update procedure for the training criteria. The translation performance using growth transformation based updates is investigated in detail.

Training a highly accurate ASR systems requires availability of speech corpora with reliable verbatim transcripts. However, accurately transcribed training data are not always available and manually generating them is not always a feasible option. A novel lightly supervised approach to training acoustic models is presented that leverages information from *non-literal* transcripts. In particular, a method for discriminatively training acoustic models using non-literal transcripts is presented. Reliable segments in the acoustic frame are automatically identified and the unreliable frames are filtered during model parameter estimation.

Advisor:	Prof. William J. Byrne
Readers:	Prof. William J. Byrne and Prof. Frederick Jelinek
Thesis Committee:	Prof. William J. Byrne, Prof. Frederick Jelinek, Prof. Gerard G. L. Meyer and Prof. Trac D. Tran

Acknowledgements

First and foremost, I would like to thank my advisor, Prof. William Byrne, without whose guidance and support this dissertation would not have been possible. He has been a good mentor, constantly encouraging me and providing the necessary direction while allowing me the freedom to pursue my ideas. I am indebted to him for the faith he has shown in me over all these years. His work ethic, research insights and patience have certainly influenced me over the years and are qualities that I think will help me in shaping my future career.

I would also like to thank Prof. Frederick Jelinek for giving me the opportunity to work at the Center for Language and Speech Processing (CLSP). Also, thanks go to Prof. Sanjeev Khudanpur for his advice and support over these years. FInally, I would like to extend my thanks to the dissertation committee - Prof. Gerard Meyer and Prof. Trac D. Tran for their time and feedback.

Of course, none of this would have been as enjoyable if it were not for the colleagues and friends I have made here at CLSP. Erin's halloween parties, dinner at the Trombles', David's poetry renditions, Arnab's (food) snobbery, and the various gut busting lunches at Paola's place are few of the reasons that made the 5.5 years I spent in Baltimore entertaining. Equally important was the company of my colleagues at CLSP, notably - Vlasios, Stavros, Ahmad, Veera, Jia, Noah, Yi, Ali, Srihari, Chris, Haolong, Markus, John, Keith, Ariya, Carolina and Binit. I would also like to thank Yonggang Deng and Shankar Kumar for the countless technical discussions and advice they have provided me over the course of my PhD.

I would also like to extend my thanks to the CLSP administrators - Laura, Sue, Monique and Eileen for their excellent support and for keeping us shielded from the various administrative red tape. They have been key in ensuring that CLSP runs smoothly. Special thanks goes to Eiwe, Ian and Justin, the CLSP system administrators for the countless uninterrupted CPU cycles.

Finally, and most importantly, I dedicate this work to Uma, whose love and patience has seen me through all these years.

To Uma

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Statistical modeling approaches to natural language processing have been remarkably successful over the last two decades. Speech recognition, for example, has benefitted vastly from the various statistical learning approaches applied to language modeling, acoustic modeling, pronunciation modeling, data clustering, and hidden Markov models [1, 2]. Statistical modeling techniques have also been applied successfully in other areas of natural language processing, machine translation being one of them.

Most statistical modeling approaches rely heavily on the availability of data. The availability of large amount of text and speech corpora have played a critical role in the success of many of these approaches to natural language processing. However, all of these statistical learning approaches have to deal with *uncertainty*. Sparsity of data, information loss due to noise, ambiguity in natural language, incorrect modeling assumptions and errors during inference are just some sources of uncertainty in statistical modeling approaches.

For example, in the case of automatic speech recognition (ASR), large amounts of speech and text corpora are needed to accurately train acoustic models capable of reliably recognizing a test utterance. It is easy to collect tens of thousands of hours of speech data, for example from broadcast news television feeds. However, manually transcribing all of this speech data accurately is both a time consuming

and a very expensive process. Although an off the shelf speech recognizer can be applied to transcribe the speech and then retrain the model, this approach is prone to errors and might result in unreliable model estimates. A careful approach is needed to correctly train the speech recognizer that can leverage any domain knowledge or additional information into the training process. The statistical learning approach must be able to identify and filter out the unreliable information during model parameter estimation.

Statistical models are prone to errors either due to modeling incorrectness or due to approximations during inference. This problem is compounded further in a cascaded architecture where the output of one statistical system forms the input to another system in the pipeline. In such a complex information processing pipeline the errors in one system adversely affects the performance of the other systems in the pipeline. Speech translation is one such example of a complex information processing system. A typical speech translation architecture consists of an ASR component followed by a statistical machine translation (SMT) component. The ASR system is prone to recognition errors due to pruning of the hypothesis space and various modeling assumptions that go in to the system. Hence, there is uncertainty due to errors which severely degrades the translation performance of the SMT system in the pipeline. Hence, it is generally desirable that the ASR component should pass on as much information as possible for use by the SMT system.

1.2 Research Objective

In this thesis, methodologies to cope with uncertainty in two areas of natural language processing - speech translation and automatic speech recognition are discussed.

A novel approach to statistical phrase-based speech translation is presented. This approach is based on a generative, source-channel model of translation, similar in spirit to the modeling approaches that underly HMM-based ASR systems - in fact, our model of speech-to-text translation contains the acoustic models of a large vocabulary ASR system as one of its components. This model of speech-to-text translation

is developed as a direct extension of the phrase-based models used in text translation systems. Speech is translated by mapping ASR word lattices to lattices of phrase sequences which are then translated using operations developed for text translation. The uncertainty in speech translation is due to the ambiguity in selecting the optimal translation candidate from the ASR lattice for translation. In order to deal with uncertainty, efficient phrase extraction from ASR lattices and word and phrase level pruning strategies for speech translation are investigated.

In order to achieve good translation performance it is necessary to find optimal parameters under a particular training objective. In this thesis, two discriminative training criteria are presented for learning the SMT parameters - maximum mutual information (MMI) criterion and expected BLEU criterion. A novel optimization procedure, using growth transformations is introduced as a parameter update procedure for the training criteria. Furthermore, the problem of training with multiple references is also investigated under the MMI framework. Finally, translation performance is evaluated for the MMI and expected BLEU training procedure and compared with the state of the art minimum error training procedure.

Training Automatic Speech Recognition (ASR) systems require availability of training transcripts for the speech data. Although, it is easy to obtain several hours of speech data, obtaining the corresponding transcripts is a time consuming and costly process. However, partial data transcripts might be available in the domain the loosely correspond to the spoken utterance. For example, in the medical domain the medical reports which are generated as a by-product of the normal medical transcription workflow are available easily. In this work, a novel method for the automatic generation of transcripts from these non-literal transcript is presented. In particular, reliable regions in the transcript that can be used for training acoustic models are identified. Furthermore, a lattice based frame filtering approach is discussed for discriminatively training the acoustic models from these non-literal transcripts.

1.3 Thesis Organization

This thesis is divided into three parts. In Part I, a generative source-channel model for speech translation is presented. Discriminative training for machine translation is also investigated. In Part II, a discriminative training technique for training acoustic models from non-literal transcripts is presented. Finally, in Part III, conclusions and possible directions for future research are explored.

- Chapter 2 begins with a brief description of text translation. The translation template model (TTM) [3] is presented and the BLEU evaluation criteria for evaluating translation performance is discussed. Next, the noisy channel model formulation for speech translation and the various architectures proposed to solve this problem are discussed. A detailed formulation of the speech translation phrase based generative model implemented using weighted finite state machines is presented. Methods for extracting phrases from lattices, pruning at word and phrase level are also discussed.

- Chapter 3 introduces various discriminative training objectives to optimize the MT parameters. Also, Chapter 3 discusses minimum error training and growth transformation based parameter updates in great detail. Chapter 4 shows how the growth transformations can be applied to training the various MT objective functions.

- Translation experiment results for discriminative training using growth transforms and speech translation are reported in Chapter 5 and Chapter 6 respectively.

- Chapter 7 gives a brief overview of the models underlying the ASR system and the various training procedures. Chapter 8 addresses the problem of training acoustic models from non-literal transcripts. An automatic transcript generation strategy is discussed. Also a novel discriminative training procedure based on filtering of unreliable frames is presented. Chapter 9, presents recognition experiments to evaluate the novel training procedure.

- Finally, in Chapter 10 the research goals achieved in this thesis are discussed. Also, future directions for research are outlined.

Part I

Statistical Machine Translation of Speech

Chapter 2

Statistical Phrase-Based Speech Translation

2.1 Overview of Machine Translation of Text

2.1.1 Statistical Machine Translation

Statistical machine translation has achieved significant advancement in recent years. This is attributed to increased availability of parallel corpora and the progress of statistical modeling and automatic evaluation [4]. The most widely used model in statistical MT systems is the source-channel model [5]. The source string, say an English sentence e_1^I, goes through a stochastic noisy channel and generates the target string, say a foreign sentence f_1^J. It typically includes two components: a monolingual language model $P(e_1^I)$, which assigns probabilities to source language strings, and a translation model $P(f_1^J|e_1^I)$ that assigns probabilities to target language strings given a source string. Bilingual sentence pairs are required to learn the statistical parameters of the translation model, and the translation process is usually implemented by source decoding algorithms, for instance, Maximum A Posteriori (MAP)

$$\hat{e}_1^{\hat{I}} = \underset{e_1^I, I}{\operatorname{argmax}}\ P(f_1^J|e_1^I)\ P(e_1^I) \tag{2.1}$$

Translation can be carried out based on word identity [5, 6]. Target words are translated into source words, and source words are reordered to produce grammatical sentences. Translation performance can be improved, though, when based on

phrases [7, 8]. For instance, the target sentence is segmented into target phrases, and each target phrase is translated into a source phrase, and finally the source phrases are reordered to produce the final output source word hypothesis. We next briefly introduce the Translation Template Model (TTM) [3], which is a phrase-based, weighted finite state [9] implementation of the source-channel translation template model. Later, in Section 2.7, the speech translation model is discussed which is a straightforward extension of the TTM.

2.1.2 Translation Template Model

The Translation Template Model (TTM) [3] is a source-channel model of translation with joint probability distribution over all possible segmentations and alignments of target language sentences and their translations in the source language. Translation is modeled as a mapping of source language phrase sequences to target language sentences. The model considers whole phrases rather than words as the basis for translation.

First, the source sentence is segmented into source phrases; source phrases are then mapped onto target phrases, which form the target sentence naturally. Target phrases are allowed to be inserted in the generative process. This corresponds to the deletion of target phrases during translation. Translation is in monotone phrase order. Each of the conditional distributions that make up the model is realized independently and implemented as a weighted finite state acceptor or transducer. Translation of sentences under the TTM can be performed using standard Weighted Finite State Transduce (WFST) operations involving these transducers.

2.2 Evaluating Translation Performance

Evaluating translation is inherently a difficult problem as there are many possible "right" translations in the source language for a given target language sentence. These translations may vary in word choice or in word order even when they use the same words. Although humans can recognize good translations from bad ones, using humans in the evaluation of machine translation is a slow and expensive process. A quick, inexpensive, and language-independent evaluation criterion that correlates

highly with human evaluation,and that has little marginal cost per run is needed. Although automatic translation evaluation is a debatable problem, it is valuable in significantly accelerating MT system development and enabling experiments with many models and algorithms that might otherwise not be tested. Accordingly, many different automatic translation metrics have been proposed in literature - BLEU-score [10], NIST-score [11], F-measure [12], multi-reference Position-independent Word Error Rate (mPER) [13], multi-reference Word Error Rate (mWER) [13] and Translation Error Rate (TER) [14]. Each of these criterion assumes that human references exist for machine-generated translations against which to be compared. It is unlikely that any one of these metrics would perform better than the others for all translation tasks. In this thesis, we will use the BLEU criterion to measure translation performance, which we introduce briefly below.

BLEU [10] is an automatic machine translation evaluation metric that has been widely recognized in the research community. It was adopted in NIST MT evaluation [15] from 2002 to 2005 and has been found to correlate highly with human judgments in terms of fluency and adequacy. BLEU score computes the geometric mean of the modified n-gram precisions $p_n(E, E^+)$ (typically upto n-grams of length 4) between a hypothesis E and a reference sentence E^+, and includes a brevity penalty $\gamma(E, E^+)$, if the hypothesis is shorter than the reference,

$$BLEU(E, E^+) = \gamma(E, E^+) * \exp\left(\frac{1}{4}\sum_{n=1}^{4} \log p_n(E, E^+)\right) \tag{2.2}$$

The brevity penalty is defined as

$$\gamma(E, E^+) = \begin{cases} 1 - \frac{|E^+|}{|E|}, & |E| \leq |E^+| \\ 1, & |E| \geq |E^+| \end{cases} \tag{2.3}$$

BLEU is defined over all the sentences in the test set i.e. the E and E^+ are concatenation of all the hypotheses and reference sentences respectively, in the test set.

2.3 Speech Translation Noisy Channel Model

The goal of a speech translation system is to translate a speech utterance in one language to text in the desired language. A precise mathematical formulation of

speech translation is needed if we are to discuss the problem of speech translation system design.

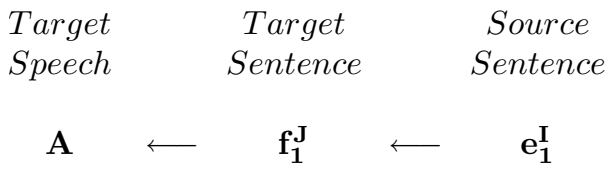

Figure 2.1: Speech Translation noisy-channel model

We begin by formalizing the source-channel model for speech translation [16] shown in Figure 2.1. The source-channel model is a generative model that describes how a source string (e.g English text), e_1^I generates a target speech signal (e.g spoken language Mandarin), A. Strictly speaking, the target sentence f_1^J is not of interest in the decoding process. Mathematically, f_1^J is introduced as a hidden variable in the Bayes' decision rule. The decoding problem is to recover the source sentence e_1^I, which can be obtained by the MAP decoder

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\operatorname{argmax}} P(e_1^I|A) \tag{2.4}$$

$$= \underset{I,e_1^I}{\operatorname{argmax}} \sum_{f_1^J} P(e_1^I, f_1^J|A) \tag{2.5}$$

$$= \underset{I,e_1^I}{\operatorname{argmax}} \sum_{f_1^J} P(A|f_1^J, e_1^I)\ P(f_1^J|e_1^I)\ P(e_1^I) \tag{2.6}$$

$$= \underset{I,e_1^I}{\operatorname{argmax}}\, P(e_1^I) \sum_{f_1^J} P(A|f_1^J)\ P(f_1^J|e_1^I) \tag{2.7}$$

$$\cong \underset{I,e_1^I}{\operatorname{argmax}}\, P(e_1^I) \left\{ \max_{f_1^J} P(A|f_1^J)\ P(f_1^J|e_1^I) \right\} \tag{2.8}$$

$$= \underset{I,e_1^I}{\operatorname{argmax}} \max_{f_1^J} \underbrace{P(e_1^I)}_{\substack{Language \\ Model}} \underbrace{P(f_1^J|e_1^I)}_{\substack{Translation \\ Model}} \underbrace{P(A|f_1^J)}_{\substack{Acoustic \\ Model}} \tag{2.9}$$

In Equation 2.7, it is assumed without loss of generality that the target language speech signal A given the target language string f_1^J is independent of the source string e_1^Ii.e. $P(A|f_1^J, e_1^I) = Pr(A|f_1^J)$. Equation 2.8 follows by approximating the complete likelihood by the likelihood of the most likely path i.e. the summation is replaced by a maximum over the target string f_1^J. The source-channel formulation

neatly decomposes the speech translation problem in to three sub-components : the monolingual source language model that assigns probabilities to the source language word string, the translation model that assigns probabilities to a target language word string given the source language word string, and a target language acoustic model that assigns probability to the target language speech utterance given the target language word string.

2.4 Speech Translation Architectures

The main difference from the text translation case is the introduction of the target language acoustic model, as observed by comparing the noisy-channel formulations in Equation 2.1 and Equation 2.9. The acoustic model component can interact in varying degrees with the translation model component of the system. Depending on the level of interaction speech translation can be broadly classified in to two different architectures - *serial architecture* and *integrated architecture*

2.4.1 Serial Architecture

In a serial architecture the translation process is broken down into two separate decoding steps. First, a conventional speech recognizer in the target language is used to obtain the target language hypotheses. Then, the decoded target language sentence is input to a translation system which finds the best possible translation in the source language. The formulation is as follows:

1. **Target Language ASR Decoding**
   The best target language sentence (possibly multiple hypotheses) f_1^J is searched for given a target acoustic model $P(A|f_1^J)$ and a target language model $P(f_1^J)$

$$\hat{f}_1^{\hat{J}} = \underset{f_1^J, J}{\operatorname{argmax}}\ P(A|f_1^J)\ P(f_1^J) \tag{2.10}$$

2. **Translation of Target Language Sentence**
   This is the translation component where the target language sentence obtained from the ASR decoder $\hat{f}_1^{\hat{J}}$ is input to a statistical text translation system to obtain the best possible translation in the source language.

$$\hat{e}_1^{\hat{I}} = \underset{e_1^I, I}{\operatorname{argmax}}\ P(e_1^I)\ P(\hat{f}_1^{\hat{J}}|e_1^I) \tag{2.11}$$

This approach is clearly sub-optimal as there is no interaction between the ASR and SMT components. Any errors made by the decoder during search are propagated to the SMT component. This can be mitigated to some extent by translating multiple hypotheses (N-best lists) instead of a single 1-best string from the ASR decoder. One disadvantage of using N-best lists is that the translation performance is largely limited by the ASR N-best word error rate. Searching over a much larger hypothesis could help the SMT component to better recover from ASR search errors. Furthermore, the SMT system can possibly exploit the target language information provided by the ASR system - target acoustic model and target language model probabilities - in searching for the best possible target language candidate to be translated.

2.4.2 Integrated Architecture

In an integrated architecture, there is no decoupling of the ASR component and the SMT component . The translation is obtained in a single pass by jointly searching for the best source sentence and target sentence. The formulation of the integrated architecture is specified below:

$$\hat{e_1^I} = \underset{I, e_1^I}{\operatorname{argmax}} \quad P(e_1^I) \left\{ \max_{f_1^J} P(f_1^J | e_1^I) \; P(A | f_1^J) \right\} \tag{2.12}$$

The maximization in Equation 2.12 is over both the target language sentence f_1^J and the source language sentence e_1^J indicating that the search for the target sentence and the source sentence are coupled in this framework.

In this thesis, the tight coupling between the ASR and SMT components is achieved by modeling $P(A|f_1^J)$ as a word lattice. The ASR word lattice compactly represents the distribution over the target language sequences hypothesized by the ASR recognizer and encodes a much larger hypothesis space than the N-best lists. The goal is to aid the SMT component in searching for the best translation in the source language, by allowing it to explore a larger search space of candidate sentences to be translated guided by the target language information (e.g. acoustic model scores). In the following sections we discuss the various approaches to speech translation in more detail.

2.5 Previous Approaches to Speech Translation

Various approaches to speech translation have been investigated - *serial architecture* [17, 18, 19] and *integrated architecture* [20, 16, 21]).

In the serial architecture, there is very little interaction between the SMT component and the ASR component. The JANUS III system [17] uses a N-best list based approach. First the ASR system produces a N-best transcription list. The N-best list is then passed on to a rule-based translation module which generates the final translation. A similar example based translation approach is presented in [18]. The IBM MASTOR system [19] is a speech to speech translation system for translating conversational speech from English to Mandarin, in limited domains. In this approach, a large vocabulary ASR system produces a single transcript which is analyzed by a statistical parser to extract semantic and lexical features. The features are then used in a sentence-level maximum entropy framework to generate the translations.

In an integrated architecture, the objective is to allow the SMT system to search among many likely ASR hypotheses and hopefully produce a better translation than if it had been restricted to the single, best ASR hypothesis. In practice, the close coupling of ASR and SMT can be realized by translating ASR N-Best lists [22, 23] or word lattices [24, 25]. N-Best translation is straightforward: a text-based SMT system can be used without modification to translate each entry, and the resulting translations can be sorted by some combination of ASR and SMT scores. The unified approach to speech translation presented in [22], uses N-best lists for the close coupling between the ASR and SMT systems. A log-linear modeling framework is used to integrate features from the ASR system (acoustic and language model scores) and the features from the SMT system (e.g. word translation probability, word reordering, word length). The translation hypotheses generated are then rescored using additional language models to obtain the final translation. An integrated N-best re-ranking approach using N-best lists is presented in [23], which is similar to the approach presented in [22]. In addition, the authors in [23] explore the effect of parameter optimization in re-ranking the translation hypotheses.

Although it is a complicated modeling and implementation problem, lattice-based translation offers potential advantages over translation of N-Best lists. Lattices provide larger search spaces, as well as detailed, sub-sentential information, such as word-level acoustic and language model scores, that can be passed directly to the SMT system. However it is not trivial to obtain gains in lattice-based translation relative to simply translating the ASR transcription. Initial attempts at incorporating word lattice information in translation did not yield consistent improvements in translation performance [24]. A significant drawback of the approach is that it did not incorporate the ASR language model scores in to the translation process. However approaches have subsequently been developed by which lattices and confusion networks can be translated with improvements in translation quality [26, 25]. The speech translation system presented in [25] models the components as weighted finite state machines and uses a tuple based decomposition of the translation model. Instead of using lattices, the authors in [26] used a confusion network based decoding approach. The confusion network is a compact representation of the word lattice, where the word graph representing the lattice is reduced to a series of adjacent segments, such that the arcs in each segment represents a word and its associated posterior probability [27].

Stochastic Finite-State Transducers (SFSTs) and their generalization - *Weighted Finite State Transducers* (WFSTs) [28, 29], have been applied successfully to various speech and language tasks. Apart from their simplicity, they lend themselves to easy integration of conventional ASR (lattices are easily represented as finite state machines (FSMs)) and SMT systems, thereby allowing the use of standard Viterbi style (with beam pruning) decoding to search for the optimal translation candidate. Various WFST based approaches have been successfully used in speech translation systems [20, 30, 25]. In the AT&T approach, the translation component is decomposed into lexical choice and lexical reordering modules, and integration with the ASR component is achieved using finite state machines [30]. The speech translation approach used in [21], uses SFSTs directly inferred from bilingual data based on specialized maximum likelihood style learning approaches for finite state machines.

2.6 Weighted Finite State Transducers

The speech translation system used in this thesis is a weighted finite state transducer (WFST) based implementation. We give a brief introduction of the various definitions and operations used to model the various translation components following the presentation in [29].

A system $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ is a *semiring* if $(\mathbb{K}, \oplus, \bar{0})$ is a commutative monoid with identity element $\bar{0}$, $(\mathbb{K}, \otimes, \bar{1})$ is a monoid with identity element $\bar{1}$, $\otimes$ distributes over $\oplus$, and $\bar{0}$ is an annihilator for $\otimes$: $\forall a \in \mathbb{K},\ a \otimes \bar{0} = \bar{0} \otimes a = \bar{0}$. Thus, a semiring is a ring that may lack negation. The two most commonly used semirings are the *tropical semiring*: $(\mathbb{R} \cup \{-\infty, +\infty\}, min, +, +\infty, 0)$ and the *log semiring*: $(\mathbb{R} \cup \{-\infty, +\infty\}, \oplus_{\log}, +, +\infty, 0)$, where $x \oplus_{\log} y = -\log(e^{-x} + e^{-y})$.

A WFST $\mathcal{T}$ over a semiring $\mathbb{K}$ is an 8-tuple $\mathcal{T} = (\Sigma^i_{\mathcal{T}}, \Sigma^o_{\mathcal{T}}, Q, I, F, E, \lambda, \rho)$ where: $\Sigma^i_{\mathcal{T}}$ is the finite input alphabet of the transducer; $\Sigma^o_{\mathcal{T}}$ is the finite output alphabet; Q is a finite set of states; $I \subseteq Q$ is a set of initial states; $F \subseteq Q$ is a set of final states; $E \subseteq Q \times (\Sigma^i_{\mathcal{T}} \cup \epsilon) \times (\Sigma^o_{\mathcal{T}} \cup \epsilon) \times \mathbb{K} \times Q$ is a finite set of transitions; $\lambda : I \to \mathbb{K}$ the initial weight function; and $\rho : F \to \mathbb{K}$ the final weight function.

A WFSA $\mathcal{A}$ over a semiring $\mathbb{K}$ is an 8-tuple $\mathcal{T} = (\Sigma_{\mathcal{A}}, Q, I, F, E, \lambda, \rho)$ where: $\Sigma_{\mathcal{A}}$ is the finite vocabulary; Q is a finite set of states; $I \subseteq Q$ is a set of initial states; $F \subseteq Q$ is a set of final states; $E \subseteq Q \times (\Sigma_{\mathcal{A}} \cup \epsilon) \times \mathbb{K} \times Q$ is a finite set of transitions; $\lambda : I \to \mathbb{K}$ the initial weight function; and $\rho : F \to \mathbb{K}$ the final weight function.

Weighted Finite State Acceptors (WFSAs) can be obtained by simply omitting the input or output label of a WFST. Accordingly, $\Pi_1(\mathcal{T})$ denotes the projection of the transducer $\mathcal{T}$ onto the input label (i.e. omitting the output label) and $\Pi_2(\mathcal{T})$ denotes the projection of the transducer $\mathcal{T}$ onto the output label.

Given a transition $e \in E$, let $p[e]$ denote its origin or previous state, $n[e]$ the destination state, $i[e]$ the input label, $o[e]$ the output label, and $w[e]$ its weight. In case the automaton is a WFSA, $l[e]$ denotes simply the label of e. A path $\pi = e_1, e_2, \ldots, e_k$ is an element of E^* with consecutive transitions : $n[e_{i-1}] = p[e_i],\ i = 1, 2, \ldots, k$. Here

E^* is the set of all strings consisting of symbols in E. Also, for a path π, we can define $n[\pi] = n[e_k]$, $p[\pi] = n[e_1]$ and $w[\pi] = w[e_1] \otimes \cdots \otimes w[e_k]$ and $l[\pi] = l[e_1], \ldots, l[e_k]$.

For an weighted finite automaton (WFA) A, a path π is a complete path if $n[\pi] \in I$ and $e[\pi] \in F$. Henceforth, we will define the set of all such complete paths for the WFA A as Π_A. Also, the language of the WFA A is denoted by L_A, where:

$$L_A = \{s_1^n \big|\ l[\pi] = s_1^n,\ \pi \in \Pi_A\}$$

Also, a transducer $\mathcal{T}$ is regulated if the weight associated by $\mathcal{T}$ to any pair of input output strings (x, y) is given by:

$$[[\mathcal{T}]](x, y) = \bigoplus_{\pi \in \Pi_{\mathcal{T}}} \lambda[p[\pi]] \otimes w[\pi] \otimes \rho[n[\pi]]$$

We now define some WFST operations on transducers used in the speech translation model:

1. **Composition**

   The composition of two weighted transducers $A \circ B$ is also a weighted transducer defined for all input-output label pairs (x, y)

   $$[[A \circ B]](x, y) = \bigoplus_{z \in \Sigma_A^o \cap \Sigma_B^i} A(x, z) \otimes B(z, y)$$

2. **Best Path**

   The best path of a transducer $\mathcal{T}$ is the path π in the transducer with minimal cost. Usually, the transducer is represented in the tropical semiring. So we have

   $$BestPath(\mathcal{T}) = \min_{\pi \in \Pi_{\mathcal{T}}} \lambda[p[\pi]] \otimes w[\pi] \otimes \rho[n[\pi]]$$

   where, $P(I, F)$ is the set of all complete paths in $\mathcal{T}$.

The composition and best path operations can be defined analogously for acceptors.

2.7 Generative Models for Translation of Speech

2.7.1 Statistical Phrase-Based Speech Translation

The statistical speech translation model is a simple extension of the text translation system described in Section 2.1.2. Speech translation is formulated as a gen-

erative source-channel model of translation that describes how the source sentence e_1^I generates the target language acoustics A. The transformation is effected via a series of transformative operations specified by conditional probability distributions.

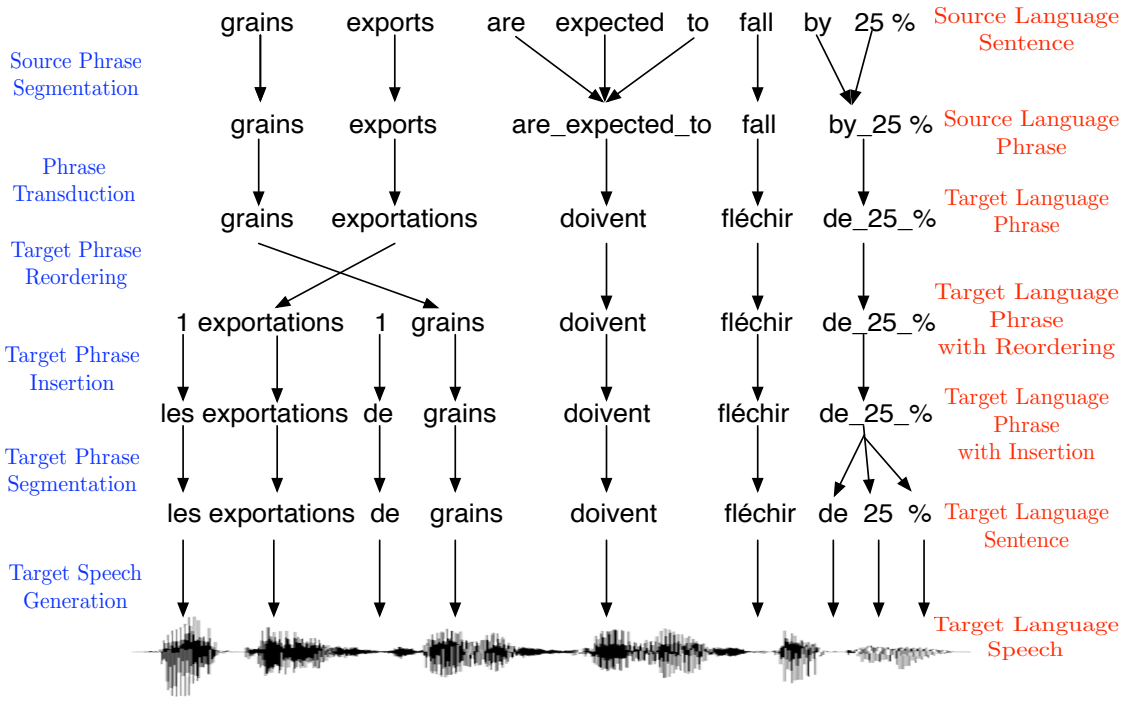

Figure 2.2: An example showing the generative process for speech translation: a source language sentence is transformed in to a target language speech utterance via a series of intermediate transformations

Figure 2.2 illustrates the generative process through which the source language sentence is transformed in to the target language speech. In this example, the source language model generates the source language sentence *grain exports are expected to fall by 25 %*. This sentence is segmented into a source phrase sequence: *grain exports are_projected_to fall by_25_%* under the Source Phrase Segmentation Model. This source phrase sequence is then translated into a target language phrase sequence *grains exportations doivent fléchir de_25_%* under the Phrase Transduction Model. The target phrase sequence obtained is still in source language order. The target phrase sequence is then reordered into the target language order *1 exportations grains 1 doivent fléchir de_25_%* under the Target Phrase Reordering Model. The integer markings indicate placeholders for the length of the target phrases to

be inserted. The insertion is governed by the Target Phrase Insertion Model which gives us the target phrase sequence *les exportations de grains doivent fléchir de_25_%*. Next, the target phrase is segmented in to a target word sequence *les exportations de grains doivent fléchir de 25 %*. Finally, the target word sequence is transformed in to the corresponding speech utterance.

The joint distribution over the target language speech A and the source language sentence e_1^I is specified as:

$$
\begin{aligned}
P(A, f_1^J, v_1^R, y_1^K, x_1^K, u_1^K, e_1^I) = & \\
P(A|f_1^J) & \quad Target\ Acoustic\ Model \\
P(f_1^J|v_1^R) & \quad Target\ Phrase\ Segmentation\ Model \\
P(v_1^R|y_1^K) & \quad Target\ Phrase\ Insertion\ Model \\
P(y_1^K|x_1^K, u_1^K) & \quad Target\ Reordering\ Model \\
P(x_1^K|u_1^K) & \quad Phrase\ Transduction\ Model \\
P(u_1^K|e_1^I) & \quad Source\ Phrase\ Segmentation\ Model \\
P(e_1^I) & \quad Source\ Language\ Model
\end{aligned}
\tag{2.13}
$$

Each of the conditional distributions are modeled independently as WFSTs and are presented in detail in Section 2.7.3.

2.7.2 Phrase Pair Inventory

The Speech Translation Model relies on an inventory of target language phrases and their source language translations. The phrase pairs are not unique, which implies there can be multiple translations of phrases in either language. Before we go on to describe the phrase pair extraction procedure we will cover a few preliminary definitions.

A phrase in the context of machine translation is defined as a substring (i.e. a contiguous sequence of words). In order to extract phrase pairs we begin with *parallel training data* which is a corpus of target language sentences along with the corresponding source language translations. Next we define a set of word alignment between each sentence pair in the parallel training data. An alignment is simply

a function that maps a word position in the source (target) language sentence to a word position in the target (source) language sentence. The word alignment is obtained from a corpus of aligned sentence pairs in an unsupervised fashion using Expectation Maximization (EM) [5, 31, 32, 33].

Given these word alignments we can now extract phrase pairs which align well according to a set of heuristics [13]. Let $u = f_j^{j+m}$ be a length m target phrase and $v = e_i^{i+n}$ be a length n source phrase. Then for a sentence pair (f_1^J, e_1^I) the phrase pair inventory $\mathcal{BP}$ according to some underlying word alignment A is specified by the set:

$$\mathcal{BP}(f_1^J, e_1^I, A) = \{(u, v) : \forall (i', j') \in A : j \leq j' \leq j + m \wedge i \leq i' \leq i + n, f_{j'} \leftrightarrow e_{i'}\} \tag{2.14}$$

To restrict the memory requirements of the model, we extract only the phrase pairs which have at most five words in the target phrase. Furthermore, we restrict the phrase-pair extraction to only a subset of target phrases that are actually needed during translation i.e. we extract the phrase-pairs from the parallel text only if the target phrase occurs in the target language sentences we want to translate. Finally, we augment the phrase-pair with single source word to target word translations obtained by retaining the best alignment under A, so as to get complete coverage all single word translations.

Before applying the phrase-pair extraction procedure to build the phrase inventory, we first have to define a candidate set of target language phrases. However, in the case of speech translation, where we are interested in translating from target language word lattices, extracting phrases is a non-trivial task. Extracting all possible substrings from a lattice can be computationally very expensive. Later, in Section 2.8.4 we address this issue and discuss an efficient procedure for extracting phrases from a lattice.

2.7.3 Speech Translation System Components

Figure 2.3 illustrates the generative process for translation of speech and the underlying components of the generative model. A (simplified) description of the generative process has the following steps.

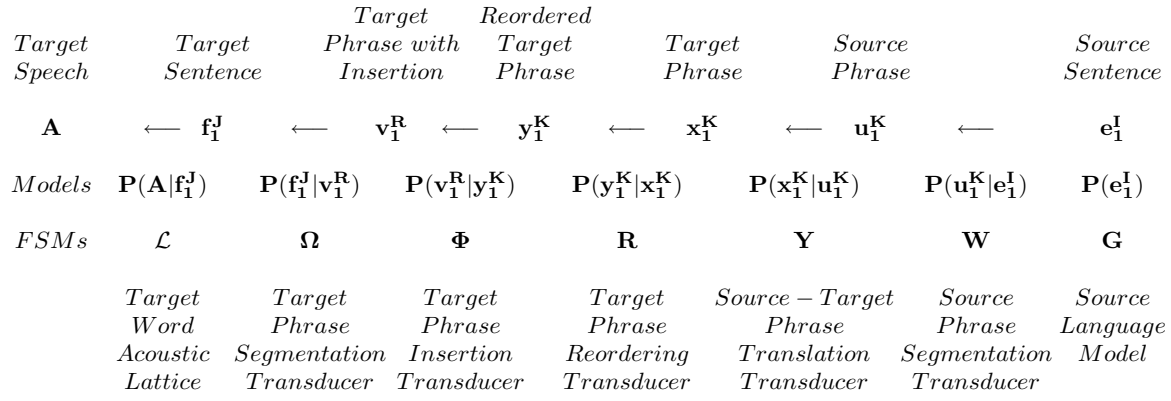

Figure 2.3: A detailed view of the component models used in the generative process for speech translation

Step 1 Source Language Model

The source language sentence $e_1, \ldots, e_I$ is generated by the *Source Language Model*, $P(e_1^I)$. The source language model is modeled as an n-gram language

$$P(e_1^I) = \prod_{i=1}^{I} P(e_i|e_{i-n+1}^{i-1}) \tag{2.15}$$

trained on monolingual source language data.

Step 2 Source Phrase Segmentation

The source language sentence e_1^I is segmented into a series of source language phrases, u_1^K. There are many possible sequences of phrases that can be derived from a single sentence, as defined by the *Source Phrase Segmentation* distribution, $P(u_1^K|e_1^I)$. This distribution assigns a uniform likelihood to all phrase segmentations of the source sentence that can be obtained using the phrase inventory. In practice, however, this probability is degenerate i.e.

$$P(u_1^K|e_1^I) = 1(e_1^I, u_1^K) \tag{2.16}$$

where $1(e_1^I, u_1^K)$ enforces the constraint that the words in e_1^I agree with the words in u_1^K.

Step 3 Phrase Transduction

The sequences of source language phrases u_1^K are translated into the target language phrase sequence v_1^K under the *Phrase Transduction* distribution $P(v_1^K|u_1^K)$. The target phrases are conditionally independent of each other

and depend only on the source language phrase which generated each of them. This gives us the following distribution:

$$P(v_1^K|u_1^K) = \prod_{k=1}^{K} P(v_k|u_k) \tag{2.17}$$

The probability $P(v|u)$ is simply a maximum likelihood estimate of the frequency of occurrence of the phrase pair (v, u) in the phrase-pair inventory.

Step 4 Target Language Reordering

The target language phrase sequence we obtained after the transduction step is still in source language order. However, there is no reason to believe that the target language has the same ordering as the source language. This is certainly the case in language pairs such as Chinese-English and Arabic-English where there is long distance phrase movement between the languages. Hence we need to construct a model that basically reorders the target language phrase sequence x_1^K in to a sequence y_1^K in target language order. The reordering is controlled by a parameterized *Target Phrase Reordering* distribution $P(y_1^K|x_1^K, u_1^K)$ that expresses a preference for swapping target phrases within a certain window [34]. Given an input phrase sequence x_1^K, we associate a unique jump sequence b_1^K with each permissible output phrase sequence y_1^K. The jump b_k measures the displacement of the kth phrase x_k. The jump sequence b_1^K is constructed such that the sequence y_1^K is a valid permutation of x_1^K. In our model we constrain b_1^K so that $b_k \in \{0, +1, -1\}$. We can then redefine the model in terms of the jump sequence

$$P(y_1^K|x_1^K, u_1^K) = \prod_{k=1}^{K} P(b_k|x_k, u_k) \tag{2.18}$$

Step 5 Target Phrase Insertion

The processes described thus far allow a mapping of a source language sentence into a reordered sequence of target language phrases, whose order is the phrase order of the target language. The constraint that the target language phrase sequence have the same number of phrases as the source language phrase sequence is overly restrictive. Our goal is to construct a model to allow insertion of target language phrases anywhere in the reordered source language phrase sequence. This process is governed by the *Target Phrase Insertion* probability distribution $P(v_1^R|y_1^K)$ such that the likelihood of inserting a phrase is

inversely proportional to the number of words in the phrase. Therefore, a greater penalty is assigned for the insertion of longer phrases. Given an input phrase sequence y_1^K, we associate a sequence c_1^K such that

$$P(v_1^R|y_1^K) = \prod_{k=1}^{K} P(c_k|y_k) \tag{2.19}$$

where

$$c_k = y_k \cdot p_k[1] \cdot p_k[2] \cdots p_k[i] \quad p_k[i] \in \{1, 2, \ldots, M\}$$

Here, i is the number of phrases being spontaneously inserted immediately following target phrase y_k and $p_k[i]$ specifies the length of the target phrases being inserted and M is the maximum number of words in an inserted phrase. Also, $|c_1^K| = |v_1^R|$. For example, if $v_k = terms\_of\_reference$ and $c_k = terms\_of\_reference \cdot 1 \cdot 2 \cdot 3$, this specifies that three phrases are spontaneously inserted after v_k of lengths one word ,two words and three words respectively. We define

$$P(c_k|y_k) = \begin{cases} \alpha_0, & c_k = y_k \cdot \epsilon \\ \alpha^{\sum_{i=1}^{M} p_k[i]}, & c_k = y_k \cdot p_k \\ 0, & else \end{cases} \tag{2.20}$$

The tendency towards phrase insertion is controlled by a single parameter, the *Phrase Exclusion Probability* (PEP) α which is set manually to a specified value. Typical range for the PEP is $0 \leq \alpha \leq 1$. Also, α_0 is chosen such that $\sum_k P(c_k|y_k) = 1$.

Step 6 Target Phrase Segmentation

The target language phrase sequences which is now in target language order are transformed to target language word sequences, $f_1, f_2, \ldots, f_J$, under the *Target Phrase Segmentation* distribution, $P(f_1^J|v_1^R)$. In practice, this is a degenerate transformation which maps every target phrase sequence to its unique word sequence i.e.

$$P(f_1^J|v_1^R) = 1(f_1^J = v_1^R) \tag{2.21}$$

where $1(f_1^J = v_1^R)$ enforces the requirement that words in the target sentence agree with those in the target phrase sequence.

Step 7 Target Speech Generation

Finally, the target language sentence f_1^J is transformed in to the target language speech which is governed by the acoustic model probability $P(A|f_1^J)$. In practice, $P(A|f_1^J)$ is modeled as a word lattice.

Taken together, these distributions form a joint probability distribution over the source and target language sentences, and over the possible intermediate source and target phrase sequences as given in in Equation 2.13.

2.8 Speech Translation under the Generative Model

Given a target language acoustic sequence A we can obtain the source language translation using the MAP decoder:

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\text{argmax}}\{ \max_{f_1^J \in \mathcal{L}} \max_{v_1^R, y_1^K, x_1^K, u_1^K, K} P(A, f_1^J, v_1^R, y_1^K, x_1^K, u_1^K, e_1^I) \} \tag{2.22}$$

$$= \underset{I,s_1^I}{\text{argmax}}\{ \max_{f_1^J \in \mathcal{L}} \max_{v_1^R, y_1^K, x_1^K, u_1^K, K} \underbrace{P(A|f_1^J)}_{\substack{Acoustic \\ Lattice}} \underbrace{P(f_1^J, v_1^R, y_1^K, x_1^K, u_1^K, e_1^I)}_{\substack{Text \\ Translation}} \} \tag{2.23}$$

where $P(A, f_1^J, v_1^R, y_1^K, x_1^K, u_1^K, e_1^I)$ is the joint probability distribution defined in Equation 2.13. In the above decoder, we maximize over both the source and target language jointly, which indicates that the ASR component is integrated in to the translation search. Furthermore, the generative model for speech translation is a straightforward extension of the text SMT system, we simply add the acoustic model as one of the components in the generative model, to describe how the spoken language is generated from the source text.

2.8.1 Proper Inclusion of the Target Language Model

In the case of text translation, we are given the input target string f_1^J that we want to translate. Translating from speech poses a different modeling problem - the SMT system is given a lattice of target strings, and is asked to select one of these to translate. Given this, it would be appropriate to replace the acoustic lattice with a lattice containing posterior distributions over target language sentences: $P(f_1^J|A) = P(A|f_1^J)\ P(f_1^J)\ /\ P(A)$. The main benefit is that the target language sentence

selection would benefit by considering the strong monolingual target language model $P(f_1^J)$ used in ASR [35]. Simply replacing the acoustic score lattice in Equation 2.22 by a lattice containing posterior scores, yields the following decoder

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\text{argmax}}\{ \max_{f_1^J \in \mathcal{L}} \max_{v_1^R, y_1^K, x_1^K, u_1^K, K} \underbrace{P(A|f_1^J)\ P(f_1^J)}_{\substack{Target \\ Lattice}} \underbrace{P(f_1^J, v_1^R, y_1^K, x_1^K, u_1^K, e_1^I)}_{\substack{Text \\ Translation}} \} \quad (2.24)$$

Whether the posterior distribution is actually added as $P(A|f_1^J)\ P(f_1^J)$ or simply as $P(A|f_1^J)$ it is clear that, relative to the correct model, there is a free-floating $P(f_1^J)$ to be accounted for. However, this quantity does not appear in either translation model training from parallel text or in translation from text; it only appears in translating from speech, where there is uncertainty as to what target language sentence should be translated. Operationally, only the weights of target language sentences within the lattice change.

2.8.2 Translation using Weighted Finite State Machines

Referring to Figure 2.3, the component distributions are formulated so that each can be implemented as a WFST. To translate a given target language speech utterance A into the source language, we construct an acceptor $\mathcal{L}$ containing target acoustic model scores $P(A|f_1^J)\ P(f_1^J)$. In theory, we could then create a lattice of translations via the following sequence of FSM compositions

$$\mathcal{T} = G \circ W \circ Y \circ R \circ \Phi \circ \Omega \circ \mathcal{L} \quad (2.25)$$

In order to obtain the translation $\hat{s}_1^{\hat{I}}$, we simply project the transducer $\mathcal{T}$ onto the output labels and find the highest scoring path (or lowest cost path) in the composite translation network. In terms of the WFST operation this is a best path search.

$$\hat{s}_1^{\hat{I}} = BestPath[\Pi_1(\mathcal{T})] \quad (2.26)$$

2.8.3 Transforming ASR Word Lattices into Phrase Lattices

The framework presented thus far takes as input the ASR word lattice and produces a sentence translation in the source language. The TTM however, is a phrase based model and in practice the ASR word lattice is first converted in to a target

phrase sequence lattice. It is this target phrase sequence lattice that is the input to the translation system. The lattice of target phrase sequences is generated by applying the target phrase segmentation transducer to the ASR word lattice : as the target sentences are segmented into phrase sequences, the acoustic scores and language model scores are retained. Figure 2.4 illustrates the conversion of a Spanish language ASR lattice in to a Spanish language phrase sequence lattice. Substituting $P(A|v_1^R) = P(A|f_1^J)P(f_1^J)P(f_1^J|v_1^R)$ in Equation 2.27 gives the phrase based MAP decoder in Equation 2.28.

$$\hat{e}_1^{\hat{I}} = \underset{I,e_1^I}{\text{argmax}}\left\{\max_{f_1^J,v_1^R,y_1^K,x_1^K,u_1^K,K} P(f_1^J)P(A|f_1^J)P(f_1^J|v_1^R)\ P(v_1^R,y_1^K,x_1^K,u_1^K,e_1^I)\right\} \quad (2.27)$$

$$= \underset{I,e_1^I}{\text{argmax}}\left\{\max_{v_1^R,y_1^K,x_1^K,u_1^K,K} \underbrace{P(A|v_1^R)}_{\substack{Target\\ Phrase\ Lattice}} \quad \underbrace{P(v_1^R,y_1^K,x_1^K,u_1^K,e_1^I)}_{\substack{Phrase\\ Translation}}\right\} \quad (2.28)$$

In terms of the corresponding transducer we build the target phrase sequence lattice Q via the following operation:

$$Q = \Pi_1(\Omega \circ \mathcal{L})$$

The projection of the composition onto the output labels gives us the desired target phrase lattice, with the appropriate target acoustic scores on each of the arcs in the lattice. So, now we can obtain the source language translation via the composition:

$$\hat{s}_1^{\hat{I}} = BestPath[\Pi_1(G \circ W \circ Y \circ R \circ \Phi \circ Q)] \quad (2.29)$$

2.8.4 Phrase Extraction from a Lattice

A simple method for extracting phrases from a lattice is to simply traverse the lattice arcs and output the substrings encountered thus far. However, this approach is very slow and inefficient due to the large number of word sequences the lattice encodes. A more efficient algorithm to extract phrases from the lattice is outlined below.

Let $\mathcal{L}$ be the acoustic word lattice represented as a WFST. Let $C(w_1^k|\mathcal{L})$ represent the count of the word sequence (phrase) w_1^k in $\mathcal{L}$. Then

$$C(w_1^k|\mathcal{L}) = \sum_{\pi \in \mathcal{L}} C(w_1^k|\pi)\ [[\mathcal{L}]](w_1^k) \quad (2.30)$$

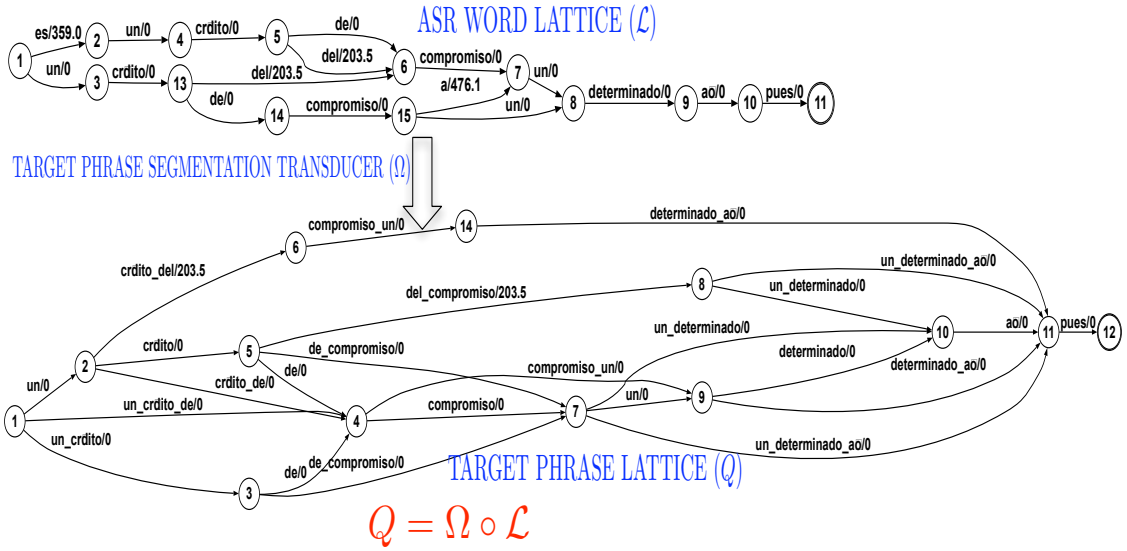

Figure 2.4: Transforming ASR word lattice $\mathcal{L}$ to a target sequence phrase lattice Q via the composition $\Omega \circ \mathcal{L}$

where, $C(w_1^k|\pi)$ denotes the number of occurences of the sequence w_1^k in the path π and $[[\mathcal{L}]](w_1^k)$ denotes the weight assigned by the WFST $\mathcal{L}$ to the sequence w_1^k.

The GRM Library [36] tool *grmcount* provides and efficient implementation of the above procedure. Given an input WFST $\mathcal{L}$, *grmcount* generates a count automaton that contains all the phrases observed in the WFST along with their associated counts.

2.9 ASR Lattice Pruning for Translation

We need to control for the complexity of the ASR lattice, since composing the ASR word lattice with the translation components will most likely result in an explosion in the number of paths in the resulting transducer. In order to reduce the computational effort and memory requirements, it is necessary to prune the unlikely translation candidates from the ASR lattice. Pruning can either be at the word level or at the phrase level.

2.9.1 Forward-Backward Pruning

Forward-backward pruning is a posterior based pruning strategy that retains strings that are likely under the target acoustic and language models. The important point is that the pruning strategy accounts for entire path scores and not just partial scores.We will now describe the lattice forward-backward based pruning procedure [27].

We start with a WFSA $\mathcal{L}$. Borrowing the notation from Section 2.6 , the WFSA has a designated start state $q_0 \in I$ and an end state $q_f \in F$. Let $i[e]$ be the label associated with edge $e \in E$; $w[e]$ is the weight associated with the edge e and is the combination of the target acoustic probability and the target language model probability; $p[e]$ is the previous state and $n[e]$ is the next state for the edge e.
We then define a forward probability, which is the sum of partial path probabilities of all the partial paths starting from q_0 and ending at node $n[e]$.

$$F(n[e]) = \sum_{p[e]:e\in E} F(p[e])\ w[e] \tag{2.31}$$

We also define a backward probability, which is the sum of all partial paths starting from the current node $p[e]$ and ending at q_f.

$$B(p[e]) = \sum_{n[e]:e\in E} B(n[e])\ w[e] \tag{2.32}$$

We can then obtain the posterior probability $q(l[e]|\mathcal{L})$ where $l[e]$ is the label associated with edge e:

$$q(l[e]|\mathcal{L}) = \frac{F(p[e])\ w(e)\ B(n[e])}{B(q_0)} \tag{2.33}$$

The probability $q(l[e]|\mathcal{L})$ is the posterior probability of the arc and is the sum of the probabilities of all paths in the word graph that pass through the edge e with label $l[e]$. Hence, $q(l[e]|\mathcal{L})$ accounts for complete path scores in the lattice. We can now assign a threshold τ and prune all arcs that fall below this threshold i.e. prune an edge e with label $l[e]$ if $q(l[e]|\mathcal{L}) < \tau$. This will restrict the size of the lattice and retain only the high probability arcs as translations candidates.

2.9.2 Phrase posterior pruning

As discussed in Section 2.8.3, translation is from a lattice of target phrase sequences. In order to build this target phrase sequence lattice phrases need to be

extracted form the lattice. An efficient procedure for this was discussed in Section 2.8.4. The goal is to build a phrase sequence lattice as described in Section 2.8.3 consisting of only high confidence phrases. Consequently, during the phrase extraction procedure, the low confidence phrases need to be filtered out.

The posterior probability of the phrase can easily be calculated by simply normalizing the counts in Equation 2.30. Let $p(w_1^k|\mathcal{L})$ denote the posterior probability of the phrase w_1^k in the ASR lattice $\mathcal{L}$. The posterior probability is specified as:

$$p(w_1^k|\mathcal{L}) = \frac{C(w_1^k|\mathcal{L})}{\sum_{w_1^k \in \mathcal{L}} C(w_1^k|\mathcal{L})} \tag{2.34}$$

A threshold γ can now be defined below which the phrases are pruned i.e. extract a phrase from a lattice if $p(w_1^k|\mathcal{L}) > \gamma$.

2.10 Summary

In this chapter, we presented a generative source-channel model for translation of speech. The generative source-channel model is a simple and straightforward extension of the text translation system and describes how the source language sentence generates the target language acoustic utterance througha series of underlying transformations. Next, we presented a detailed description of the conditional distributions underlying the generative process. We also showed that modeling the conditional distributions as WFSTs allows us to build a translation newtork using just WFST composition; the final translation is then a search for the highest scoring path in this network. We emphasize that the WFST based approach to speech translation neatly avoids the difficult problem of developing specialized statistical translation decoders that can process ASR word lattices. That problem is replaced instead by a modeling problem, namely how to extract phrase sequences from word lattices. We outlined our approach to extracting phrase sequences from lattices and methods for transforming the ASR word lattice into a target phrase sequence lattice. In order to filter the low confidence hypotheses in the ASR lattice, we presented a forward-backward pruning procedure to reduce the lattice density. We also introduced a method to extract high quality phrases directly from the target phrase

sequence lattice.

This concludes the discussion of speech translation until Chapter 6. In the next chapter we will address the problem of parameter estimation for SMT.

Chapter 3

Parameter Optimization for Machine Translation

3.1 Discriminative Objective Functions for MT

3.1.1 Parameter Estimation

Let us assume we have a model of translation with the parameter set $\theta = \{\theta_1, \theta_2, \cdots, \theta_Q\}$. Furthermore, let $\{(\mathbf{f}_1, \mathbf{e}_1^+), (\mathbf{f}_2, \mathbf{e}_2^+), \ldots, (\mathbf{f}_S, \mathbf{e}_S^+)\}$ be a corpus of S sentence pairs where $\mathbf{f}_s$ is the target language sentence and $\mathbf{e}_s^+$ is the corresponding reference translation in the source language. Let $F(\theta)$ be a real-valued scalar function that characterizes particular aspects of a training procedure that we wish to use in order to optimize our parameters θ. The optimization problem of interest is then formally stated as:

$$\hat{\theta} = \underset{\theta}{\text{argmax}}\ F(\theta) \tag{3.1}$$

This is known as the *training problem*: how do we estimate the parameters θ of the translation model given the data ?

3.1.2 Maximizing the posterior distribution

As an alternative to the source-channel maximum likelihood framework we could choose to maximize the posterior distribution $p_\theta(\mathbf{e_s}|\mathbf{f_s})$ directly. The training objective $F_{MMI}(\theta)$ is equivalent to maximum mutual information (MMI) training crite-

rion [37] and is specified by:

$$F_{MMI}(\theta) = \sum_{s=1}^{S} \log p_\theta(\mathbf{e_s^+}|\mathbf{f_s}) = \sum_{s=1}^{S} \log \frac{p_\theta(\mathbf{e_s^+}, \mathbf{f_s})}{\sum_{\mathbf{e_s}} p_\theta(\mathbf{e_s}, \mathbf{f_s})} \tag{3.2}$$

Ideally, we would like to find a θ, such that

$$\mathbf{e_s^+} = \underset{\mathbf{e_s}}{\operatorname{argmax}}\ p_\theta(\mathbf{e_s}|\mathbf{f_s}) \tag{3.3}$$

In practice this is not possible, because the model $p_\theta(\mathbf{e_s}|\mathbf{f_s})$ is an approximation of the true unknown distribution $P(\mathbf{e_s}|\mathbf{f_s})$. Instead, we settle for maximizing $p_\theta(\mathbf{e_s^+}|\mathbf{f_s})$; with the idea (in theory) that if $p_\theta(\mathbf{e_s^+}|\mathbf{f_s})$ is increased sufficiently, then $p_\theta(\mathbf{e_s^+}|\mathbf{f_s}) > p_\theta(\mathbf{e_s}|\mathbf{f_s}),\ \forall \mathbf{e}_s \neq \mathbf{e}_s^+$, which ensures the desired result Equation 3.3.

The MMI training criterion attempts to separate the class conditional probabilities of the correct class $\mathbf{e_s^+}$ from the alternative classes e_s, so that it is better able to recover from incorrect modeling assumptions. Since the objective function is continuous and differentiable with respect to the parameters we can use any gradient based approach to optimize this objective function.

3.1.3 Minimizing Direct Loss

In machine translation tasks, the translation performance is evaluated using specific evaluation metrics such as BLEU [10], METEOR [38], and NIST score [11]. The training criterion discussed in Section 3.1.2 does not explicitly model the translation performance under a specific evaluation metric. Instead, it relies on the ability of the estimate $p_\theta(\mathbf{e_s}|\mathbf{f_s})$ to correctly rank the space of translations, while achieving separation between the correct class and the incorrect ones. We want to adjust the model parameter set θ so as to minimize the decision errors under specific evaluation metrics.

Let us posit the existence of a loss function $L(\mathbf{e}^+, \mathbf{e})$ that assesses a penalty for choosing the hypothesis $\mathbf{e}$ when the reference $\mathbf{e}^+$ is correct. Furthermore, let us assume that this loss function is the sum of losses over individual sentences in the test corpus i.e. $L(\mathbf{e}^+, \mathbf{e}) = \sum_{s=1}^{S} L(\mathbf{e_s^+}, \mathbf{e_s})$. One such training objective that

minimize the direct loss is given by

$$F_{MET}(\theta) = -\sum_{s=1}^{S} L(\underset{\mathbf{e_s}}{\text{argmax}}\ p_\theta(\mathbf{e_s}|\mathbf{f_s}), \mathbf{e_s^+}) \tag{3.4}$$

The negative sign indicates that in order to minimize the loss we need to maximize $F_{MET}(\theta)$. This training criterion, adjusts the model parameters so as to minimize the loss incurred by choosing the best hypothesis (1-best) according to the current model. Also, since the objective function is not smooth and differentiable with respect to the parameters, it cannot be optimized using gradient based techniques. In Section 3.3 we discuss the line search based approach which is a gradient free optimization approach and is the current state of the art in MT.

3.1.4 Minimizing Expected Loss

Instead of looking at only the top hypothesis, we can also choose to minimize the *expected loss* under $p_\theta(\mathbf{e_s}, \mathbf{f_s})$ across all hypotheses $\mathbf{e_s} \in \mathbf{E}$. The objective function of interest is then

$$F_{MBR}(\theta) = -\sum_{s=1}^{S} \sum_{\mathbf{e_s} \in \mathbf{E}} L(\mathbf{e_s^+}, \mathbf{e_s})\ p_\theta(\mathbf{e_s}|\mathbf{f_s}) \tag{3.5}$$

If we assume the loss function to be the 0/1 loss function

$$L(\mathbf{e_s^+}, \mathbf{e_s}) = \begin{cases} 0 & \text{if } \mathbf{e_s} = \mathbf{e_s^+} \\ 1 & \text{else} \end{cases}$$

then minimizing the expected loss in Equation 3.5 is equivalent to maximizing the posterior in Equation 3.2. Also, the objective function is continuous and differentiable and we can use any gradient based approach to optimize this objective function.

3.1.5 Enumerating the joint distribution $p_\theta(\mathbf{e_s}, \mathbf{f_s})$

Throughout this discussion, the following form of the joint distribution is assumed

$$p_\theta(\mathbf{e_s}, \mathbf{f_s}) = \prod_{q=1}^{Q} \Phi_q(\mathbf{e_s}, \mathbf{f_s})^{\theta_q} \tag{3.6}$$

In this framework, we have a set of Q feature functions $\Phi_q(\mathbf{e_s}, \mathbf{f_s}),\ q = 1 \ldots Q$. For each feature function there exists a model parameter $\theta_q, q = 1 \ldots Q$ that we are interested in estimating. For example, the feature can be the various components of the MT generative model - source language model log probability, translation model log probability etc. The features need not be a component of the generative model. Additional features such as length of the sentence, additional larger language models can be introduced. The *modeling problem* involves defining a suitable set of feature functions that captures various aspects of the translation task and is beyond the scope of this discussion.

3.2 Previous Approaches to Discriminative Training

Automatic evaluation techniques such as BLEU [10] and evaluation-specific optimization [39] have significantly improved machine translation performance. The authors in [40] proposed a framework for MT based on directly maximizing the posterior probability $p_\theta(\mathbf{e}|\mathbf{f})$ using maximum entropy modeling techniques. A small set of feature functions were defined over the source and target language sentences and the generalized iterative scaling algorithm was applied to optimize the features on an N-best list of translation candidates generated from a baseline MT system.

The minimum error training approach proposed in [39] directly optimized the translation model parameters for the BLEU evaluation criterion. The authors in [39] observed that the objective function is piecewise constant and so it can be characterized exhaustively along any line in parameter space. By calling this global one-dimensional line minimization procedure as a subroutine of the multidimensional minimization routine, they obtained significant improvements in translation performance under different evaluation criteria.

Instead of considering only the top hypothesis when minimizing the loss, an approach to minimize expected loss was presented in [41]. A risk based annealing procedure with entropy regularization was used as the objective function to opti-

mize the translation parameters. Since the BLEU criterion in the objective function involves a non-linear combination over the sentences in the training corpus, a first order approximation was used to approximate BLEU in the training objective. Significant improvements in BLEU were obtained over standard minimum error training and maximum likelihood.

More recently, a systematic comparison of the various training criteria used in training statistical machine translation systems was presented [42]. The authors in [42] used simplex search for optimizing the MMI and Expected BLEU criteria. The expected BLEU training criteria was shown to give significant performance improvements as compared to MMI and minimum error training.

A discriminative re-ranking algorithm using a variant of perception training was also investigated [43]. The authors introduced a new perceptron-like splitting algorithm to achieve separability of the good translation candidates from the bad ones. Ordinal regression with uneven margins was used to re-rank the hypothesis in the N-best lists. In the experiments, translation performance similar to minimum error training was obtained.

In the following sections, we will discuss Minimum Error Training (MET) which is a line search based multidimensional search procedure that optimizes the direct loss objective function presented in Equation 3.4.

3.3 Minimum Error Training

Minimum Error Training (MET) is the current state of the art in discriminatively training the parameters of the MT system [39]. Following the development in [39], we outline the MET procedure in detail.

Let BLEU be the evaluation criterion used to measure translation performance. The goal is to maximize BLEU over a representative corpus of S sentence $\mathbf{f}_s$ with given reference translations $\mathbf{e}_s$ and a set of K_s different translation hypotheses $\mathbf{C}_s =$

$\{\mathbf{e}_{s,1}, \ldots, \mathbf{e}_{s,K_s}\}$. Restating the objective function in Equation 3.4:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{s=1}^{S} \sum_{k=1}^{K} BLEU(\mathbf{e}_{sk}, \mathbf{e}_s^+) \delta(\hat{\mathbf{e}}_s(f_s; \theta), \mathbf{e}_{sk}) \tag{3.7}$$

where,

$$\hat{\mathbf{e}}_s(f_s; \theta) = \underset{\mathbf{e} \in \mathbf{C}_s}{\operatorname{argmax}}\ p_\theta(\mathbf{e}_s | \mathbf{f}_s) \tag{3.8}$$

The optimization function defined in Equation 3.7 is not straightforward to optimize:

- It includes an argmax operation. Therefore, it is not possible to compute a gradient and gradient descent methods cannot be used to perform optimization.
- The objective function has many different local optima. The optimization algorithm must yield robust parameter estimates despite this.

There exist gradient free optimization methods such as Nelder-Mead simplex and Powell's method [44] to solve this Q dimensional optimization problem. A random Q dimensional point in parameter space is initialized and then a search is carried out to find a better scoring point in the parameter space by making a one-dimensional line minimization along the directions given by optimizing one parameter while keeping all other parameters fixed. To avoid finding a poor local optimum, we start from different initial parameter values. A major problem with the standard approach is the fact that grid-based line optimization is hard to adjust such that both good performance and efficient search are guaranteed. If a fine-grained grid is used then the algorithm is slow. If a large grid is used then the optimal solution might be missed.

The authors in [39] observed that the objective function is piecewise constant and can be exploited to speed up the optimization. Each candidate $\hat{\mathbf{e}}_s(\mathbf{f}_s; \theta)$ has the

functional form:

$$\hat{\mathbf{e}}_s(\mathbf{f}_s;\theta) \propto \underset{\mathbf{e}\in\mathbf{C}_s}{\operatorname{argmax}} \sum_{q=1}^{Q} \theta_q \log \Phi_q(\mathbf{e}_s,\mathbf{f}_s) \tag{3.9}$$

$$\propto \underset{\mathbf{e}\in\mathbf{C}_s}{\operatorname{argmax}} \underbrace{\sum_{q\neq p,q=1}^{Q} \theta_q \log \Phi_q(\mathbf{e}_s,\mathbf{f}_s)}_{t(\mathbf{e}_s,\mathbf{f}_s)} + \underbrace{\theta_p \log \Phi_p(\mathbf{e}_s,\mathbf{f}_s)}_{m(\mathbf{e}_s,\mathbf{f}_s)} \tag{3.10}$$

$$t(\mathbf{e}_s,\mathbf{f}_s) \propto \underset{\mathbf{e}\in\mathbf{C}_s}{\operatorname{argmax}} \Big\{ t(\mathbf{e}_s,\mathbf{f}_s) + \theta_p \; m(\mathbf{e}_s,\mathbf{f}_s) \Big\} \tag{3.11}$$

Thus, each candidate in the N-best list of translation hypotheses represents a line in $\mathbb{R}^2$ with respect to the parameter θ_p. As a consequence, the error only changes when we move θ_d from a one line to another, which in turn implies that we need to evaluate errors only at the intersection of the candidate lines in the N-best list, to get a complete representation of the error surface. When searching for the intersection points we start from a line with minimum slope and calculate intersection points at all lines with a steeper slope. These intersection points form the "critical set" of points at which the objective function is evaluated for a particular sentence. In addition a record of the incremental change in error counts involved at the corresponding intersection point is also kept. Specifically we define a pair $(\theta_{p,i}, \Delta_i Error)$ that associates the change in error with the critical point $\theta_{p,i}$ when crossing the ith intersection. For the BLEU score the error counts correspond to the sufficient statistics: the number of correct and suggested n-grams, as well as the length of the closest reference. Error deltas are then a set of deltas for each relevant statistic.

When the $(\theta_{p,i}, \Delta_i Error)$ pairs are merged over all source sentences and sorted according to the intersection, the error deltas are simply summed over, when crossing intersection boundaries to track the current value of each statistic. If there are duplicate intersection points in the merged list, the error is only considered once to filter out duplicate intersection points. Finally, a new θ_d^* is selected as the midpoint of the interval corresponding to the lowest error and the search for the next parameter dimension is continued. Termination conditions can be based on the number of iterations or successive reduction of error across iterations.

Thus in MET, the Q dimensional optimization problem is solved by calling a global line based optimization as a subroutine of a multidimensional optimization procedure. In the next section, a update procedure to estimate the parameters

$\theta_q,\ q = 1, \ldots, Q$ is introduced as an alternative to MET.

3.4 Growth Transformations

The line search algorithm discussed in 3.3 was introduced to solve the optimization problem when the objective function is not smooth and differentiable. However, the objective functions presented in Equation 3.2 and Equation 3.5 are differentiable with respect to the parameters and so we can use any gradient based methods to optimize them. In this thesis, we introduce growth transformations which is a gradient based iterative update procedure to locally optimize differentiable objective functions.

First, a the general theory of growth transformation based updates is presented for the case of polynomial functions and then later extended to general functions.

3.4.1 Growth Transforms for Rational Functions

The well known Baum-Eagon inequality [45] provides an effective iterative schema for finding the local maximum for homogenous polynomials with positive coefficients defined over a domain of probability values. However, for a growing class of problems (employing conditional maximum likelihood or maximum mutual information parameter estimation) we are interested in maximizing a general (rational) function. In [46], the authors extended the Baum-Eagon inequality to the case of rational functions over linear domains. The main theorem can be restated as follows:

Theorem 3.4.1. *Let* $R(\Theta) = \frac{N(\Theta)}{D(\Theta)}$, *where* $N(\Theta)$ *and* $D(\Theta)$ *are polynomials in variables* $\{\Theta_q\},\ \ q = 1, \ldots, Q$ *defined over a domain* $D : \{\theta_q \geq 0, \sum_{q=1}^{Q} \theta_q = 1\}$ *and* $D(\Theta) > 0,\ \forall \Theta \in D$. *For some point* $\theta \in D$, *let us define the polynomial*

$$P_\theta(\Theta) = N(\Theta) - R(\theta)D(\Theta) + C$$

where, C *is some constant that ensures that the polynomial* $P_\theta(\Theta)$ *has positive coefficients. Furthermore, let* $\hat{\theta} = T(\theta)$ *be a point in* D *whose* q*th coordinate is*

$$\hat{\theta_q} = T(\theta)_q = \frac{\theta_q \left(\frac{\partial P_\theta(\Theta)}{\partial \theta_q} + C \right)}{\sum_{q=1}^{Q} \theta_q \left(\frac{\partial P_\theta(\Theta)}{\partial \theta_q} + C \right)} \tag{3.12}$$

For a sufficiently large $C > 0$, T defined by Equation 3.12 is a growth transform in D, which in turn implies $R(\hat{\theta}) > R(\theta)$, unless $\hat{\theta} = \theta$

Theorem 3.4.1 outlines a general procedure for maximizing a rational function (ratio of homogenous polynomials). The growth transform defined in Equation 3.12 specifies a general iterative procedure to find parameter updates that guarantee an increase in the objective function at each iteration, for a sufficiently large C.

3.4.2 Growth Transform for General Functions

However, in many applications we might be interested in maximizing the objective function over a more general (not necessarily rational) function. This is the case in our current setup where the joint distribution defined in Equation 3.6 is not a polynomial in the parameters θ that we wish to estimate. So, for example, if we want to apply growth transformations towards maximizing the MMI objective defined in Equation 3.2 Theorem 3.4.1 does not apply.

However, the particular form of the joint distribution, makes it possible to express it as a polynomial in θ. First, note that the joint distribution can be written as:

$$p_\theta(\mathbf{e}_s, \mathbf{f}_s) = \prod_{q=1}^{Q} \exp\left(\theta_q \ \log \Phi_q(\mathbf{e}_s, \mathbf{f}_s)\right) \tag{3.13}$$

Then a Taylor series expansion of the joint distribution is:

$$p_\theta^{(n)}(\mathbf{e}_s, \mathbf{f}_s) \approx \prod_{q=1}^{Q} \sum_{k=0}^{n} \frac{\left(\theta_q \ \log \Phi_q(\mathbf{e}_s, \mathbf{f}_s)\right)^k}{k!} \tag{3.14}$$

Here, $p_\theta^{(n)}(\mathbf{e}_s, \mathbf{f}_s)$ is the nth order polynomial approximation of $p_\theta(\mathbf{e}_s, \mathbf{f}_s)$.

Similarly, let $F^{(n)}(\theta)$ be the nth order polynomial approximation of an objective function $F(\theta)$ and $T^{(n)}(\theta)$ be the nth order polynomial approximation of the growth transformation defined as

$$T^{(n)}(\theta)_q = \frac{\theta_q\left(\frac{\partial F^{(n)}(\theta)}{\partial \theta_q} + C\right)}{\sum_{q=1}^{Q} \theta_q\left(\frac{\partial F^{(n)}(\theta)}{\partial \theta_q} + C\right)} \tag{3.15}$$

Then, for a sufficiently large C and using Theorem 3.4.1 we have

$$F^{(n)}(T^{(n)}(\theta)) \geq F^{(n)}(\theta)$$

.

Now, for a sufficiently large n if

$$\lim_{n\to\infty} T^{(n)}(\theta) \to T(\theta) \tag{3.16}$$

$$\lim_{n\to\infty} F^{(n)}(\theta) \to F(\theta) \tag{3.17}$$

we want to prove

$$\lim_{n\to\infty} F^{(n)}(T^{(n)}(\theta)) \geq \lim_{n\to\infty} F^{(n)}(\theta) \leftrightarrow F(T(\theta)) \geq F(\theta)$$

This would allow us to apply growth transforms to objective functions that can be expressed locally as a power series. In [47], the authors showed that the growth transforms could indeed be extended from the case of rational functions to the more general case of functions defined on general manifolds provided they can be expressed as a power series. The following theorem encapsulates the basic premise discussed above.

Theorem 3.4.2. *Let the function $F(\theta)$ be differentiable at $\theta \in D$. Furthermore, let F be analytic at θ (i.e. F can be represented locally by a power series) and let C be a non-negative constant. Also, let us define the projection*

$$T(\theta)_q = \frac{\theta_q\left(\frac{\partial F(\theta)}{\partial \theta_q} + C\right)}{\sum_{q=1}^{Q} \theta_q\left(\frac{\partial F(\theta)}{\partial \theta_q} + C\right)} \tag{3.18}$$

There exists a constant $C \geq 0$ such that the following holds: $T(\theta)_q \in D$ and $F(T(\theta)) \geq F(\theta)$

Theorem 3.4.2 states that under very general assumptions on a manifold D, T is a growth transform for a sufficiently large C. We will make use of these results to define growth transformation based parameter updates for the various training objective functions we have discussed so far.

3.4.3 Convergence Factor

There is still the issue of defining the convergence factor C. By definition, C should be sufficiently large to guarantee that the updates derived are actually a growth transformation. Furthermore, the constant C controls the speed of convergence of the iterative algorithm - too high a value leads to slow convergence, and a very low value might result in instability of the optimization process. Hence, it is desirable to use a smaller C, provided that (3.18) is still a growth transform. In [46], the author suggested a practical value of C that works well in practice but there is no longer a theoretical guarantee of convergence. A modified version is provided here:

$$C = N_c * \left[\max \left\{ \max_q \{ -\nabla \mathcal{F}(\Theta)|_{\theta=\theta_q} \}, 0 \right\} + \epsilon \right] \tag{3.19}$$

The intuition behind this formulation is to select a value of C that is equal to the most negative derivative component. This ensures that the growth transform based updates are always non-negative. Furthermore, the N_c adds an additional multiplicative constant to control convergence rate of the iterative procedure. N_c is fixed by experimentation for the particular task.

3.4.4 Iterative Training Algorithm

Theorem 3.4.2 suggests a simple iterative procedure that finds the local maximum of $F(\Theta)$.

Algorithm 1 Growth Transform Update Procedure

1. Initialize the parameter vector $\theta = \{\theta_1, \cdots \theta_Q\}$, such that $\sum_{q=1}^{Q} \theta_q = 1$, and $i = 0$.
2. For each parameter $\theta_q^{(i)}$, calculate the gradient $\nabla \mathcal{F}(\Theta^{(i)})|_{\theta=\theta_q^{(i)}}$
3. For each parameter $\theta_q^{(i)}$, calculate the parameter update

$$\theta_q{}^{(i+1)} = \frac{\theta_q^{(i)} \left(\nabla \mathcal{F}(\Theta^{(i)})|_{\theta=\theta_q^{(i)}} + C \right)}{\sum_{q=1}^{Q} \theta_q^{(i)} \left(\nabla \mathcal{F}(\Theta^{(i)})|_{\theta=\theta_q^{(i)}} + C \right)} \tag{3.20}$$

4. If $\mathcal{F}(\Theta^{(i+1)}) \leq \mathcal{F}(\Theta^{(i)})$ or if $i == MAXITER$, then terminate. Else, $i \leftarrow i + 1$, goto Step 2.

Thus once we have calculated $\nabla\mathcal{F}(\Theta)$, the first order derivatives of our objective function $\mathcal{F}(\Theta)$ we can easily calculate the parameter updates using growth transformations. In the following chapter we will show how the growth transform based update procedure can be applied to estimate the parameters of a machine translation system.

Chapter 4

Growth Transformations for Machine Translation

In this chapter, a detailed derivation of growth transformation based parameter updates is provided for the two objective functions we will consider - MMI and Expected Loss. The general procedure is to calculate the derivatives of the objective function and plug these into Equation 3.20 to obtain the parameter updates.

4.1 Growth Transformations for MMI Training

4.1.1 MMI for single reference system

The objective function is assumed to be of the form given in Equation 3.2 where the objective is to maximize the posterior of the true class (reference translation $\mathbf{e}_s^+$) given the input sentence $\mathbf{f}_s$. Here, we assume that for each sentence $\mathbf{f}_s$ there is only one available reference translation $\mathbf{e}_s^+$.

We begin by calculating the derivative of Equation 3.2

$$\nabla_\theta \mathcal{F}_{\mathcal{MMI}}(\theta)) = \sum_{s=1}^{S} \nabla_\theta \log p_\theta(\mathbf{e}_s^+, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{\mathbf{e}_s \in \mathbf{E}} \frac{1}{\sum_{\mathbf{e}'_s \in \mathbf{E}} p_\theta(\mathbf{e}'_s, \mathbf{f}_s)} \nabla_\theta p_\theta(\mathbf{e}_s, \mathbf{f}_s) \quad (4.1)$$

Observe that

$$\frac{1}{\sum_{\mathbf{e}'_s \in \mathbf{E}} p_\theta(\mathbf{e}'_s, \mathbf{f}_s) \nabla_\theta p_\theta(\mathbf{e}_s, \mathbf{f}_s)} = p_\theta(\mathbf{e}_s|\mathbf{f}_s) \nabla_\theta \log p_\theta(\mathbf{e}_s, \mathbf{f}_s) \quad (4.2)$$

Plugging this into Equation 4.1 , we have:

$$\nabla_\theta \mathcal{F}_{\mathcal{MMI}}(\theta) = \sum_{s=1}^{S} \nabla_\theta \log p_\theta(\mathbf{e}_s^+, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{\mathbf{e}_s \in \mathbf{E}} p_\theta(\mathbf{e}_s|\mathbf{f}_s) \nabla_\theta \log p_\theta(\mathbf{e}_s, \mathbf{f}_s) \tag{4.3}$$

The derivative of the joint defined in Equation 3.6 is:

$$\nabla_\theta \log p_\theta(\mathbf{e_s}, \mathbf{f_s})\Big|_{\theta=\theta_q} = \log \Phi_q(\mathbf{e}_s, \mathbf{f}_s) \tag{4.4}$$

Substituting Equation 4.4 in Equation 4.3 we obtain

$$\nabla_\theta \mathcal{F}_{\mathcal{MMI}}(\theta)\Big|_{\theta=\theta_q} = \sum_{s=1}^{S} \log \Phi_q(\mathbf{e}_s^+, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{\mathbf{e}_s \in \mathbf{E}} p_\theta(\mathbf{e}_s|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_s, \mathbf{f}_s) \tag{4.5}$$

Finally, in all our experiments we approximate the translation hypothesis space by an N-best list. Let N_s be the N-best list of translations associated with the input sentence $\mathbf{f}_s$. This simplifies the objective function to:

$$F_{MMI}(\theta) = \sum_{s=1}^{S} \log \frac{p_\theta(\mathbf{e}_s^+, \mathbf{f}_s)}{\sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s)} \tag{4.6}$$

and also the derivative:

$$\nabla_\theta \mathcal{F}_{\mathcal{MMI}}(\theta)\Big|_{\theta=\theta_q} = \sum_{s=1}^{S} \log \Phi_q(\mathbf{e}_s^+, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) \tag{4.7}$$

Substituting the derivative in Equation 3.20, results in the following growth transform

$$\hat{\theta}_q = \frac{\theta_q \Big(\sum_{s=1}^{S} \log \Phi_q(\mathbf{e}_s^+, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) + C \Big)}{\sum_{q=1}^{Q} \theta_q \Big(\sum_{s=1}^{S} \log \Phi_q(\mathbf{e}_s^+, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) + C \Big)} \tag{4.8}$$

In MT evaluation, translation performance is usually evaluated against multiple references. In order to account for this we need to modify the MMI objective function appropriately. In the following section, two choices for MMI objective functions with multiple references are evaluated - MMI sum of references and MMI product of references.

4.1.2 MMI sum of references

Let R_s be the number of reference translations for sentence $\mathbf{e_s}$. Then, $\mathbf{e^+_{s,r}}$ denotes the rth reference for sentence $\mathbf{e_s}$. The modified objective approximated over N-best lists is

$$\mathcal{F}_{\mathcal{MMI}sum}(\theta) = \sum_{s=1}^{S} \log \left(\frac{1}{R_s} \sum_{r=1}^{R_s} \frac{p_\theta(\mathbf{e}^+_{s,r}, \mathbf{f}_s)}{\sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s)} \right) \tag{4.9}$$

This objective function attempts to maximize the posterior over any (but not necessarily all) of the references. It can be viewed as a 'logical OR' operation over the reference posteriors - the parameters are update towards the reference with maximum posterior probability.

The derivative of the objective function is:

$$\nabla_\theta \mathcal{F}_{\mathcal{MMI}sum}(\theta)\Big|_{\theta=\theta_q} = \sum_{s=1}^{S} \sum_{r=1}^{R_s} \frac{p_\theta(\mathbf{e}^+_{s,r}, \mathbf{f}_s)}{\sum_{r'=1}^{R_s} p_\theta(\mathbf{e}^+_{s,r'}, \mathbf{f}_s)} \log \Phi_q(\mathbf{e}^+_{s,r}, \mathbf{f}_s) - \sum_{s=1}^{S} \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) \tag{4.10}$$

Substituting this derivative into (3.20) gives us the desired updates.

$$\hat{\theta_q} = \frac{\theta_q \left(\left\{ \sum_{s,r} \frac{p_\theta(\mathbf{e}^+_{s,r}, \mathbf{f}_s)}{\sum_{r'} p_\theta(\mathbf{e}^+_{s,r'}, \mathbf{f}_s)} \log \Phi_q(\mathbf{e}^+_{s,r}, \mathbf{f}_s) - \sum_{s,k} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) \right\} + C \right)}{\sum_{q=1}^{Q} \theta_q \left(\left\{ \sum_{s,r} \frac{p_\theta(\mathbf{e}^+_{s,r}, \mathbf{f}_s)}{\sum_{r'} p_\theta(\mathbf{e}^+_{s,r'}, \mathbf{f}_s)} \log \Phi_q(\mathbf{e}^+_{s,r}, \mathbf{f}_s) - \sum_{s,k} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) + C \right\} \right)} \tag{4.11}$$

4.1.3 MMI product of references

Another alternative to the MMI objective function is maximizing the posterior with respect to all the references simultaneously (equivalent to 'logical AND' operation) which gives us the following objective function:

$$\mathcal{F}_{\mathcal{MMI}product}(\theta) = \sum_{s=1}^{S} \frac{1}{R_s} \sum_{r=1}^{R_s} \log \frac{p_\theta(\mathbf{e}^+_{sr}, \mathbf{f}_s)}{\sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s)} \tag{4.12}$$

The derivative of the objective function $\mathcal{F}_{\mathcal{MMI}_{product}}(\theta)$ is given by:

$$\nabla_\theta(\mathcal{F}_{\mathcal{MMI}_{product}}(\theta))\Big|_{\theta=\theta_q} = \sum_{s=1}^{S}\left\{\frac{1}{R_s}\sum_{r=1}^{R_s}\log\Phi_q(\mathbf{e}_{sr}^{+},\mathbf{f}_s) - \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\log\Phi_q(\mathbf{e}_{sk},\mathbf{f}_s)\right\} \tag{4.13}$$

Again, we can use the update rule defined in (3.20) to calculate the parameter updates.

$$\hat{\theta_q} = \frac{\theta_q\left(\sum_{s=1}^{S}\left\{\frac{1}{R_s}\sum_{r=1}^{R_s}\log\Phi_q(\mathbf{e}_{sr}^{+},\mathbf{f}_s) - \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\log\Phi_q(\mathbf{e}_{sk},\mathbf{f}_s)\right\} + C\right)}{\sum_{q=1}^{Q}\theta_q\left(\sum_{s=1}^{S}\left\{\frac{1}{R_s}\sum_{r=1}^{R_s}\log\Phi_q(\mathbf{e}_{sr}^{+},\mathbf{f}_s) - \sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\log\Phi_q(\mathbf{e}_{sk},\mathbf{f}_s)\right\} + C\right)} \tag{4.14}$$

The MMI objective functions introduced above for the case of multiple references are just two of the possible choices we consider here. We will evaluate our choice of objective functions by measuring translation performance on different tasks.

4.1.4 Labeling the correct class for MMI Training

Since, MMI training attempts to separate the class conditional probability of the true class from all the other competing classes we need to identify the "correct" candidate translation from the hypotheses space of each sentence we want to translate. An obvious choice for the correct class is to choose a candidate from the hypothesis space that is identical to the reference translation provided for each sentence in the training corpus. However, there is the problem that none of the candidate translations in the hypothesis space might match the provided reference translation. In such a case, the candidate from the hypothesis space that has the fewest errors under the particular evaluation criterion (also known as oracle reference) being used to measure performance, can be labeled as the correct class.

MMI using reference translations as the correct class is inherently problematic because a proposed translation that differs from a reference translation need not

be incorrect. It may differ in word choice or style, and yet be a fully acceptable translation. Pushing our system to avoid such alternate translations is undesirable. It is also possible that the reference translation is *unreachable* by the decoder. Since, in our training framework we need to extract a set of relevant features we can attempt to force align the reference under the current translation models. However, it may do so by abusing the hidden structure (sentence segmentation and phrase alignment). We can therefore never be entirely sure whether or not a proposed output is safe to update towards.

Another alternative is to select a candidate from the space of translations that is closest to the true reference, as the correct class. In machine translation, we use BLEU to measure translation performance and so we can measure the "closeness" of the hypothesis to the reference translation under BLEU: the candidate hypothesis with the highest sentence level BLEU score is chosen as the correct class. We call this the oracle-BLEU hypothesis. The parameters are now updated so as to maximize the likelihood of the oracle-BLEU hypothesis under the model.

Note that BLEU computation as discussed in Section 2.2 involves a non-linear combination over the sentences in the corpus [41]. However, for the training criterion we use BLEU measured at the sentence level and note that this is just an approximation to the true document level BLEU.

4.1.5 Growth Transformations for Expected BLEU Training

Following the discussion in Section 3.1.4, the goal is to minimize the expected risk (or maximize the expected gain) for the evaluation criterion under consideration. Since we are using the BLEU criterion for measuring translation performance, and since an improved translation performance corresponds to an increase in BLEU, minimizing expected loss is equivalent to maximizing expected BLEU. The objective function we are interested in is

$$\mathcal{F}_{\mathcal{MBR}}(\theta) = \sum_{s=1}^{S} \sum_{\mathbf{e}_s \in \mathbf{E}} BLEU(\mathbf{e}_s^+, \mathbf{e}_s)\ p_\theta(\mathbf{e}_s|\mathbf{f}_s) \tag{4.15}$$

Restricting the hypothesis space to N-best lists we obtain:

$$\mathcal{F}_{\mathcal{MBR}}(\theta) = \sum_{s=1}^{S} \sum_{k=1}^{N_s} BLEU(\mathbf{e}_{sk}^+, \mathbf{e}_s)\ p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \tag{4.16}$$

The objective function of interest is continuous and differentiable, and so we can obtain the derivatives

$$\nabla_\theta \mathcal{F}_{\mathcal{MBR}}(\theta)\Big|_{\theta=\theta_q} = \sum_{s=1}^{S}\sum_{k=1}^{N_s} BLEU(\mathbf{e}_{sk}^{+}, \mathbf{e}_s)\ \nabla_\theta p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\Big|_{\theta=\theta_q} \tag{4.17}$$

The derivative for the posterior distribution can be written as

$$\nabla_\theta p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) = \frac{\Big(\sum_{l=1}^{N_s} p_\theta(\mathbf{e}_{sl}, \mathbf{f}_s)\Big)\nabla_\theta p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s) - p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s)\Big(\sum_{l=1}^{N_s} \nabla_\theta p_\theta(\mathbf{e}_{sl}, \mathbf{f}_s)\Big)}{\Big(\sum_{l=1}^{N_s} p_\theta(\mathbf{e}_{sl}, \mathbf{f}_s)\Big)^2} \tag{4.18}$$

$$= p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\ \Big\{\nabla_\theta \log p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s) - \sum_{l=1}^{N_s} p_\theta(\mathbf{e}_{sl}|\mathbf{f}_s)\nabla_\theta \log p_\theta(\mathbf{e}_{sl}, \mathbf{f}_s)\Big\} \tag{4.19}$$

Substituting Equation 3.6 in the derivative we obtain:

$$\nabla_\theta \mathcal{F}_{\mathcal{MBR}}(\theta)\Big|_{\theta=\theta_q} = \sum_{s,k} BLEU(\mathbf{e}_{sk}^{+}, \mathbf{e}_s) \tag{4.20}$$
$$p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\ \Big\{\log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) - \sum_{l} p_\theta(\mathbf{e}_{sl}|\mathbf{f}_s)\log \Phi_q(\mathbf{e}_{sl}, \mathbf{f}_s)\Big\}$$

Finally, substituting this derivative in to the update rule defined in (3.20) gives us the desired parameter updates.

$$\hat{\theta_q} = \frac{\theta_q\Big(\sum_{s,k} BLEU(\mathbf{e}_{sk}^{+}, \mathbf{e}_s)p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\ \Delta\Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) + C\Big)}{\sum_{q=1}^{Q}\theta_q\Big(\sum_{s,k} BLEU(\mathbf{e}_{sk}^{+}, \mathbf{e}_s)p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\ \Delta\Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) + C\Big)} \tag{4.21}$$

where,

$$\Delta\Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) = \log \Phi_q(\mathbf{e}_{sk}, \mathbf{f}_s) - \sum_{l} p_\theta(\mathbf{e}_{sl}|\mathbf{f}_s)\log \Phi_q(\mathbf{e}_{sl}, \mathbf{f}_s) \tag{4.22}$$

4.2 Regularization

In order to avoid overfitting the training data, regularization terms are introduced into the objective function. The convergence factor C already controls overfitting to some extent by controlling the speed of convergence. In addition to this, we

would also like that the distribution defined by the parameters not be too peaked. We can express directly the optimization objective in terms of a preference for a higher entropy over the parameters [41]. A higher entropy corresponds to a flatter distribution over the parameters. The form of the modified objective function is:

$$\mathcal{G}(\theta) = \mathcal{F}(\theta) + T\ H(p_\theta) \tag{4.23}$$

A scaling factor $T > 0$ is introduced to weight the contribution of the regularization term. T is manually set by experimentation on a development corpus. Furthermore, since the growth transform based update involves calculation of the derivative we need to modify the updates accordingly. The derivative of the modified objective function is then:

$$\nabla_\theta \mathcal{G}(\theta) = \nabla_\theta \mathcal{F}(\theta) + T\ \nabla_\theta\ H(p_\theta) \tag{4.24}$$

where, the entropy over the N-best list is expressed as

$$H(p_\theta) = -\sum_{s=1}^{S}\sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\log p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \tag{4.25}$$

and the derivative is expressed as

$$\nabla_\theta H(p_\theta) = -\sum_{s=1}^{S}\sum_{k=1}^{N_s}\Big(1+\log p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\Big)\nabla_\theta p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \tag{4.26}$$

$$= -\sum_{s=1}^{S}\sum_{k=1}^{N_s}\Big(1+\log p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\Big)\ p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \Big\{\nabla_\theta \log p_\theta(\mathbf{e}_{sk},\mathbf{f}_s) - \sum_{l=1}^{N_s} p_\theta(\mathbf{e}_{sl}|\mathbf{f}_s)\nabla_\theta \log p_\theta(\mathbf{e}_{sl},\mathbf{f}_s)\Big\} \tag{4.27}$$

And so, we have

$$\nabla_\theta H(p_\theta)\Big|_{\theta=\theta_q} = -\sum_{s=1}^{S}\sum_{k=1}^{N_s}\Big(1+\log p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s)\Big)\ p_\theta(\mathbf{e}_{sk}|\mathbf{f}_s) \Big\{\log \Phi_q(\mathbf{e}_{sk},\mathbf{f}_s) - \sum_{l=1}^{N_s} p_\theta(\mathbf{e}_{sl}|\mathbf{f}_s)\log \Phi_q(\mathbf{e}_{sl},\mathbf{f}_s)\Big\} \tag{4.28}$$

Finally, the update equations for the growth transformations are of the form:

$$\hat{\theta_q} = \frac{\theta_q\Big(\nabla\mathcal{G}(\theta)|_{\theta=\theta_q} + C\Big)}{\sum_{q=1}^{Q}\theta_q\Big(\nabla\mathcal{G}(\theta)|_{\theta=\theta_q} + C\Big)} \tag{4.29}$$

4.3 Posterior scaling

A scaling factor α is introduced as an additional free parameter to scale the posterior probabilities. A large α leads to very peaked distributions, and most of the probability mass is concentrated towards the top scoring hypothesis. In this case, the objective function maximization will focus only on the top scoring hypotheses in the N-best list. A small α leads to a flatter posterior distribution and the probability mass is distributed over a larger set of hypotheses in the N-best list. A small α leads to a flatter decision surface where a larger set of N-best list candidates affect the parameter optimization.

Since, we do not have access to the true posterior probability, we approximate the posterior distribution under the current model parameters. The normalized posterior probability for a N-best candidate $\mathbf{e}_{sk}$ i.e the kth candidate for sentence s, can we written as:

$$p_{\theta,\alpha}(\mathbf{e}_{sk}|\mathbf{f}_s) = \frac{p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s)^\alpha}{\sum_{k=1}^{N_s} p_\theta(\mathbf{e}_{sk}, \mathbf{f}_s)^\alpha} \tag{4.30}$$

The optimal α is chosen by experimentation.

4.4 Parameter Estimation for log-linear models

The growth transformation based update is a general procedure for iteratively estimating the parameters of a model by locally optimizing a particular objective function. The particular form of the joint distribution expressed in Equation 3.6 allows us to define the following conditional probability distribution

$$p_\theta(\mathbf{e}_s|\mathbf{f}_s) = \frac{\exp\ \bar{\theta} \cdot \mathbf{h}(\mathbf{e}_s, \mathbf{f}_s)}{\sum_{\mathbf{e}_s} \exp\ \bar{\theta} \cdot \mathbf{h}(\mathbf{e}_s, \mathbf{f}_s)} \tag{4.31}$$

where,
$\bar{\theta} = [\theta_1, \theta_2, \ldots, \theta_Q]$ is the parameter vector,
$\mathbf{h}(\mathbf{e}_s, \mathbf{f}_s) = [\log \Phi_1(\mathbf{e}_s, \mathbf{f}_s), \log \Phi_2(\mathbf{e}_s, \mathbf{f}_s), \ldots, \log \Phi_Q(\mathbf{e}_s, \mathbf{f}_s)]^T$ is the feature vector
$\bar{\theta} \cdot \mathbf{h}(\mathbf{e}_s, \mathbf{f}_s) = \sum_{q=1}^{Q} \theta_q \log \Phi_q(\mathbf{e}_s, \mathbf{f}_s)$ is the dot product of the feature vector and parameter vector.

This is the form of the log-linear model used in most feature-based SMT systems [40, 48, 43]. Consequently, growth transform based updates can be employed

as an alternative estimation technique for maximizing the log-linear model based conditional likelihood objective.

4.5 Summary

In this chapter, a detailed derivation of the growth transformation based updates was provided for the two discriminative objective functions for training the parameters of a SMT system - MMI and Expected BLEU. The MMI objective function maximizes the posterior of the correct class and attempts to separate it from the incorrect class. The expected BLEU training criterion on the other focuses on maximizing the expected BLEU over a training set.

For MMI, growth transformation based updates were derived for the case of a single reference system. Since, in SMT evaluation it is typical to use multiple references during evaluation two different MMI objectives explicitly incorporating the multiple references into the objective function were introduced - MMI with sum over references and MMI with product over references. The MMI objective with sum over references attempts to increase the likelihood of at least one, though not all of the references simultaneously while driving down the likelihood of the competing classes. The MMI objective with product over references on the other hand attempts to drive up the likelihood of all the references simultaneously. Finally, the problem of labeling the correct class for MMI training was addressed. In the case where the true reference is not a part of the hypothesis space, choosing the highest BLEU score hypothesis was suggested as an alternative.

An objective function for maximizing the expected BLEU over a N-best list was introduced. During training, the BLEU score is calculated over sentence pairs instead of over the entire corpus. This is an approximation to the true BLEU score.

Finally, additional parameters were introduced to control the optimization process. A posterior scaling factor α was introduced which controls the peakedness of the decision surface by distributing the probability mass over different regions in the hypothesis space. Also, a entropy based regularization factor was introduced in to the training objective in order to avoid overfitting. The parameter T controls the

contribution of the regularization term to the entire objective function optimization.

Chapter 5

Experiments with Discriminative Training for Text Translation

In this chapter, experimental results for the different discriminative training criteria are presented. Translation performance of expected BEU training and MMI training are compared against the state-of-the-art line-based MET search procedure. The experiments are divided into two sections: static re-anking and dynamic re-ranking. In the static re-ranking experiments the N-best lists are kept fixed for the various experiments. The learned parameters merely re-rank the translation hypotheses. This is done for convenience and also to do a fair comparison of the various optimization strategies. Dynamic re-ranking involves an iterative procedure where the decoder utilizes the optimized parameters to re-generate N-best lists in an iterative fashion.

5.1 Experimental Setup

Translation performance is evaluated for three different language pairs - Arabic-English (AR-EN), Chinese-English (ZH-EN) and Spanish-English (ES-EN). Table 5.1 gives the corpus statistics used for training the various translation models. The training data is lower-cased and sentence aligned.

5.1.1 Translation Model Training

As mentioned earlier in Section 2.7.2, the TTM relies on an underlying inventory of phrase-pairs. Before building the phrase pair inventory, word alignments need to

Language Pair	Statistics	
Arabic-English(Ar-En)	# of Sentence Pairs	6.3M
	# of Arabic words	127M
	# of English words	142M
Chinese-English(Zh-En)	# of Sentence Pairs	10M
	# of Chinese words	204M
	# of English words	220M
Spanish-English(Es-En)	# of Sentence Pairs	1.4M
	# of Spanish words	36M
	# of English words	37M

Table 5.1: Statistics for the parallel training data used to train the translation model components

be obtained from the sentence aligned parallel corpora. A word alignment is simply a function mapping the words in one sentence to the words in another sentence , for the given pair of sentences in the parallel corpus. Since the word alignment is just a function mapping, it can be obtained in two directions: words in the source sentence map to a particular word in the target sentence or a word in the target sentence maps onto a word in the source sentence. Word to phrase alignment models are trained using MTTK [49] tool. A set of word alignment models are trained in two directions: source language to target language ($E \rightarrow F$) and target language to source language ($F \rightarrow E$). In order to improve the word alignment coverage and allow for many-to-many alignments, the final word alignment set is the union of the two directions i.e $(F \rightarrow E) \bigcup (E \rightarrow F)$.

Next, a set of heuristics as discussed in Section 2.7.2 are used to extract the phrase pairs. The phrase pair inventory contains a list of source phrases and their corresponding translations along with a probability associated with the translation. The translation probability is simply a maximum likelihood estimate of the frequency of occurrences of the phrase pairs in the training corpus. The phrase pair inventory thus obtained is used to build the TTM components - the target phrase segmentation model, the target phrase insertion model, the target phrase reordering model, the source segmentation model and the source-to-target phrase transduction model. The component models are modeled as WFSTs.

In addition, a source word language model is also built from the source language

side of the parallel training corpus. In our experiments, the English language model component of the TTM is a 4-gram language model trained on the english side of the parallel text corpus using the SRILM toolkit [50].

5.1.2 Discriminative Training Setup

All the discriminative training experiments are carried out by optimizing the parameters on a development (DEV) set and finally evaluating on a blind test (TST) set. Table 5.1.2 gives the statistics of the development and test data sets for the three language pairs on which translation performance is reported in terms of the BLEU score.

Language Pair	Statistics	DEV	TST
Arabic-English(Ar-En)	Arabic Sentences	2,075	2,040
	Arabic words	56K	55K
	Arabic vocabulary	13K	13K
	English References/Sentence	4	4
Chinese-English(Zh-En)	Chinese Sentences	2,347	2,320
	Chinese words	70K	67K
	Chinese vocabulary	9K	9K
	English References/Sentence	4	4
Spanish-English(Es-En)	Spanish Sentences	1,452	1,780
	Spanish words	54K	58K
	Spanish vocabulary	6K	7K
	English References/Sentence	2	2

Table 5.2: Translation Task Data Statistics

For the static re-ranking experiments the following general procedure is followed. First the TTM component models are used to obtain a set of 1000-best translation lists for each target language sentence. Also, a set of real-valued sentence level features are extracted for each entry in the N-best list. The seven features used are:

1. Source (English) language language model log probability.
2. Log probability of target (Zh, Es or Ar) phrase given source phrase (En)
3. Log probability of target phrase given source phrase
4. Log probability of target phrase reordering

5. Log probability of target phrase insertion

6. Source word length

7. Target phrase length

Such a 1000-best list with the associated sentence level feature vectors is extracted for every sentence in both the development and the test set. In addition to the feature vectors the BLEU score statistics for each translation candidate - number of correct and matched n-gram matches and shortest reference length - are also extracted and stored in the list. Once this feature file is constructed the various training criteria are used to optimize the parameters on this feature file. There are seven parameters one for each of the features mentioned above that are optimized in our experiments. Finally, the translation performance in terms of the document level BLEU score is measured over the development set and the blind evaluation set under the optimized parameters.

The direct corpus BLEU objective function (described in Section 3.1.3) is non-differentiabe and so we cannot use the growth transformation based optimization in this case. Instead minimum error training based line search is used to optimize the corpus level BLEU score. The MMI and Expected BLEU training criterion (described in Section 3.1.2 and Section 3.1.4) are continuous and differentiable and so the growth transform based update procedure is applicable in these cases.

5.2 Static Re-Ranking Experiments

In the static re-ranking experiments the optimization is carried out over a fixed N-best list. The goal is to compare the different training and optimization criteria - MET line search based BLEU maximization and growth transformation based MMI training and expected BLEU training.

5.2.1 Expected BLEU Training

The goal is to maximize the Expected BLEU objective function given in Equation 4.16 using the growth transformation based update given in Equation 4.21. Figure 5.1, shows the plots of the expected BLEU objective function at each epoch

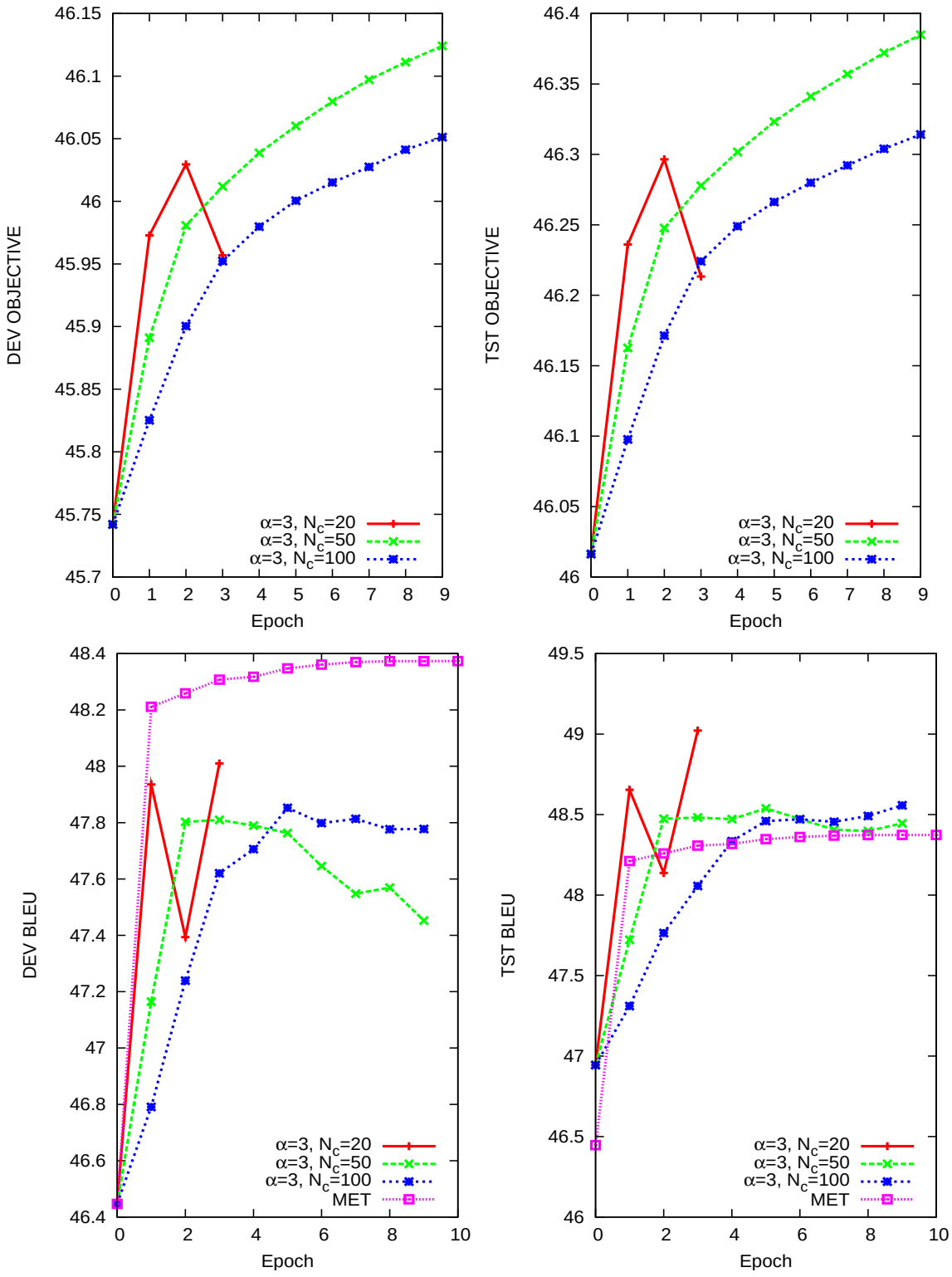

Figure 5.1: Arabic-English set: Expected BLEU Training over 1000-best list: Epochs vs 1-best BLEU. $N_c = [20, 50, 100]$ and $\alpha = 3$

(each iteration on a fixed N-best list is called an epoch) of the growth transform updates for both the development and test sets for the Ar-En translation task. The top two plots show the trend of the objective as a function of the epochs. The bottom two plots show the trend of the BLEU score evaluated for the development and test sets respectively.

For each epoch, there is an increase in the objective function value on both the development and test sets. This suggests that the growth transformation based updates result in a locally increasing objective function, as expected. The translation performance measured under BLEU on both the development set and test set shows that the BLEU score significantly increases after about two iterations as compared to the baseline (the BLEU score at epoch 0). From the top two plots in Figure 5.1, we observe that the objective function increases as the epochs progress both for the development and test sets. The bottom two plots in Figure 5.1 show translation performance measured under the BLEU criterion, over the entire test (development) corpus. The BLEU score peaks just after two iterations on both the development and test sets. For a posterior scale factor of $\alpha = 3$ and convergence factor of $N_c = 20$ there is a +1.6 absolute increase in BLEU when compared with the baseline for the development set and on the test set there is a +2.0 point improvement in BLEU over the baseline. As expected, MET line search performs significantly better (+1.8 increase in BLEU) on the development set as it exactly optimizes our test objective. However, on the test set both MET (+1.7 increase in BLEU) and expected BLEU (+2.0 increase in BLEU) achieve comparable performance.

5.2.2 MMI training with true reference

In order to handle multiple references, two different MMI objective functions were proposed - MMI with sum of references(Section 4.1.2) and MMI with product of references (Section 4.1.3). Furthermore, the true references are force aligned under the TTM to obtain the sentence level feature scores. The MMI training procedure uses the true references as the correct class to update towards. The MMI sum of references uses the objective function defined in Equation 4.9 with the parameter updates defined in Equation 4.11 while the MMI product of references uses the objective function defined in Equation 4.12 and the corresponding parameter update

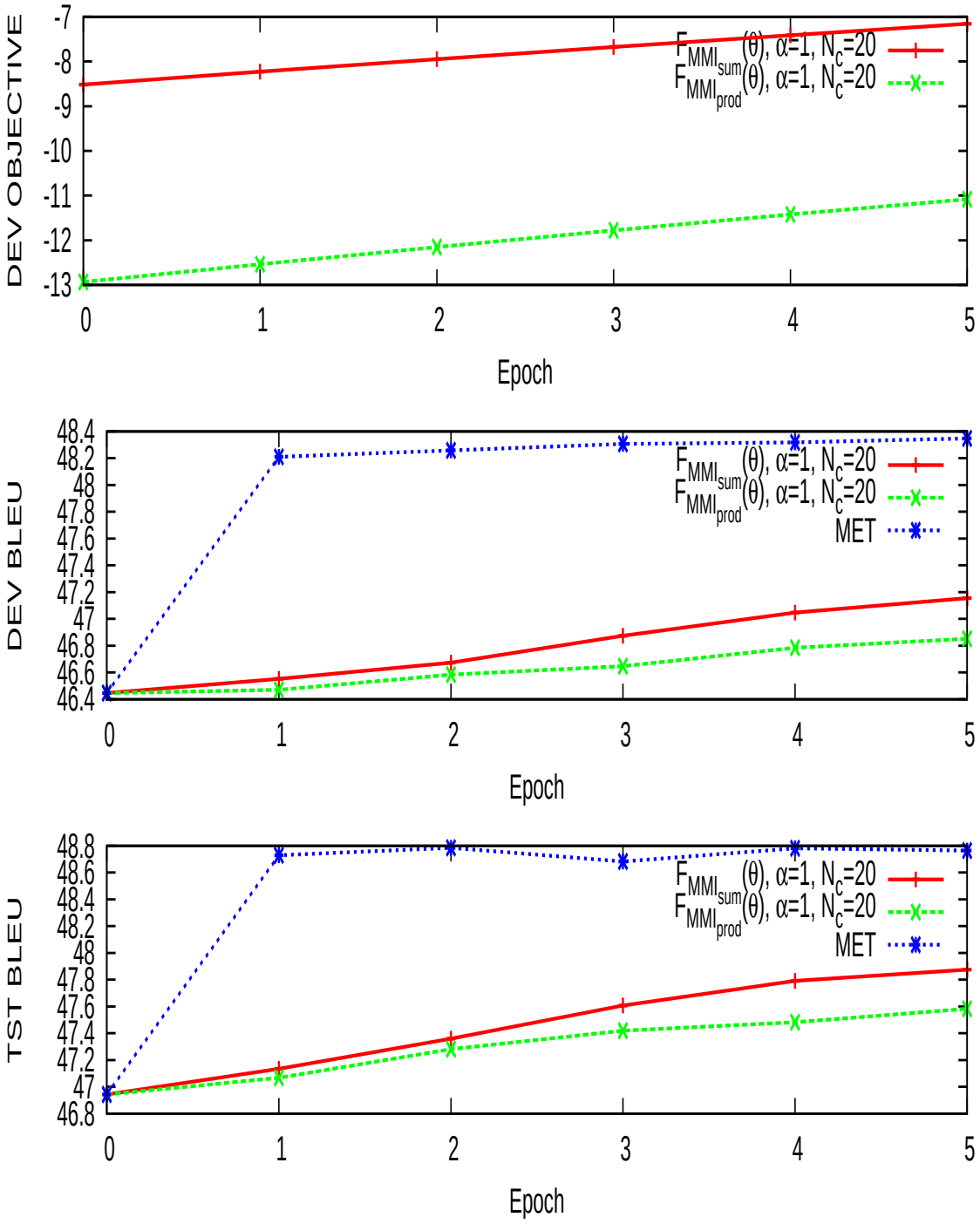

Figure 5.2: Arabic-English set: MMI sum or references vs MMI product of references training over a fixed 1000-best list using the true reference as correct class

in Equation 4.14.

In Figure 5.2, the growth transformation update is reported for a posterior scaling factor of $\alpha = 1$ and a convergence rate scaling factor of $N_c = 20$. The top figure plots the MMI objective functions as a function of the epochs. The middle figure plots the translation performance in BLEU as a function of the epochs over the development set and the bottom figure plots the same over the test set. The objective function increases smoothly with each epoch for both the MMI training criteria. For the MMI with product over references a +0.4 increase in BLEU over the baseline corresponds to a +0.6 increase in BLEU over the test set. For the MMI with sum over references objective function an increase of +0.8 BLEU over the development set corresponds to a +0.8 increase in BLEU over the test set. This suggests that there is no significant difference in the two objective criteria for MMI training.

However, both the MMI training criteria significantly underperform as compared to the MET training procedure as observed in Figure 5.2. This suggests that using the true references as the correct class in the MMI objective function is not a good choice in terms of translation performance when compared with MET.

5.2.3 MMI training with oracle-BLEU reference

Following the discussion in Section 4.1.4 and from the translation performance observed in Section 5.2.2 correctly identifying the correct class from the translation hypothesis space is crucial for MMI training to be effective. Instead of choosing the true reference as the correct class we choose the oracle-BLEU hypothesis (described in Section 4.1.4) as the correct class for MMI training. Again we use the form of the objective functions defined in Equation 4.12 and Equation 4.9 and their respective growth transform based updates defined in Equation 4.14 and Equation 4.11.

Figure 5.3, plots the MMI training results using the oracle-BLEU as the correct class for a posterior scaling factor of $\alpha = 1$ and a convergence rate scaling factor of $N_c = 20$. Using the oracle-BLEU hypothesis, significantly improves the performance of MMI training as observed in Figure 5.3. On the development set a +1.8 increase in BLEU relative to the baseline is observed which corresponds to an increase of +2.0

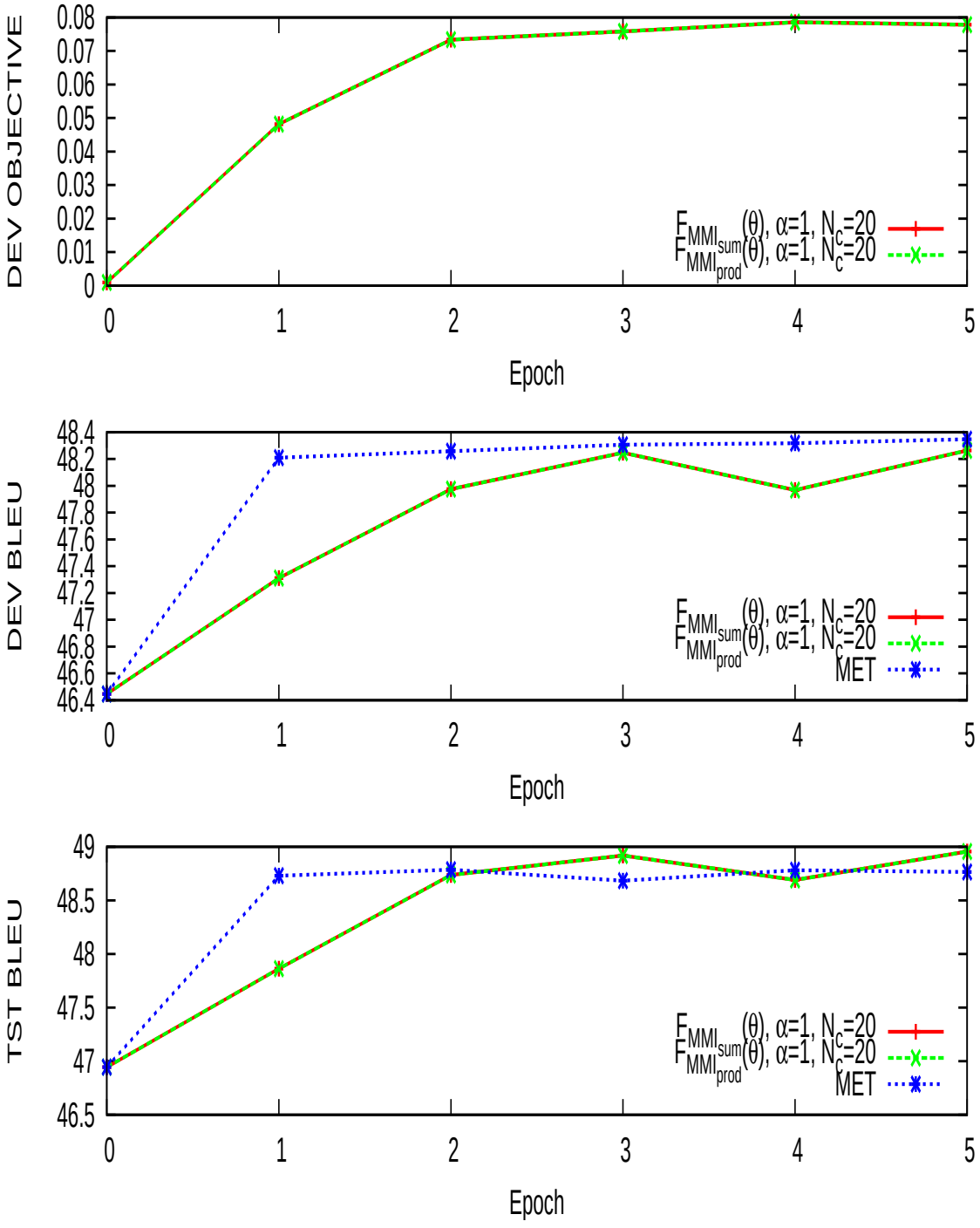

Figure 5.3: Arabic-English set: MMI sum or references vs MMI product of references training over a fixed 1000-best list using the oracle BLEU hypothesis used as the correct class

BLEU on the test set. This is comparable to the MET translation performance as observed in Figure 5.3. Furthermore, since the correct class is a single oracle-BLEU hypothesis both the MMI sum of references and MMI product of references gives identical updates and consequently identical translation performance.

5.2.4 Growth transform hyper-parameter tuning

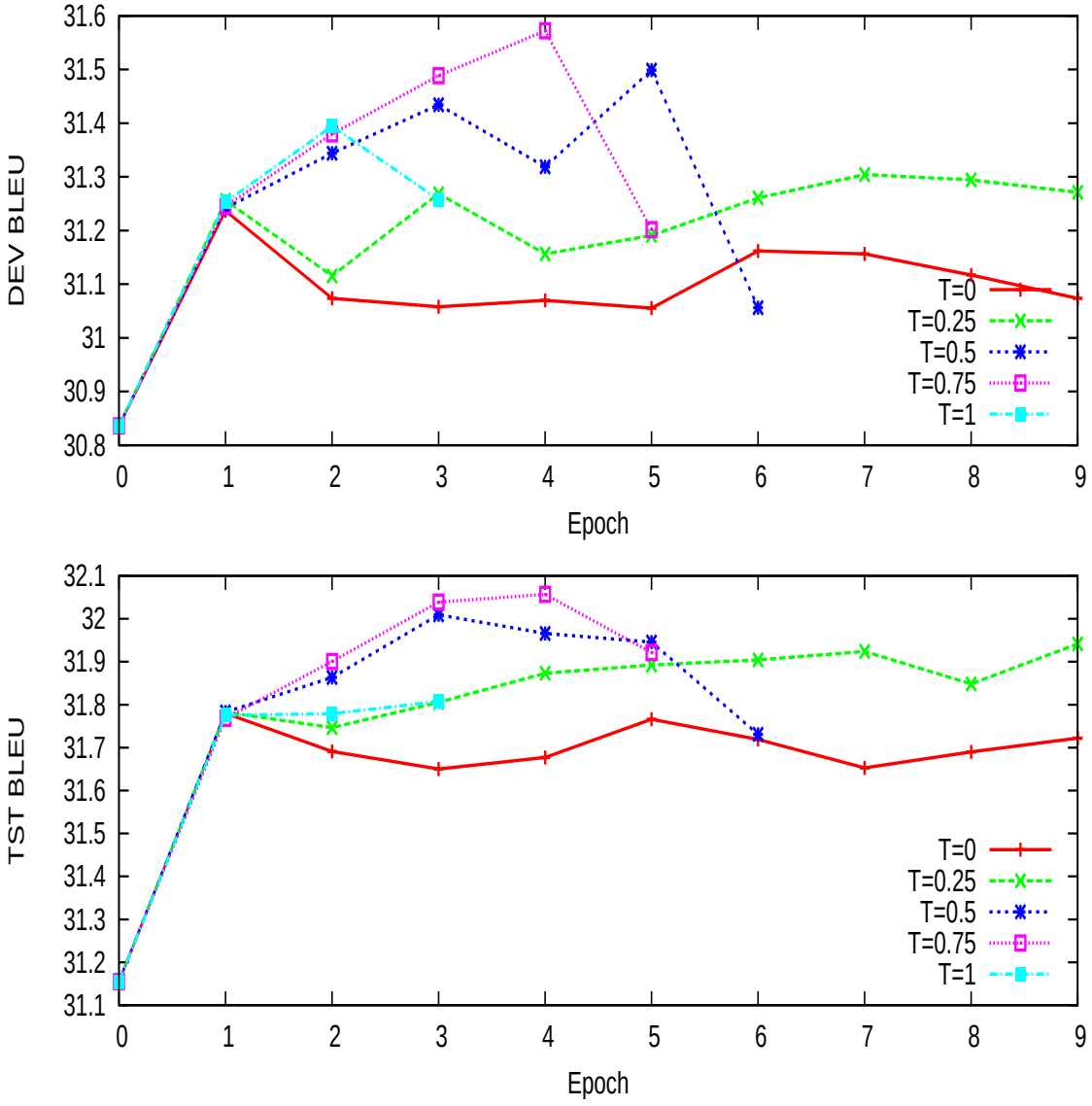

Figure 5.4: Chinese-English set: Effect of entropy regularization on translation performance $\alpha = 3$, $N_c = 20$

The growth transform based update procedure that we have presented so far have three hyper-parameters that we need to tune- posterior scaling(α), rate of conver-

gence scaling(N_c) and entropy regularization scaling (T). These hyper-parameters control different aspects of the optimization process. In this section we will evaluate the effect of these hyper-parameters on translation performance.

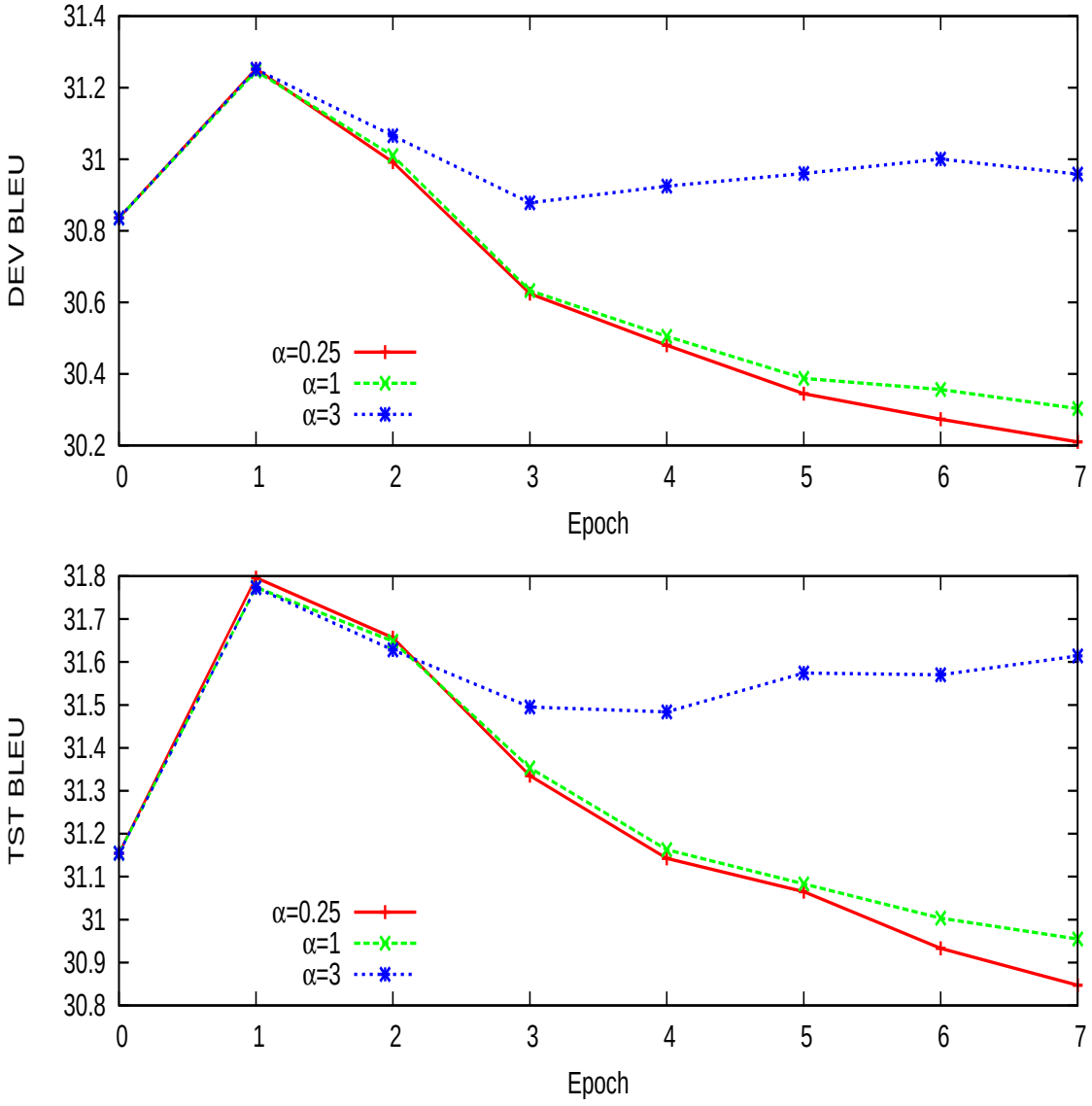

Figure 5.5: Chinese-English set: Effect of posterior scaling on translation performance $T = 0$, $N_c = 100$

In Equation 4.28 the entropy regularization term is added to the objective function. However, since there is no reason to believe that the regularization term should contribute equally to the optimization objective, a entropy regularization scaling factor T is introduced. Figure 5.4 illustrates the effect of T on translation performance. $T = 0$ implies that there is no regularization. For $0 \leq T \leq 1$, we observe that the

regularization term helps improve translation performance over both the development and test set. Also, we notice that the peak translation performance occurs after a few more iterations than at $T = 0$, suggesting that the regularization term also effects the convergence rate. By slowing down the rate of convergence, the optimization procedure is prevented from overfitting the decision surface.

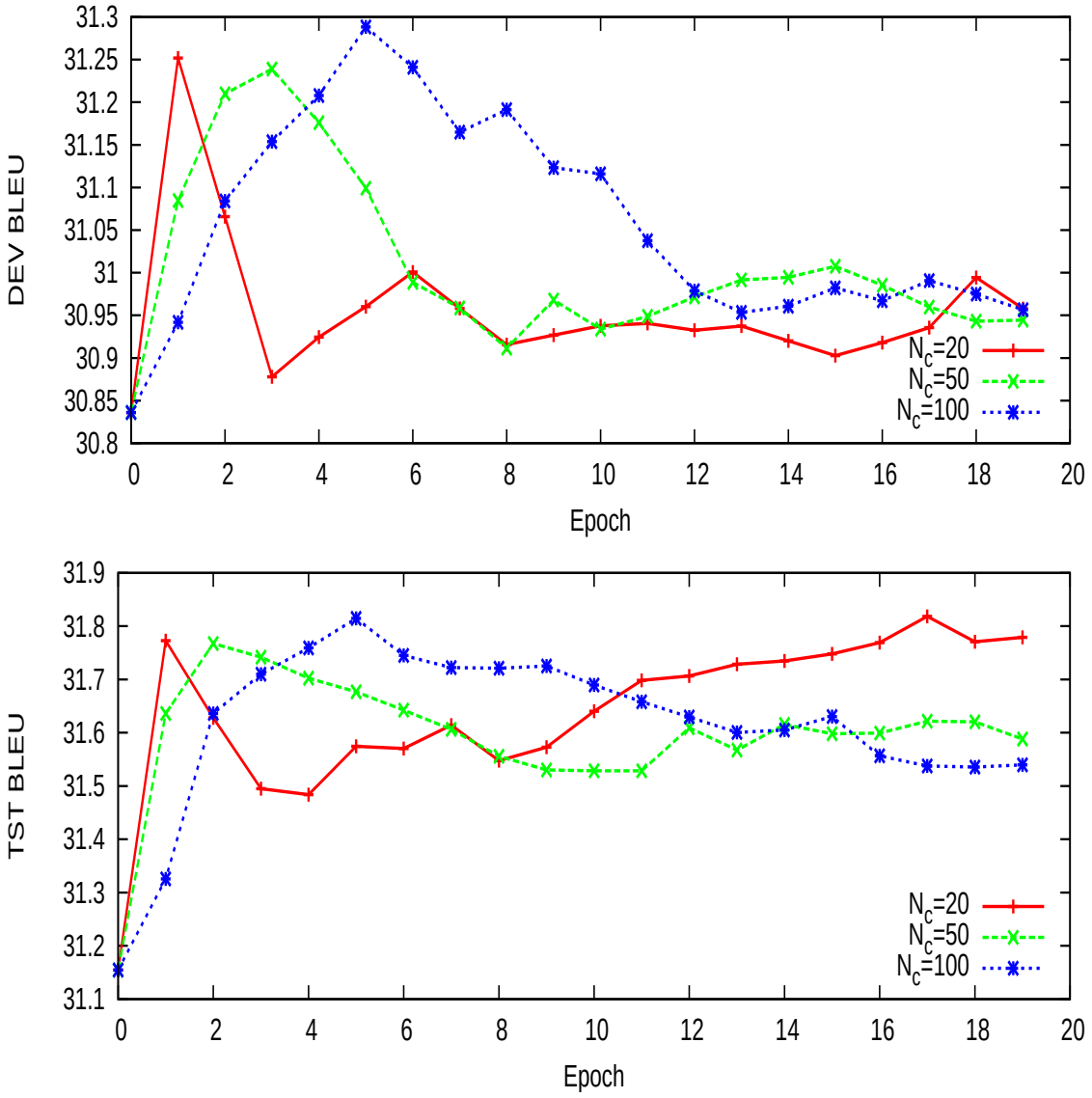

Figure 5.6: Chinese-English set: Effect of convergence rate on translation performance $T = 0$, $\alpha = 3$

Figure 5.5, shows the plot of translation performance on the development and test sets for different posterior scaling factors α. For the Chinese-English task there is no change in the peak translation performance for the different scale factors α. For higher values of α, since the probability mass is shifted towards the top few

hypotheses in the N-best lists, only the scores of these hypotheses will have an effect on the optimization procedure. For $\alpha = 3$, we observe that after about three iterations the translation performance does not drop significantly, evidence of the fact that the optimization procedure is focussed on the hypotheses closer to the top scoring hypothesis.

Figure 5.6, plots the translation performance on the development and test sets for different convergence rates N_c. The convergence rate scale factor N_c controls the rate of convergence of the growth transformation based updates. Thus as N_c increases we observe that it takes more iterations to achieve optimal translation performance on both the development and test sets.

5.2.5 Comparison with MET

Language Pair	Training Criterion	Optimization	DEV BLEU	TST BLEU
Es-En	MET	Line Search	43.4	42.5
	Expected BLEU	Growth Transform	43.1	42.3
	MMI w/ oracle	Growth Transform	**43.5**	42.8

Table 5.3: Comparison of MMI Training and MET and Expected BLEU training over fixed 1000-best list for Spanish-English language pair

In this section, the translation performance for the three language pairs - Ar-En, Es-En and Zh-En - are reported. We compare the 3 different objective criterion-MET (Equation 3.7) , MMI (Equation 3.2) and Expected BLEU (Equation 4.16), translation performance. From the experiments in Section 5.2.3 and Section 5.2.2, choosing the oracle-BLEU hypothesis performs significantly better than using the true reference. So, all the MMI training experiments reported henceforth will use the oracle as true reference.

Table 5.3 and Table 5.4 and Table 5.5 summarize the translation performance of the various training criteria over the different language pairs. In general, the growth transform based procedures outperform MET on the test set. However, growth transform based updates do not necessarily result in better translation performance over the training set when compared to MET. The reason is that MET optimization

Language Pair	Training Criterion	Optimization	DEV BLEU	TST BLEU
Ar-En	MET	Line Search	**48.4**	48.8
	Expected BLEU	Growth Transform	48.0	**49.1**
	MMI w/ oracle	Growth Transform	48.3	48.9

Table 5.4: Comparison of MMI Training and MET and Expected BLEU training over fixed 1000-best list for Arabic-English language pair

exactly optimizes the criterion used for the final evaluation and hence performs best on the trainig set. However, Expected BLEU and MMI training generalize better over the test set.

Language Pair	Training Criterion	Optimization	DEV BLEU	TST BLEU
Zh-En	MET	Line Search	**31.8**	31.9
	Expected BLEU	Growth Transform	31.6	**32.1**
	MMI w/ oracle	Growth Transform	31.4	32.0

Table 5.5: Comparison of MMI Training and MET and Expected BLEU training over fixed 1000-best list for Chinese-English language pair

Furthermore, we observe for Ar-En and Zh-En language pairs, expected BLEU training outperforms MET on the test set, while for the Es-En translation task the MMI algorithm outperforms the other training criteria. One possible reason for this is that the Es-En task has only two references/sentence (as opposed to the usual 4 references/sentence) resulting in a less robust BLEU estimate, and consequently a poorer representation of the error surface over which both MET line search and expected BLEU training procedures perform parameter search. The MMI objective function, is not prone to this problem as it attempts to directly maximizes the posterior probability of the "true" class, irrespective of the evaluation criterion being used.

5.3 Dynamic Re-Ranking

Thus far, the translation optimization has been on a fixed N-best list i.e. the optimization merely re-ranks the N-best list. This limits the performance of the optimization to the underlying hypothesis space which is fixed. Instead, the optimized

parameters can be used to re-generate a better set of N-best lists that better characterize the decision surface. Re-optimizing on this new set of N-best lists should result in further improvements in translation performance.

Language Pair	Training Criterion	DEV BLEU	TST BLEU
Es-En	Baseline(no optimization)	40.6	40.0
	MET	**45.9**	44.9
	Expected BLEU	45.7	44.7
	MMI w/ oracle	45.8	**45.1**

Table 5.6: Dynamic Re-Ranking: Comparing MMI Training and MET and Expected BLEU training

However, re-generating N-best lists with the updated parameters might generate a totally new set of N-best lists. In such a case, N-best lists are merged across iterations to obtain a more complete representation of the translation hypothesis space. The parameters are then re-optimized on this new merged N-best list to obtain a new set of parameter updates. This procedure can be iterated as follows:

1. Initialize the parameter vector θ and an empty set of N-best lists $\mathcal{N}$
2. For each sentence run the decoder to produce an N-best list under parameter θ, calculate the various loss statistics and add this to the set $\mathcal{N}$.
3. If the N-best list size does not change, or if we reached the maximum limit of iterations, stop. Else go to Step 4.
4. Adjust θ to maximize our objective function (MMI or expected BLEU or MET) over the current N-best list. Go to Step 2.

Finally, the iterative optimization procedure is evaluated by measuring translation performance in terms of the BLEU score. The iterative tuning is carried out over the development set. After the optimized parameters are obtained the translation performance is measured on a blind test set. Table 5.6, summarizes the translation performance for a Spanish-English translation task under the various training objective functions. Both Expected BLEU training and MMI training translation performance is comparable to MET line search.

5.4 Experiment Summary

We considered three different objective functions for training the parameters of an MT system - BLEU, Expected BLEU and MMI. For the BLEU training criterion we used MET line search was used to optimize the parameters. Growth transformation based updates were used to optimize the MMI and Expected BLEU training criteria. The translation performance was evaluated over three different language pairs- Arabic-English, Spanish-English and Chinese-English.

Translation performance was measured for two different setups - static re-ranking and dynamic re-ranking. In both the setups the growth transform based optimization gave comparable results to MET. Furthermore,the MMI training criteria was investigated in detail. Two MMI training objectives were considered for a multiple reference system - MMI sum of references and MMI product of references. Experiment results showed o significant difference in terms of translation performance on an Arabic to English translation task. In addition, the issue of choosing the correct class was also discussed. Experimentally it was observed that choosing the oracle-BLEU hypothesis as the correct class resulted in significantly better translation performance when compare to choosing the true reference as the correct class.

Finally, the effect of the various hyper-parameters in the growth transform based update procedure on translation performance was also investigated. In particular, the effect of the posterior scaling α, convergence rate scaling N_c and the regularization scaling T was studied. Entropy based regularization was shown to help improve translation performance.

Chapter 6

Experiments in Speech Translation

In this chapter, we investigate the performance of the generative model approach to speech translation on the Spanish to English translation task. The goal is to compare translation performance when translating from an ASR lattice as opposed to a single 1-best ASR output. First a detailed description of the data resources used in the task is presented. Next we highlight the difficulty of translating from ASR lattices. We also investigate the various pruning strategies for ASR lattices outlined in Section 2.9. Translation performance is measured under the BLEU [10] metric.

6.1 Experimental Setup

6.1.1 Training Corpus

We investigate the performance of the speech translation system on the TC-STAR Spanish to English (Es-En) European Parliamentary Speeches (EPPS) translation task [1]. The training corpus consists of final text editions of the European parliamentary sessions from April 1996 to October 2004. The corpus statistics for the bilingual sentence-aligned training data is given in Table 6.1. The training data is lower-cased and sentence aligned.

[1] http://www.tc-star.org

	Spanish	English
Sentences	1.28M	
Running Words with Punctuation	39M	37M
Running Words without Punctuation	35M	34M
Vocabulary	138K	95K
Singletons	48K	34K

Table 6.1: European Parliamentary Speeches Spanish-English Training Corpus Statistics

6.1.2 Translation Model Training

The speech translation system relies on an underlying phrase pair inventory that is extracted from the parallel training corpus. The translation model training is similar to the procedure outlined in Section 5.1.1. First, the MTTK toolkit [49] is used to generate word alignments for the parallel text in the Spanish to English and English to Spanish directions. Then phrase translation pairs are extracted using the union of the two sets of word alignments. The phrase pair inventory thus obtained is used to build the translation component models.

6.1.3 Speech Translation Data Setup

The speech-to-text translation corpus is based on seven Spanish parliamentary audio documents- two for development and five for test, as specified by the 2005 TC-STAR evaluation. The corpus statistics of the reference Spanish transcriptions provided for these audio documents is given in Table 6.2. Two English translations of each Spanish sentence transcription were commissioned for translation system scoring.

	Spanish DEV set	Spanish EVAL set
Sentences	2643	1073
Running Words with Punctuation	25704	21057
Running Words without Punctuation	25679	21054
Vocabulary	3032	3333
OOVs	113	157

Table 6.2: EPPS Development and Test Corpus Statistics

In addition to the seven audio documents, their Spanish text transcriptions, and

their corresponding English translations, we also have Spanish ASR word lattices in HTK format [51]. These were generated by the LIMSI ASR system similar to the Broadcast News system described in [52], incorporating cross-word triphone acoustic models and a 4-gram language model. The audio is segmented such that there is one lattice per sentence - there are 2643 and 1073 lattices for the development and test sets respectively.

6.1.4 ASR Lattice Pre-Processing

The ASR lattices which are in HTK format are first converted to a weighted finite state acceptor in ATT format [53]. However, the ASR lattices need to be pre-processed before converting them to the ATT format. A typical ASR lattice contains disfluencies which are usually present in speech (such as hesitations, repetitions and filler words such as uh, uhm etc.); and silence and pause markers which identify the unvoiced regions in the ASR lattice. Since these do not occur within the sentences in our bilingual text collections, it is difficult to extract phrases which cover such events in the ASR lattice. We map these symbols to ϵ. Consequently, some of the phrases extracted will span what the ASR system hypothesized as sentence breaks.

For the translation experiments, the timing information present in the lattice is ignored. Furthermore, the lattices which are converted to ATT format are reduced in size by applying ϵ-removal, determinization, and minimization [28, 29]. The arcs in the WFSA have word labels and the arc cost is the sum of the target acoustic model and target language model negative log likelihood probabilities. The acoustic model probabilities and the language model probabilities are combined using the appropriate word insertion penalty and language model scale factor optimized for ASR word error rate (WER) by LIMSI. The ASR word lattices are pruned as necessary, and after composition with the target phrase segmentation transducer, phrases are extracted up to five target words in length.

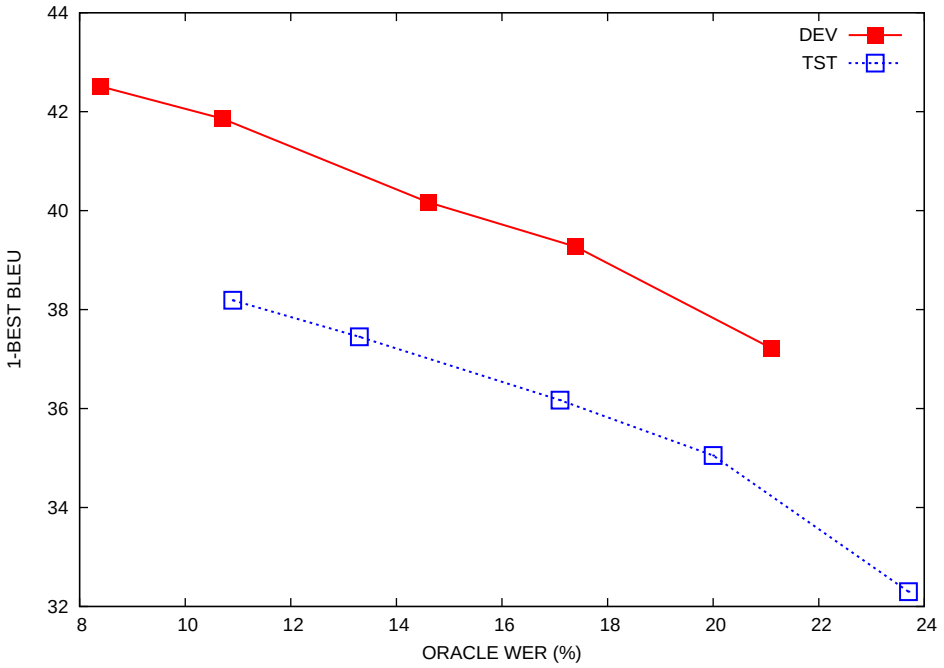

Figure 6.1: BLEU vs WER comparison for the oracle ASR Path

6.2 Oracle WER Experiments

The main motivation for translating from ASR word lattices is that the larger search space would allow the translation system to recover from ASR search errors, as opposed to translating a single 1-best string. In this experiment, the goal is to investigate if better translations can be produced by looking at translation candidates in the ASR lattice other than the ASR 1-best. More specifically, the objective is to evaluate if lower WER correlates positively with improved translation performance. Also, of interest is to characterize the degree of difficulty of the translation task: how dense the ASR lattice must be to obtain significant translation performance gains.

In this experiment, the translation performance in terms of the BLEU score is plotted as a function of the lattice oracle WER. The lattice oracle WER is the path in the lattice that is closest to the reference ASR transcription under the levenshtein distance [54]. The path corresponding to the oracle WER is called the oracle ASR path. In our experiments, the ASR lattice is first pruned to obtain a set of lattices at different densities. Next, for each pruned set of lattices a set of oracle

translation candidates is translated and the translation performance is noted for the corresponding oracle WER.

Figure 6.1 shows plots of the translation performance as a function of the oracle WER for both the development and test sets. We observe that reduction in the oracle WER results in improved BLEU performance. This is consistent with our hypothesis that exploiting the larger search space encoded by the lattice can help improve translation performance by allowing the SMT component to select better translation candidates. However, a 60% decrease in ASR WER corresponds to only a 13% relative improvement in BLEU. This implies that speech translation is a difficult task.

6.3 Evaluating Translation of ASR Lattices

In this section, ASR word lattices translation experiments on the EPPS corpus described in 6.1.1 are presented. Parameter optimization using MET line search over a 1000-best list is used to tune the translation performance on a development set. The optimized parameters are then used to evaluate translation performance on a blind test set.

Spanish Source	DEV BLEU	EVAL BLEU
Reference Transcription	48.6	42.4
ASR 1-best	39.5	32.5
ASR Lattice w/ word posterior pruning	40.7	**33.6**
ASR Lattice w/ phrase posterior pruning	40.6	33.3

Table 6.3: EPPS Spanish-English Translation Performance

Table 6.3 , reports the translation performance over the development and test sets for the different input conditions - manual reference transcriptions, ASR 1-best hypothesis, ASR lattice with word level and phrase level pruning.

The manual reference transcriptions correspond to the truth. Hence the translation performance obtained from translating these correspond to the best possible translation performance if the speech translation had access to the true transcript.

Translating the ASR 1-best is significantly worse than the reference transcriptions. The errors in the ASR 1-best output severely limit the translation performance. A +1.1 improvement in the BLEU score is obtained over the test set by translating the ASR lattice. This suggests that the SMT system is able to take advantage of the the larger search space of the ASR lattice in searching for a suitable candidate translation. Finally, the two pruning strategies performed similarly in terms of translation performance.

Table 6.4 shows a translation example for the ASR 1-best and the ASR lattice for a particular sentence in the EPPS test corpus. The highlighted words in the table inidicate the regions that differ from the ASR 1-best translation. As observed the ASR Lattice translation is closer to the reference than the ASR 1-best translation.

Input	Translation
Reference	1. in accordance with the committee on budgets the period for tabling amendments for the second reading of the european union budget will end on wednesday the first of december at twelve noon. 2. in agreement with the committee on budgets the deadline for the presentation of projects of amendment for the second reading of the european union budget will finish on wednesday first of december at twelve noon.
ASR 1-best	according to the committee on budgets of the deadline for the submission of projects amendment concerning the second reading of the budget union will end on wednesday , one of the twelve noon
ASR Lattice	**in accordance with** the committee on budgets of the deadline for the submission of projects amendment concerning the second reading of the budget union will end on wednesday , one of the twelve noon

Table 6.4: Translation examples for the EPPS Eval05 test set : ASR 1-best and ASR Lattice with MET line search based optimization

6.4 Evaluating Translation Quality

In comparing the speech and text translation systems, we are interested in quantifying the benefit of translating ASR word lattices vs. a single string (ASR 1-best or reference transcription). Two sets of measure are provided to evaluate the translation quality - *Average Phrase Density* and N-best list translation quality.

6.4.1 Average Phrase Density

Improved phrasal coverage is known to correlate with improved translation performance [55]. Phrasal coverage is defined as the proportion of phrases in the test set covered by the underlying phrase-pair inventory. The *Average Phrase Density* measures the phrase coverage of the test set under a particular phrase-pair inventory. So the higher the average phrase density, the better the phrasal coverage. For each word in a target (source) segment, the number of phrases that covered that word in alternative segmentations of that segment are obtained. The number of phrases per word is then averaged over the test set. The average phrase density is expressed as the average number of phrases per word per acoustic segment.

$$Average\ Phrase\ Density = \frac{1}{N_{trg}} \sum_{i=1}^{N_{trg}} \left(\frac{\sum_{w_i=1}^{N_{w_i}} n_p(w_i)}{N_{w_i}} \right) \tag{6.1}$$

where, N_{trg} is the number of target (source) acoustic segments, N_{w_i} is the number of words in segment i, and $n_p(w_i)$ is the number of phrases containing the word w_i. Average phrase density is calculated over both the source and target segments.

	Phrase Pair Inventory Statistics			Average Phrase Density	
Translation Input	# Spanish Phrases	# English Phrases	# Phrase-Pairs	Spanish	English
ASR 1-best	20.3K	15.6K	38.6K	5.1	4.1
ASR Lattice w/ word pruning	26.3K	16.8K	51.1K	6.0	4.9
ASR Lattice w/ phrase pruning	27K	16.9K	52.8K	5.8	4.7

Table 6.5: Phrase Pair Inventory Statistics and Average Phrase Density for EPPS test set

From Table 6.5, the phrase extraction procedure is able to extract a substantially larger number of target phrases from the ASR lattice relative to the ASR 1-best string. Also, the phrase-pair inventory is larger in the ASR lattice case (34% relative to that of ASR 1-best). This corresponds to a relative increase in the average target phrase density (17% relative to the ASR 1-best) and the average source phrase density (19% relative to the ASR 1-best). The relative increase in the average phrase density for both the source and target language suggests that extracting phrases from lattices leads to improved phrase coverage. Furthermore, this correlates with

the improved translation performance observed in Table 6.3. There is no significant difference in the average phrase density and the size of the phrase pair inventory for the two ASR pruning strategies, suggesting that the two approaches result in similar phrasal coverage and hence similar translation performance.

6.4.2 N-best List Translation Quality

Translation Input	oracle-best BLEU	
	DEV	TST
Reference Transcription	68.4	60.3
ASR 1-best	55.6	48.1
ASR Lattice w/ word pruning	57.8	49.7
ASR Lattice w/ phrase pruning	57.8	49.4

Table 6.6: Oracle-best BLEU for EPPS development and test set measured over a 1000-best translation list

The goal of this experiment is to evaluate the quality of the translation hypotheses generated by the translation system for a fixed phrase pair inventory and different input conditions - Reference transcriptions, ASR 1-best and ASR lattice. The quality of the N-best translation space is measured in terms of the *oracle-best* BLEU. The oracle-best BLEU for a set of N-best lists is obtained using the following procedure:

- For each sentence to be translated, the oracle-BLEU hypothesis is selected from the corresponding N-best list of translations for that sentence. The oracle-BLEU hypothesis in the N-best list of translations is defined as the translation candidate with the highest sentence level BLEU score.

- Next, the oracle-BLEU hypotheses are collected over all sentences in the development (test) corpus and the corpus level BLEU thus obtained is the oracle-best BLEU.

The oracle-best BLEU over N-best lists quantifies the quality of the translation hypotheses generated by the translation system for a fixed phrase pair inventory. In Table 6.6, the oracle BLEU for the ASR lattice is higher than that of the ASR 1-best, which suggests that translating from ASR lattices results in a better translation hypothesis space and consequently better translations as observed in Table 6.3. Also,

for the word and phrase level pruning strategies the translation hypothesis space is similar in quality and hence there is no appreciable difference in the BLEU score in Table 6.3.

6.5 Discriminative Training for Speech Translation

Translation Input	Training Criterion	BLEU	
		DEV	TST
ASR 1-best	MET	39.5	32.5
	MMI	35.8	**33.2**
	Expected BLEU	38.1	32.3
ASR Lattice w/ word pruning	MET	40.7	33.6
	MMI	36.8	34.3
	Expected BLEU	37.2	**34.6**
ASR Lattice w/ phrase pruning	MET	40.6	33.3
	MMI	36.9	34.1
	Expected BLEU	37.8	**34.5**

Table 6.7: Comparing MMI and Expected BLEU and MET training criteria for the Spanish-English ASR translation task

In the speech translation experiments thus far, MET line search (Equation 3.7) has been applied to optimize the MT parameters. In this experiment, the two discriminative training criteria proposed in Chapter 3 - MMI Training (Equation 3.2) and Expected BLEU Training (Equation 4.16) - are compared with MET. For MMI training the oracle reference is chosen as the correct class. Growth transformation based updates (Equation 3.20) are used to optimize the MMI and Expected BLEU training criteria.

For the ASR 1-best input, the MMI training criterion gives the best translation performance over the test set. Although on the development set the MET procedure gives the best translation performance it does not generalize well over the test set. This suggest that the MET optimization has led to overfitting of the MT parameters. Even when optimizing for the ASR lattice translation case, MET optimization seems to overfit on the development set resulting in a poorer translation performance on

the test set. The MMI and Expected BLEU training perform significantly better. Specifically, for the ASR lattice translation the Expected BLEU training algorithm results in the best translation performance. Overall, there is a gain of +1.4 BLEU points when translating from the ASR lattice as opposed to the ASR 1-best.

6.6 Experiment Summary

In this chapter speech translation experiments were presented. First, the oracle WER experiments were investigated to characterize the degree of difficulty translating from speech entails. It was shown that obtaining about 13% relative improvement in BLEU required about a 60% drop in the lattice oracle WER. Although, this is a difficult objective it does suggest that improvements in translation can be obtained by looking deeper in to the lattice.

Following the oracle WER experiments, translation of ASR 1-best was compared with translation of the ASR lattice. Significant improvements in the BLEU score was observed when translating from lattices. Also, the word level and phrase level pruning of the ASR lattices resulted in similar translation performance. In addition, experiments were provided to analyze the quality of the translation N-best lists and the phrasal coverage. It was observed that translation from ASR lattices resulted in significant improvement in phrasal coverage and also improvements in the translation hypothesis list quality, two factors that are known to correlate well with improved translation performance [55].

Finally, discriminative training experiments were presented for the speech translation task. Both MMI and Expected BLEU were shown to significantly outperform MET optimization.

Part II

Statistical Speech Recognition

Chapter 7

Overview of Statistical Speech Recognition

Automatic Speech Recognition (ASR) is a sequential pattern recognition problem in which the patterns to be hypothesized are words while the evidence presented to the recognizer is the acoustics of a spoken utterance. Given an acoustic signal, a speech recognizer attempts to classify it as the sequence of words that was spoken. One of the main applications of speech recognition is to transform spoken utterances in to a written and structured document. In this chapter, we will present a brief overview of the statistical approach to ASR.

7.1 Mathematical Formulation Review

The goal of a speech recognition system is to automatically transcribe speech to text. A precise mathematical formulation of the speech recognition problem is needed if we are to discuss the problem of speech recognizer design. Let $\mathbf{O}$ be the acoustic evidence or observations on the basis of which we will make a decision about which words were spoken. We can assume that

$$\mathbf{O} = o_1, o_2, \cdots, o_T \quad o_i \in \mathcal{O}$$

is a sequence of symbols drawn from some (possibly infinite) space $\mathcal{O}$. Let

$$\mathbf{W} = w_1, w_2 \cdots w_n \quad w_i \in \mathcal{V}$$

denote a n word sequence drawn from a vocabulary $\mathcal{V}$. Then given $P(\mathbf{W}|\mathbf{O})$, the posterior probability that the words $\mathbf{W}$ were spoken given the acoustic evidence

$\mathbf{O}$, the Maximum a Posteriori (MAP) decoder [2] chooses the word sequence $\hat{\mathbf{W}}$ satisfying

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}}\ P(\mathbf{W}|\mathbf{A}) \tag{7.1}$$

In other words, the recognizer will pick the most likely word string $\hat{\mathbf{W}}$ given the observed acoustic evidence. Applying Bayes' rule, the posterior probability $P(\mathbf{W}|\mathbf{O})$ can be re-written as

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}}\ \frac{P(\mathbf{O}|\mathbf{W})P(\mathbf{W})}{P(\mathbf{O})} \tag{7.2}$$

Since, the maximization in Equation 7.2 is independent of the acoustic evidence $\mathbf{O}$, we have the following search problem

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}}\ \underbrace{P(\mathbf{O}|\mathbf{W})}_{\substack{Acoustic\\ Model}}\ \underbrace{P(\mathbf{W})}_{\substack{Language\\ Model}} \tag{7.3}$$

The search problem is now conveniently decomposed in to two different modeling problems - acoustic modeling and language modeling. In the next section we will briefly describe the different components of the MAP decoder.

7.1.1 Language Modeling

The main task of the language model is to assign a probability to the word sequence $\mathbf{W} = w_1, w_2, \cdots, w_n$. The language model helps in guiding and reducing the acoustic search space by providing contextual information. The probability $P(\mathbf{W})$ can be decomposed as

$$P(\mathbf{W}) = \prod_{i=1}^{n} P(w_i|w_1, w_2, \cdots, w_{i-1})$$

where $P(w_i|w_1, w_2, \cdots, w_{i-1})$ is the probability that w_i will be spoken given that words $w_1, w_2, \cdots, w_{i-1}$ were said previously. The sequence $w_1, w_2, \cdots, w_{i-1}$ is referred to as the *history* and is denoted by $\mathbf{h_i}$. In most ASR systems, the language model most frequently used is the n-gram language model where the history is restricted to a context of $n-1$ previous words. For such a n-gram model,

$$P(\mathbf{W}) = \prod_{i=1}^{n} P(w_i|w_{i-n+1}, \cdots, w_{i-1})$$

In most present-day speech recognizers, 4-gram and 5-gram models are frequently used.

7.1.2 Acoustic Modeling using HMMs

The acoustic likelihood is typically modeled by a Hidden Markov Model (HMM) [56], which forms the central component of most speech recognizers today. A HMM is a statistical model well suited for modeling a sequence of observation symbols (discrete or continuous) and has been quite successful in capturing the acoustic and temporal characteristics of speech. We will now describe how the HMM can be use to model speech. Formally, an HMM is defined by:

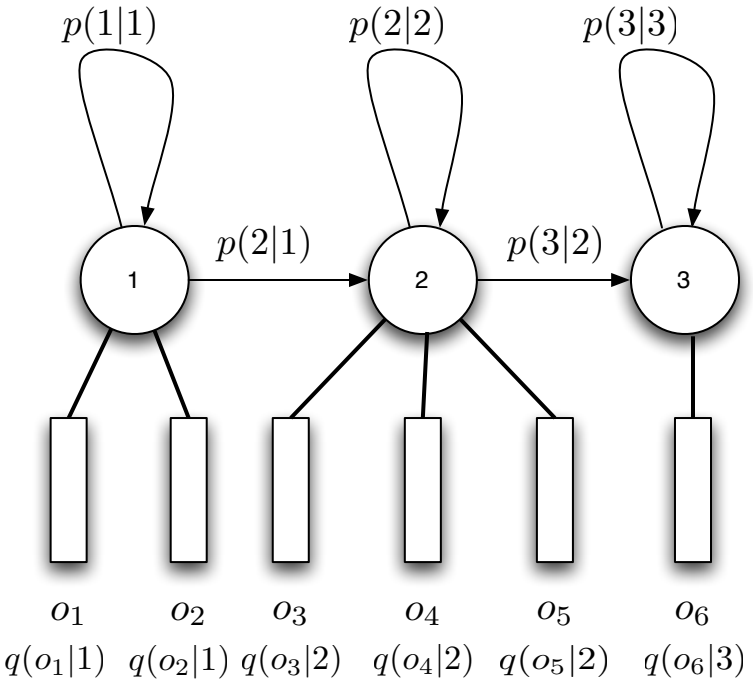

Figure 7.1: 3-state left to right HMM with discrete observations

- A finite state space $\mathcal{S} = \{1, 2, \cdots, S\}$. A state s_t at time t takes values $s_t \in \mathcal{S}$

- An observable output alphabet $\mathcal{O} = \{1, 2, \cdots, M\}$. If the observation space is continuous then we define $\mathcal{O} \in \mathbb{R}^d$.

- A set of state transition probabilities $p(s_t|s_{t-1})$, defined as the probability of transitioning from state s_{t-1} at time $t-1$ to state s_t at time t.

- A set of output probability distributions $q(o|s_t)$, defined as the probability of observing observation symbol o at time t in state s_t. If the observation space

is discrete, then the output probability distribution is given by:

$$q(k|s_t = j),\ 1 \leq k \leq M,\ 1 \leq j \leq S$$

If the observation space is continuous then we specify the output probability distribution as a probability density function. The probability density function is typically approximated by a mixture of M gaussians.

$$q(\mathbf{o}|s_t = j) = \sum_{m=1}^{M} c_{jm}\mathcal{N}(\mu_{\mathbf{jm}}, \mathbf{\Sigma_{jm}}) \tag{7.4}$$

$$= \sum_{m=1}^{M} c_{jm}\ \frac{1}{(2\pi)^{d/2}|\Sigma_{jm}|}\ exp^{-\frac{1}{2}(\mathbf{o}-\mu_{\mathbf{jm}})^T\mathbf{\Sigma_{jm}^{-1}}(\mathbf{o}-\mu_{\mathbf{jm}})} \tag{7.5}$$

where,
$\mathbf{o}$ = a real valued observation vector drawn from $\mathbb{R}^d$,
c_{jm} = is the mixture weight for gaussian component m in state j,
$\mu_{\mathbf{jm}}$ = is the mean vector for gaussian component m in state j,
$\mathbf{\Sigma_{jm}}$ = is the covariance matrix for gaussian component m in state j,
and $\sum_{m=1}^{M} c_{jm} = 1 \quad c_{jm} \geq 0, 1 \leq j \leq S$

The HMM is thus parameterized by the state distribution and the output distribution. Furthermore, when using the HMM to model speech we make the following independence assumptions:

- The probability of being in state s_t at time t depends only on the previous state s_{t-1} at time $t-1$, regardless of the previous state transition history.
- The output probability at time t depends only on the current state s_t at time t.

In most large vocabulary speech recognizers, the HMM is modeled at the sub-word level as *triphones*, which are phonemes with left and right context. Each triphone is associated with a HMM according to some predefined topology. Figure 7.1 is a typical topology used in most ASR systems. The word level HMM is then obtained by concatenating the phonetic HMMs that constitute that word.
Let $\mathbb{M} = \{m_1, m_2, \cdots, m_U\}$ be the set of elementary phonetic HMMs and let $\theta = \{\theta_1, \theta_2, \cdots, \theta_U\}$ be the set of associated HMM parameters. Then $\mathcal{M}_w$ is the HMM corresponding to word w obtained by concatenating the phonetic HMMs from

the set $\mathbb{M}$ associated with the word w.

Given an observation sequence $\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \cdots, \mathbf{o}_T$ and the HMM model $\mathcal{M}$ with parameters θ we can obtain the likelihood of the HMM having generated that particular observation sequence:

$$P(\mathbf{O}|\mathcal{M}, \theta) = \sum_{s_1, s_2, \cdots, s_T} \prod_{t=1}^{T} p(s_t|s_{t-1})\ q(\mathbf{o}_t|s_t) \tag{7.6}$$

Here, $s_1, s_2, \cdots, s_T$ is the state sequences associated with the observations sequence $\mathbf{O}$.

7.2 Estimating the HMM Parameters

The HMM parameters need to be estimated from data, before we can use the HMM in a speech recognizer. The learning problem can be state thus: Given the acoustic evidence $\mathbf{O}$ and the transcript $\bar{\mathbf{W}}$ corresponding to the acoustic evidence, we need to estimate the HMM parameters θ (the state transition distributions and the output probability distribution parameters). The two most common approaches to estimating the HMM parameters are Maximum Likelihood (ML) [2] and Maximum Mutual Information (MMI) [37] criteria. The following sections give a brief overview of these two approaches.

7.2.1 Maximum Likelihood Estimation (MLE)

Given a training corpus of R acoustic observations $\mathbf{O}^{(1)}, \mathbf{O}^{(2)}, \cdots, \mathbf{O}^{(R)}$ and the corresponding transcriptions $\bar{\mathbf{W}}^{(1)}, \bar{\mathbf{W}}^{(2)}, \cdots, \bar{\mathbf{W}}^{(R)}$, the maximum likelihood estimation procedure attempts to find the HMM parameters θ^* which maximizes the likelihood of the observed acoustic data

$$\theta^* = \underset{\theta}{\text{argmax}} \sum_{r=1}^{R} \log P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}) \tag{7.7}$$

$$= \underset{\theta}{\text{argmax}} \sum_{r=1}^{R} \sum_{s_1^T \in \mathbf{S}^{(r)}} \log P_\theta(\mathbf{O}^{(r)}, s_1^T|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}) \tag{7.8}$$

where, $\mathbf{S}^{(r)}$ is the set of all possible state sequences through the HMM network $\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}$ corresponding to the observation sequence $\mathbf{O}^{(r)}$.

In general, it is difficult to optimize the likelihood in Equation 7.8 directly. Instead, we use Expectation Maximization (EM), an iterative procedure that locally maximizes a lower bound on the likelihood. The EM algorithm is a general method of finding the maximum-likelihood estimate of the parameters of an underlying distribution from a given data set when data are incomplete or have missing values [57].

The EM algorithm is a two-step iterative procedure described as follows:

E step The expectation step (E-step) computes the expectation of the complete data likelihood $P_\theta(\mathbf{O}^{(r)}, s_1^T|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}})$ with respect to the hidden state sequence given the observation and the current parameter estimate $\theta^{(i)}$. We define the auxiliary function

$$Q(\theta, \theta^{(i)}) = \sum_{r=1}^{R} \sum_{s_1^T} P(s_1^T|\mathbf{O}^{(r)}, \mathcal{M}_{\bar{\mathbf{W}}^{(r)}}; \theta^{(i)}) \log P(\mathbf{O}^{(r)}, s_1^T|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}; \theta) \quad (7.9)$$

Here, $\theta^{(i)}$ are the current parameters that we evaluate the expectation under and θ are the new parameters we optimize to increase $Q(\theta, \theta^{(i)})$ In the remaining discussion we will use the notation $P_\theta(\mathbf{O}^{(r)}, s_1^T|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}})$ instead of $P(\mathbf{O}^{(r)}, s_1^T|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}; \theta)$.

M-step In the maximization step (M step) we maximize the expectation calculated in the E-step.

$$\theta^{(i+1)} = \underset{\theta}{\operatorname{argmax}}\ Q(\theta, \theta^{(i)}) \quad (7.10)$$

These two steps are repeated as necessary. Each iteration is guaranteed to increase the log-likelihood and the algorithm is guaranteed to converge to a local maximum of the likelihood function. The ML criterion is the most widely used criterion in training a speech recognition due to its simplicity, ease of implementation and due to attractive statistical properties of the MLE such as consistency and efficiency [58].

However, the MLE approach gives optimal estimates (in the sense of minimum variance unbiased estimates) under three assumptions. First, the model correctly represents the underlying stochastic process. In the case of ASR, it is well known that an HMM is not the true model for speech; in fact the true model is unknown. The problem arises due to the various conditional independence assumptions that

underlie HMM models. Given these assumptions, it is unlikely that the processes that actually generate speech can be closely modelled by HMMs. Second, we have an infinite amount of training data which is of course impossible to satisfy in practice. Lastly, we can find the true global maximum of the likelihood surface. However, the likelihood defined for the speech recognition case is riddled with various local maxima and so we can only guarantee a local maximum. Furthermore, MLE is only concerned with maximizing the likelihood of the training data given the model corresponding to the data. The models from other classes do not participate in the parameter re-estimation. Therefore, under MLE each HMM is trained only to generate high probabilities for its own class, without discriminating against competing classes. Consequently, the MLE objective function is not directly related to the objective of reducing the error rate of the recognition task.

As an alternative we can use discriminative estimation procedures that directly attempt to optimize recognition performance. In general, discriminative criteria not only try to maximize the class conditional probability of the acoustic evidence given the correct classes but, also try to minimize the class conditional probabilities of the corresponding competing classes. In the next section we briefly describe one of the most popular discriminative criterion, i.e. the Maximum Mutual Information criterion

7.2.2 Maximum Mutual Information Estimation (MMIE)

Maximum mutual information estimation [37], attempts to find the HMM parameters θ that maximizes the mutual information between the acoustic observation sequence $\mathbf{O}$ and the corresponding word sequence $\mathbf{W}$.

$$I_\theta(\mathbf{O}, \mathbf{W}) = \frac{P_\theta(\mathbf{O}, \mathbf{W})}{P_\theta(\mathbf{O})P(\mathbf{W})} \tag{7.11}$$

If the language model is assumed to be independent of the acoustic model parameters, then maximizing the MMI criterion is equivalent ot maximizing the Conditional Maximum Likelihood (CML) criterion [59]. Throughout this dissertation the term MMI and CML will be used interchangeably. The CML criterion tries to improve the a posteriori probability of the correct sentence hypothesis given the acoustic evidence. Since this is also the probability used in MAP decoding (Equation 7.1), the relation of the CML objective and the Sentence Error Rate on the acoustic training

set is much more intuitive than it is with MLE.

Then for R training observation sequences $\mathbf{O}^{(1)}, \mathbf{O}^{(2)}, \cdots, \mathbf{O}^{(R)}$ and the corresponding transcriptions $\bar{\mathbf{W}}^{(1)}, \bar{\mathbf{W}}^{(2)}, \cdots, \bar{\mathbf{W}}^{(R)}$ the CML objective function is given by:

$$\theta^* = \underset{\theta}{\text{argmax}} \sum_{r=1}^{R} \log P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}) \tag{7.12}$$

$$= \underset{\theta}{\text{argmax}} \sum_{r=1}^{R} \log \frac{P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}})P(\bar{\mathbf{W}}^{(r)})}{\sum_{\mathbf{W}' \in \mathcal{W}} P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\mathbf{W}'})P(\mathbf{W}')} \tag{7.13}$$

Here, the composite HMM model $\mathcal{M}_{\bar{\mathbf{W}}^{(r)}}$ is called the *numerator* model corresponding to the correct transcription. The summation in the denominator of Equation 7.13 is taken over all possible word sequences $\mathbf{W}'$ allowable in the task and it can be replaced by

$$P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\mathbf{gen}}) = \sum_{\mathbf{W}' \in \mathcal{W}} P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\mathbf{W}'})P(\mathbf{W}') \tag{7.14}$$

where $\mathcal{M}_{\mathbf{gen}}$ is a model constructed such that for all paths in every $\mathcal{M}_{\mathbf{W}'}$ there is a corresponding path of equal probability in $\mathcal{M}_{\mathbf{gen}}$. It is usually assumed that the model used for recognition is a reasonable approximation of $\mathcal{M}_{\mathbf{gen}}$.

Maximizing the MMI criterion (Equation 7.13) leads to discriminant models since it implies that the numerator HMM parameters are adapted to increase the term $P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\bar{\mathbf{W}}^{(r)}})$ while simultaneously trying to drive down the likelihood of the denominator term $P_\theta(\mathbf{O}^{(r)}|\mathcal{M}_{\mathbf{gen}})$. The denominator term dominates the computation and this will depend on the size of the vocabulary, the grammar and any contextual constraints. In many practical situations, for example where cross-word context dependent acoustic models are used in conjunction with a long span language model, the construction of a complete model for $\mathcal{M}_{\mathbf{gen}}$ is intractable. In practice, the denominator term is approximated either by N-best lists [60] or lattices [61, 62]. The N-best lists or the lattices are generated via a recognition pass on each of the training utterances.

The most widely used algorithm for MMIE is the *Extended Baum Welch* (EBW) algorithm introduced in [46], to optimize rational objective functions. The authors in [46], applied the EBW algorithm to discriminatively train the parameters of a

discrete HMM. Later,the EBW algorithm was extended to the case of continuous Gaussian densities [63]. The idea was to create a discrete approximation of the Gaussian density so that the EBW algorithm can be applied. More recently, an auxiliary function for discriminative model estimation analogous to the EM auxiliary function was used to derive the EBW style updates [64]. The main advantage of this novel approach is that it does not require discrete density approximations and extends the EBW algorithm to arbitrary continuous density HMMs with arbitrary parameterizations. Finally, the update rules for the means μ_{jm}, the covariance $\mathbf{\Sigma}_{jm}$ and the mixture weight c_{jm} for the gaussian mixture component m associated with the HMM state j are:

$$\hat{\mu}_{jm} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}} \gamma'^{(r)}_{jm}(t;\theta^{(i)})\mathbf{o}_t^{(r)} + D_{jm}\mu_{jm}}{\sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}} \gamma'^{(r)}_{jm}(t;\theta^{(i)}) + D_{jm}} \tag{7.15}$$

$$\hat{\mathbf{\Sigma}}_{jm} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}} \gamma'^{(r)}_{jm}(t;\theta^{(i)})\mathbf{o}_t^{(r)}(\mathbf{o}_t^{(r)})^T + D_{jm}(\mathbf{\Sigma}_{jm} + \mu_{jm}\mu_{jm}^T)}{\sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}} \gamma'^{(r)}_{jm}(t;\theta^{(i)}) + D_{jm}} - \hat{\mu}_{jm}\hat{\mu}_{jm}^T \tag{7.16}$$

$$\hat{c}_{jm} = \frac{\sum_{r=1}^{R} \gamma'^{(r)}_{jm}(t;\theta^{(i)}) + c_{jm}\ C}{\sum_{m'}\sum_{r=1}^{R} \gamma'^{(r)}_{jm'}(t;\theta^{(i)}) + c_{jm'}\ C} \tag{7.17}$$

where, $\gamma'^{(r)}_{jm}(t;\theta^{(i)}) = \gamma^{(r)}_{jm}(t;\theta^{(i)}) - \gamma_{(gen)}{}^{(r)}_{jm}(t;\theta^{(i)})$ and $\gamma^{(r)}_{jm}(t;\theta^{(i)})$ is the probability of occupying the gaussian component m in state j at time t.

7.3 Evaluating Recognition Output

To measure the accuracy of the speech recognizer we compare the true transcription $\bar{\mathbf{W}}$, which is obtained manually from humans, to the output of the recognizer $\hat{\mathbf{W}}$ (referred as hypothesis). This comparison is usually performed on the sentence level using the Sentence Error Rate (SER) metric or on the wold level using the Word Error Rate (WER) metric. The SER is defined as the percentage of correctly transcribed sentences. However, the accuracy of a speech recognizer at the sentence level may not be a good indicator of performance. This is because the SER metric yields the same value without taking into account whether there is one or more incorrectly hypothesized words in a sentence between the true transcription and the

hypothesis. Instead we use WER which is a function of the cost involved to transform the true word string into the hypothesis by using three elementary operations: deletion (DEL), insertion (INS) and substitution (SUB) of a word. We use a dynamic programming procedure [54] to efficiently calculate the number of INS, DEL and SUB between the reference and the hypothesis. The WER is then given by:

$$WER = \frac{INS + DEL + SUB}{N} \times 100\%$$

where, N is the total number of words in the reference. WER is more informative than the SER since it allows to identify specific ASR errors, i.e. the most frequently confusable words.

Chapter 8

Lightly Supervised Discriminative Training

In this chapter, a method for training acoustic models using automatically generated transcripts is presented. The central idea is to automatically transform the non-literal transcripts to verbatim transcripts, then identify reliable regions in these transcripts that can be used for acoustic model (AM) training. Since, discriminative training techniques such as MMI have been shown to outperform ML training in speech recognition tasks [61], the efficacy of using the automatically generated transcripts in MMI training is investigated. More specifically, a lightly supervised approach of frame-based filtering for lattice-based MMI training that takes advantage of the reliability information available at the frame level is discussed.

8.1 Speech Recognition for Document Transcription

It is desirable in many contexts to record human speech in a written document. In general, the term transcription refers to the process of recording speech in a textual document referred to as a transcript of the speech. One example is in the medical profession, where transcripts are produced of diagnoses, prognoses, prescriptions, and other information dictated by doctors and other medical professionals. Transcripts in these and other fields need to be highly accurate (as measured in terms of the degree of correspondence between the original speech and the resulting tran-

script) because of the reliance placed on the resulting transcripts and the harm that could result from an inaccuracy (such as providing an incorrect prescription drug to a patient). Humans are very good at transcribing documents since they may have domain-specific knowledge, such as knowledge of medicine and medical terminology, which enables them to interpret ambiguities in speech and thereby to improve transcript accuracy. Human transcriptionists, however, have a variety of disadvantages. For example, human transcriptionists produce transcripts relatively slowly and are subject to decreasing accuracy over time as a result of fatigue. ASR systems on the other hand can improve this workflow by providing faster transcriptions with less human intervention. However, ASR systems are prone to recognition errors and so it is very important that we train speech recognizers in the domain of interest which are capable of producing highly accurate transcripts.

8.2 Challenges in Acoustic Model Training

Training highly accurate ASR systems requires speech corpora with accurate time-aligned orthographic transcriptions, which we will call *verbatim* transcripts. Furthermore, large amounts of acoustic training data can potentially help reduce recognition errors, by allowing more robust estimation of the ASR system model parameters. However, accurately transcribed training data are not always available. Manually generating transcripts for the vast amounts of raw acoustic data is both a time consuming and prohibitively expensive task and not always a feasible option. On the other hand, there might be some accompanying textual information that loosely corresponds to the speech utterance. We will call such textual information *non-literal* transcripts. For example, in the medical transcription domain there are thousands of hours of medical reports available accompanying the speech data. Human medical transcriptionists listen to dictated recordings made by physicians and other healthcare professionals and transcribe them into medical reports. The final medical reports with the grammatical error corrections, removal of disfluencies and repetitions, addition of non-dictated sentence and paragraph boundaries, rearranged order of dictated paragraphs, formatting and other edits, reflect only partially the speech in the original audio recordings. However, depending on the speaker, the editing of medical reports could account for a significant portion of the speech. Figure 8.1 gives the example of one such medical report. The top block in Figure 8.1

Patient Name	Ms. Jane Doe
Date	07/14/03
Chart No	815D
Synopsis	Ms. Jane Doe today complains of a right watery and itchy eye. Going on for several months now.
PMHx	Significant for hysterectomy, BSO
Meds	Estratest 1 mg q.d.
Allergies	NKDA

next patient is ms jane doe **PAUSE** chart number 815D **PARTIAL – WORD**
ms jane doe comes in today complaining of a right watery itching eye %**PERIOD**%
the patient states that it has been going on for several months now %**PERIOD**%
past medical history is **NOISE** is significant for hysterectomy **PARTIAL – WORD** and bso %**PERIOD**%
medicines are estratest one mg q.day allergies are **FILLED – PAUSE** no known drug allergies %**PERIOD**%

next patient is ms jane doe **PAUSE** chart number eight DEL:nine SUB:none:one five DEL:d
ms jane doe comes in today complaining of a DEL:right DEL:watery itching eye INS:coronary INS:HPI
the patient states that it DEL:has DEL:been INS:nothing going on for several months now %**PERIOD**%
past medical history is **NOISE** is significant for SUB:extremity:hysterectomy and BSO

Figure 8.1: Example of written non-literal transcript (Top) and its corresponding verbatim transcript (Middle) and the Partially Reliable Transcript with annotations (Bottom)

shows a typical non-literal transcript in the medical domain. The middle block in Figure 8.1 shows the verbatim transcript corresponding to the non-literal transcript. The verbatim transcript is generated by humans and corresponds exactly with the spoken utterance and highlights the significant difference in the transcripts that are needed to train a speech recognizer and the available non-literal transcript. Simply training a speech recognizer with these non-literal transcripts would lead to a sub-optimal acoustic model due to the severe mismatch between the written document and the spoken utterance. It would be advantageous, however, to train the acoustic models using such non-literal transcript because of their abundance in domains such as medicine. In the medical domain at least, the non-literal transcripts are easy to obtain as it is a normal by-product of the medical transcription workflow. In order to use the non-literal transcript in training of acoustic models it first needs to be converted into a form closely resembling the verbatim transcript. The bottom block in Figure 8.1 gives an example of the partially reliable transcript that is automatically generated and resembles to some extent the manual verbatim transcript. By partially reliable we mean that the words in the transcript annotated as substitutions (SUB), insertions (INS) and deletions (DEL) are the unreliable segments and the rest are the reliable segments. In this thesis, we will present a novel method for

generating the partially reliable transcript from non-literal transcripts and using the reliability annotations to train acoustic models.

In summary, we need improved techniques for training speech recognition systems and, in particular, improved techniques for training transcription systems based on non-literal transcripts of speech. Since, the non-literal transcripts are still used when training the acoustic model we will call such an approach *lightly supervised training*.

8.2.1 Lightly Supervised Training Approaches

Recently, there has been considerable interest in lightly supervised acoustic model training [65, 66]. A lightly supervised approach to discriminatively train the acoustic models was investigated for broadcast news domain for which only closed caption transcriptions are available [65]. In particular, the authors in [65] used language models biased to the closed caption (CC) transcripts to recognise the audio data, and the recognized transcripts were then used as the training transcriptions for acoustic model training. The biased language model was generated by interpolating the language model built on only the CC data with a general background language model. Two methods for training data selection were investigated. In the first method, the CC transcripts were Viterbi aligned with the recognized transcripts and all segments different from the CC transcript were filtered out. In the second method, data filtering based on confidence measure (CM) was investigated. In this approach, the confidence score of a word was obtained from the word posterior probability in the confusion network (CN) [27]. The sentence confidence (per frame) for each segment was then calculated by averaging the word confidences. Finally, a threshold was set to remove the segments with low sentence confidence. No significant gains were reported when compared to training using the unfiltered training data.

A confusion network based approach in the context of lightly supervised training was also investigated for the broadcast news domain [66]. The closed-captions were aligned, under edit distance, with the word lattices created by the decoding procedure, and if the closed-caption matched one of the paths through the word lattice, then the corresponding speech segment was retained. The basic idea was that even if a path does not correspond to the best hypothesis, if it could survive beam pruning

and at the same time match the closed-caption, then it was still likely to correspond to the correct transcript of the speech. In order to reduce the complexity of exploring all paths in the lattice, confusion networks were used to approximate the hypothesis space. For every position in the aligned closed caption and consensus network, if the posterior probability of the best candidate word in the consensus network was higher than a threshold the word was retained at that position. If the posterior probability was below the threshold, the CN was searched for another candidate in the same position which matched the CC transcript in the same position. If no matches were found the speech segment was discarded.

The approaches mentioned above filter either at the segment level or at the word level. This has the disadvantage of throwing away large amounts of data. We can possibly retain more data by employing filtering at the frame level. We will explore such a frame-level approach for lightly supervised training in our work.

8.3 Automatically Generating Reliable Transcripts

In Section 8.2, the importance of verbatim transcripts to train the acoustic models was discussed. Furthermore, the need to transform the non-literal transcripts into something resembling verbatim transcripts was pointed out. However, these automatically generated verbatim transcripts are likely to contain errors and so we would like to flag portions of the transcripts as reliable. Acoustic models can then be trained only on the reliable portion of the transcripts.Next, the procedure used to transform the non-literal transcripts in to partially reliable transcripts will be described. This can be achieve in two steps. First, the non-literal transcripts are transformed into automatic verbatim transcripts. Next, the automatic verbatim transcripts are converted into partially reliable transcripts by flagging reliable and unreliable portions of the transcript.

8.3.1 Partially Reliable Transcript

Automatically generating training verbatim transcripts from the non-literal transcripts can be seen as an iterative procedure, where at each iteration the currently available best acoustic models are used to generate the automatic verbatim tran-

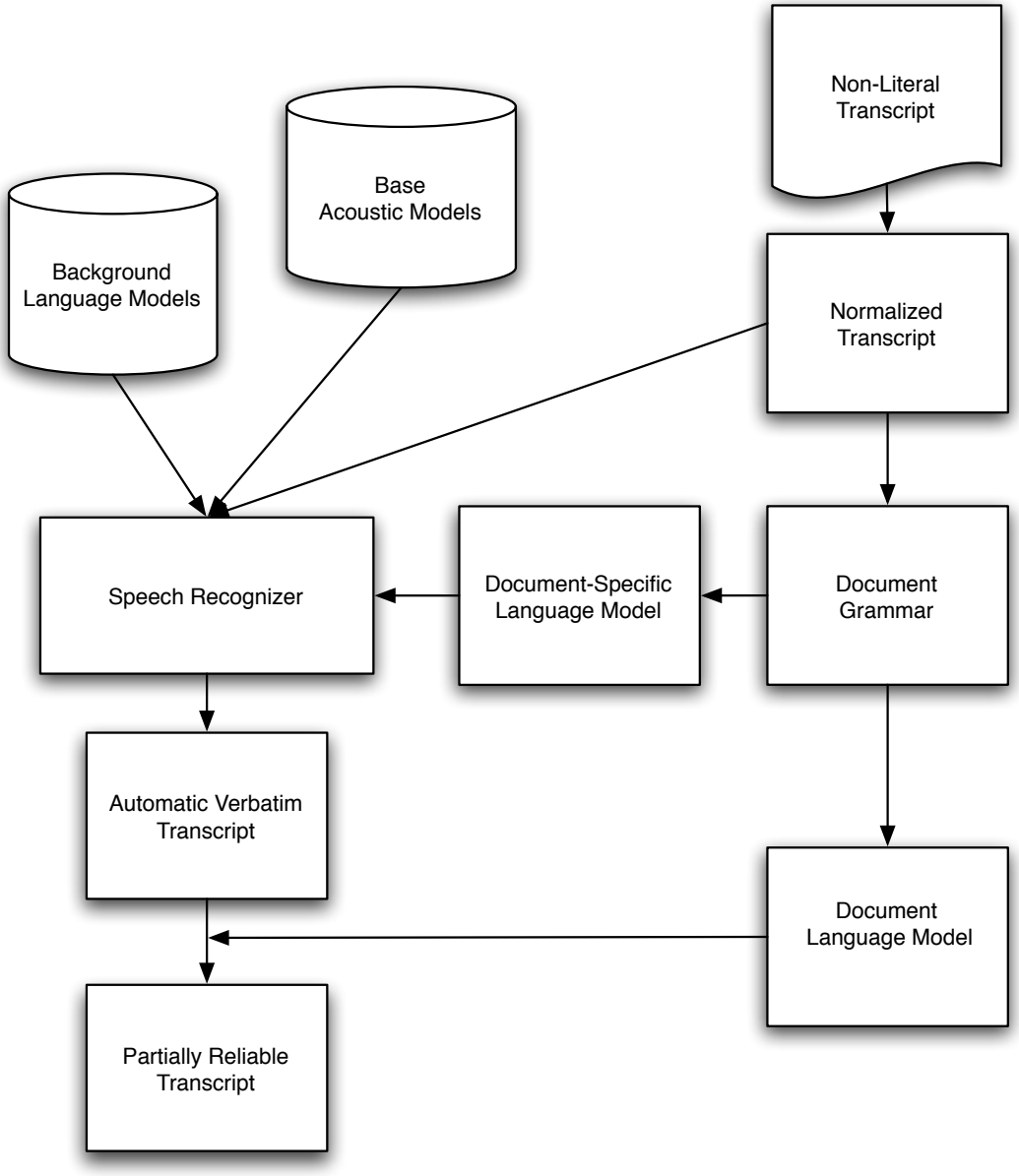

Figure 8.2: Transforming non-literal transcripts to partially reliable transcript

scriptions, given the non-literal transcripts associated with the decoded speech. The reliable segments in the automatic verbatim transcripts are then identified by filtering against the document grammar to generate partially reliable transcripts. The acoustic models are then re-trained on the reliable segments and then used to generate the next set of verbatim transcripts. This procedure is usually run for 2-3 iterations using MLE techniques to train the HMM parameters. This procedure is illustrated in Figure 8.2

The steps to generate partially reliable transcripts are outlined below:

Step 1: Normalization The non-literal transcript is first tokenized. Various techniques are employed in processing the text - normalizing dates (e.g. to mm/dd/yy format) , converting punctuation marks to their spoken form(e.g., changing . to %period%), lowercasing text and and changing blank lines to the text %new-paragraph%.

Step 2: Document Grammar Generation Various concepts in the normalized text are then identified. The concepts can either be low level concepts (such as dates, numbers, currency, etc.) or higher level concepts (sentence boundaries, paragraph boudaries, etc.). The identified concepts are then replaced with appropriate finite state grammars encoding the plurality of such concepts. For example, the concept "october 3 1979" is identified it is replaced by a finite state grammar containing the original form and also forms such as 10/03/1979 and 03/10/1979 and "3rd october 1979". The final document specific grammar contains both flat text and finite state grammars (replacing identified concepts) and is called the normalized grammar transcript.

Step 4: Document Language Model Generation Word counts are accumulated for each normalized grammar transcript. These word counts are then used to a n-gram language model [36].

Step 3: Automatic Verbatim Transcript Generation The document language model along with the best available background language model and base acoustic models is then used to recognize the training speech utterance to obtain an automatic verbatim transcript. Note that it is not necessary to train the initial acoustic models on domain dependent data. For example, the initial models can be borrowed from English Broadcast news transcription systems.

Step 4: Partially Reliable Transcript (PRT) Generation The grammar transcripts are then used to flag the non-matching segments of the automatic verbatim transcripts as unreliable. This transformed transcript with reliability annotations is called the *partially reliable transcript* (PRT).

Step 5: Re-train the Acoustic Models This procedure can be iterated by re-aligning the partially reliable transcripts against the acoustic models, retraining the HMMs using MLE techniques and then re-decoding the normalized transcripts.

8.3.2 Identifying reliable segments in Partially Reliable Transcripts

A central component in the procedure described in Section 8.3.1, is the generation of the PRT. This process is described in more detail in this section. Decoder search errors, lack of coverage of the background LM and inferior acoustic models all contribute to the errors in the automatic verbatim transcript. As a result, it is not possible to use the entire automatic verbatim transcript for re-training the acoustic models without severely degrading the acoustic model parameter estimates. This is especially of concern during discriminative training using the MMIE procedure. MMIE updates the model parameters so as to increase the likelihood of the truth and decrease the likelihood of the competing classes. As a result MMIE is more sensitive to transcription errors than MLE. Hence it is important to identify segments in the transcripts that can be used for training. Such segments are marked as "reliable" and the rest are marked as "unreliable" Given the automatic verbatim transcript, the annotated partially reliable transcripts are generated as follows:

1. The automatic verbatim transcript is first aligned with the document grammar transcript under the edit distance measure which allows for insertion (INS), deletion (DEL) and substitution (SUB) [29]. Next, the best alignment under the edit distance is used to annotate the words with their appropriate symbols (INS, DEL, SUB or MATCH).

2. A Viterbi alignment procedure is used to align the phone based network of HMM acoustic models and the training utterance corresponding to the annotated PRT. As a result of this process, a mapping of the acoustic frames to

the underlying phone HMM is obtained. If a word in the PRT is annotated as INS,DEL or SUB then the acoustic frames associated with the underlying phone HMM are marked as "unreliable". Also, taking into account cross-word context effects the frames associated with the immediate preceding and succeeding phone are also marked as "unreliable". All the other frames are marked as "reliable".

Finally, at the the end of this process acoustic frame level reliability markers corresponding to the unreliable regions in the corresponding word transcript (the PRT) are obtained.

8.4 MMIE with Frame Filtering

In Section 7.2.2, the general MMIE procedure was discussed and update equations for the various HMM parameters were provided. Also, the relative merits of MMIE over MLE were also discussed. Since MMI training is more sensitive to transcription errors than ML training, the entire partially reliable transcript cannot be used for training. Hence, the MMI training framework has to be slightly modified to exploit the reliable/unreliable frame-based annotations. Similar to the approach in [67], we introduce the lattice based MMIE procedure with frame filtering. The numerator statistics for state j and mixture component m are are

$$\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{jm}^{(r)}(t;\theta^{(i)})\mathbf{o}_t^{(r)} \tag{8.1}$$

$$\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{jm}^{(r)}(t;\theta^{(i)})\mathbf{o}_t^{(r)}\mathbf{o}_t^{(r)^T} \tag{8.2}$$

and the denominator statistics for state j and mixture component m are

$$\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{(gen)}{}_{jm}^{(r)}(t;\theta^{(i)})\mathbf{o}_t^{(r)} \tag{8.3}$$

$$\sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma_{(gen)}{}_{jm}^{(r)}(t;\theta^{(i)})\mathbf{o}_t^{(r)}\mathbf{o}_t^{(r)^T} \tag{8.4}$$

The following procedure is then implemented:

1. Initialize accumulators to zero for all parameters of all the HMMs in our inventory.

2. Get the next training utterance $\mathbf{O^{(r)}}$

3. Generate a pair of *numerator* and *denominator* lattices for each utterance in the training data, these correspond to $\mathcal{M}_{\bar{\mathbf{W}}^{(\mathbf{r})}}$ and $\mathcal{M}_{\mathbf{gen}}$, respectively. The numerator lattice is produced by aligning the acoustic data against a network of HMMs built according to the known transcription. The denominator lattice corresponds to running an unconstrained recognition pass. In both cases an appropriate N-gram language model is used. The denominator lattice is generated only once for each training utterance. All subsequent passes use the already generated denominator lattice.

4. Use the forward-backward procedure to calculate the state occupancy probabilities $\gamma_{jm}^{(r)}(t;\theta^{(i)})$ and $\gamma_{(gen)}{}_{jm}^{(r)}(t;\theta^{(i)})$

5. Accumulate the counts for the numerator and denominator statistics shown in Equations 8.1, 8.2, 8.3 and 8.4. At this point we introduce the modification based on the frame reliability markers. We update the counts of the numerator/denominator statistics only if the observation frame $\mathbf{o}_t^{(r)}$ has been flagged as reliable. Else, we do not increment the counts.

6. Repeat from Step 2 until all the training utterances have been processed.

7. Use the update equations defined in Section 7.2.2 to update the HMM parameters.

These steps, except for Step 3 are repeated several times until convergence.

8.5 Summary

In this chapter, we motivated the need for training speech recognition systems with only non-literal transcripts starting from an initial set of acoustic models. Medical transcriptioning was cited as one such domain in which we have thousands of hours of acoustic data with corresponding medical reports in written form.

A novel iterative procedure for transforming the non-literal transcripts to partially reliable transcripts was presented. Furthermore, a procedure was described to mark the words in the partially reliable transcripts as insertions, deletions, substitutions or matches by aligning this against the non-literal transcript. All words which are matches were identified as reliable and the rest were marked as unreliable. A new frame filtering approach to lattice-based MMI training was investigated. The partially reliable transcripts were Viterbi aligned with the underlying acoustic models to obtain reliability markers on the HMM phone models underlying the aligned acoustic frames. The MMI update procedure was modified to update counts only for the reliable acoustic frame segments.

Chapter 9

Experiments with Frame Filtering based MMI Training

In this chapter, we present experiment results with lightly supervised discrminiative training for estimating the parameters of the HMM. The domain of application is the radiology domain, although, the techniques described can easily be applied to other domains. Latice-based MMIE with frame filtering is used to train the system. The aim of the experiments is to evaluate whether speaker-dependent MMI (SD-MMI) trained models can outperform speaker-dependent ML (SD-ML) training when trained on the automatically generated transcripts. Also, of interest is how much data is needed to obtain performance improvements from MMI with frame filtering. The evaluation is done using two different trigram LMs, the speaker-independent language models (SI-LMs) and speaker-dependent language models (SD-LMs).

9.1 Experimental Setup

The radiology training corpus used in this study consists of about 180 hours of recorded speech across thirty-one speakers (both male and female). The data was recorded at $11kHz$ using desktop microphones [68]. The speech data is segmented so that for each formatted medical report (non-literal transcript) there is an associated speech utterance. The utterances are typically about two to twenty minutes in length. For the experiments we considered a sample of four male speakers to build our speaker dependent models. All the speakers were native English speakers.

Approximately eight to twelve hours of acoustic training data was available for each of the four speakers to build the speaker-dependent acoustic models. In addition, we built speaker dependent language models from the training text corresponding to the four male speakers. All the evaluations were performed on an independent test set consisting of 20 utterances per speaker. Human generated verbatim transcriptions were available for each of these test utterances. Finally, WER was used to evaluate the recognizer performance.

9.2 Acoustic Model Training

First, ML estimation was used to train a speaker-independent (SI) acoustic models on only the reliable segments of the PRT generated from the 180 hours of training data. This system had 6% lower WER than an ML system trained on the entire PRT (both reliable and unreliable segments). The final SI system consisted of about 2000 context-dependent mixture models and 24 Gaussians per mixture. The vocabulary size was approximately $26K$. Next, the SI models were adapted using only the PRT from the additional 8-12 hours of training data for each of the four randomly chosen speakers to build speaker-dependent (SD-ML) models. These final SD-ML models for each of the four speakers were chosen to be our baseline. Background trigram and unigram speaker-independent language models (SI-LMs) were trained on all the training text available for the 31 speakers. In addition, speaker-dependent language models (SD-LMs) were generated for each of the four speakers by interpolating the SI-LM with a speaker-specific SD-LM trained on the corresponding speaker's available training text.

For discriminative training the procedure described in Section 8.4 was followed. The SD-ML acoustic models for each of the speakers and the trigram SI-LM were used to decode the training set and generate word level phone boundary marked numerator and denominator lattices. To improve the generalization of the SD-MMI models, the lattice trigram SI-LM scores were replaced by unigram SI-LM scores and acoustic scaling was also employed to get a broader posterior probability distribution [67]. Prior to the model parameter reestimation step, counts were gathered only for the reliable frames. The unreliable frames were skipped during the count accumulation. Typically, 1 to 2 iterations of MMI were performed to get the final

SD-MMI acoustic models.

9.3 Speech Recognition Results

Speaker	SD-ML WER	SD-MMI WER	Reliable acoustic data (hours)	%reliable frames
Speaker 1	12.3	10.4	13hrs	69%
Speaker 2	14.5	12.8	7.2hrs	52%
Speaker 3	8.6	7.5	7.5hrs	77%
Speaker 4	12.2	11.3	7.1hrs	76%

Table 9.1: WER for SD-ML and SD-MMI acoustic models using speaker independent language model

The first task was to compare the WER of the SD-MMI models with the baseline SD-ML models on an independent test using the SI-LMs. Columns 2 and 3 in Table 9.3 give the WER for each speaker for the SD-ML and SD-MMI models respectively. As expected, the SD-MMI trained models outperformed the SD-ML trained models. A significant reduction in WER was achieved ($8 - 15\%$ relative gains) for each of the speakers. Also, since we only trained on the reliable frames as annotated under the PRT we were interested in quantifying the percentage of reliable frames used in training the SD-MMI models. Columns 4 and 5 in Table 9.1 give the number of absolute hours of reliable training data and also the amount of training data as a percentage of the total available data. It is interesting that for Speaker 2, although only 52% of the reliable frames were retained, a 11% relative improvement in the WER was observed.

In theory, we would like to retain as many reliable acoustic frames as possible to obtain good gains from MMIE. However, it is a time consuming task to generate the reliability markers for all the available speaker specific data and so we would like to measure the amount of reliable training data frames needed (in terms of hours) in order to obtain gains from MMIE. In Figure 9.1 the WER was plotted as a function of the amount of reliable training data available. The reliable training data is the portion of the training data that has been marked as reliable using the procedure described in Section 8.3.1. The $x - axis$ represents the number of hours of reliable training data required for training the SD-MMI models. 0 $hours$ represents the

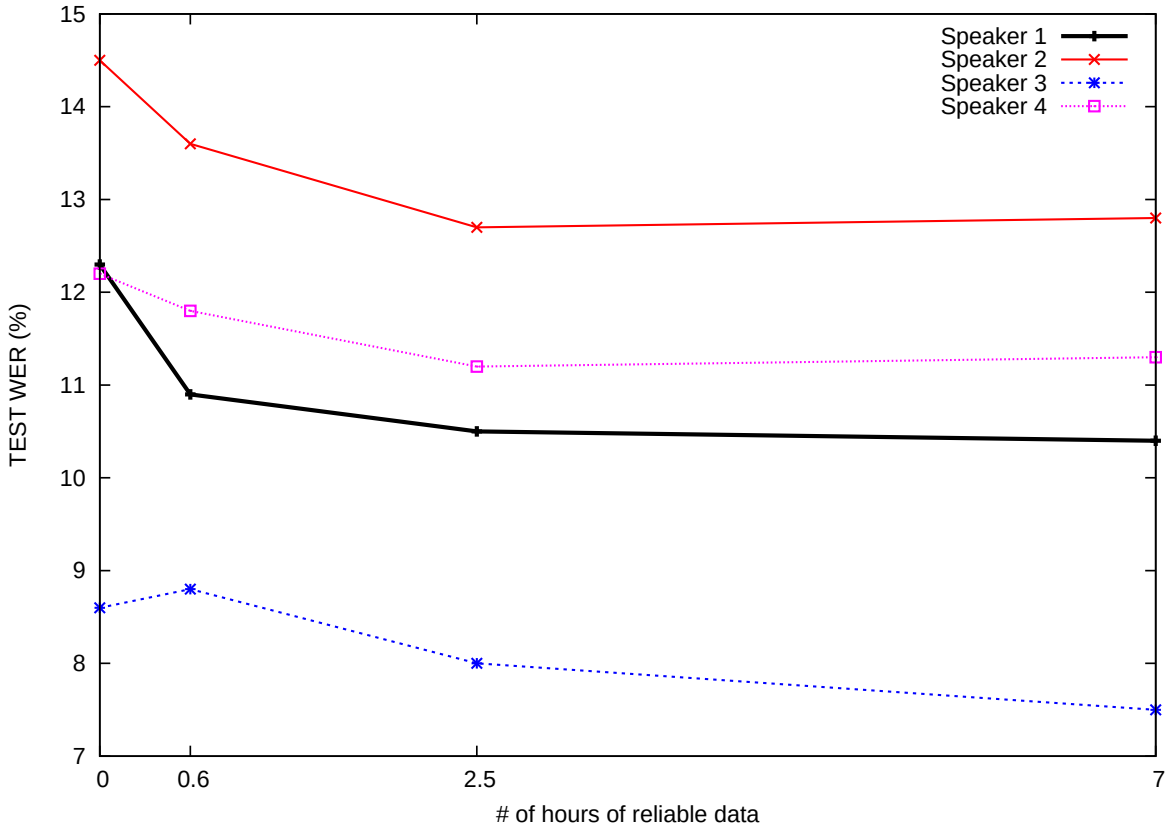

Figure 9.1: WER as a function of the amount of reliable training data for SD-MMI models using speaker independent language models

baseline SD-ML model. From the plots, it was observed that with approximately two hours of reliable training data per speaker we were able to achieve $6 - 14\%$ relative drop in WER when compared to the baseline SD-ML models.

Finally, we evaluated the system performance by replacing the speaker-independent language models with the speaker-dependent language models. The speaker-dependent language models were trained on the speaker specific text and then interpolated with the speaker-independent language models. We tabulated the results of MMIE with the speaker-dependent language models in Table 9.2. Comparing column 1 in Table 9.1 and Table 9.2 respectively we found that simply changing the language model from speaker-independent to speaker-dependent resulted in a drop in WER for all the speakers. This suggests that the speaker-dependent language models are better able to capture speaker specific contextual information than the speaker-independent language models as they are trained on speaker-specific text. Furthermore, applying MMIE for this task gave us a maximum of 17% relative reduction in WER.

Speaker	SD-ML WER	SD-MMI WER
Speaker 1	11.7	9.6
Speaker 2	14.4	13.0
Speaker 3	7.9	6.8
Speaker 4	11.8	11.5

Table 9.2: WER for SD-ML and SD-MMI models using speaker-dependent language models

9.4 Summary

In this section, we investigated the use of automatically generated transcripts for MMI training of speaker-adapted acoustic models. Experimental results showed that the MMI training with frame filtering was effective in reducing the WER by as much as 15% relative to the baseline ML trained models. Furthermore, the gains from MMI carried over when we use a speaker-specific LM for decoding. The main advantage of this approach is that it does not need any kind of transcripts to seed the initial acoustic model training. The techniques can be easily extended to other domains where there are no verbatim transcripts available to train from but there is some form of non-literal transcripts available (for example, in broadcast news where closed captioning is made available).

Part III

Conclusions and Future Work

Chapter 10

Conclusion

In this Chapter, a summary of the work presented in this thesis and the contributions are highlighted. and possible directions of future work are outlined.

10.1 Thesis Summary

The research objective of this thesis was to formulate methodologies to handle uncertainty in natural language processing tasks. The focus was on two specific areas - Statistical Speech Translation and Statistical Speech Recognition.

In the integrated approach to speech translation, the coupling of the ASR component with the SMT component is achieved via an ASR word lattice which represents a large hypothesis space of possible translation candidates to be used by the SMT component. The ASR lattice encodes the uncertainty in the sense that no *a priori* decision is made about the optimal ASR hypothesis to be used for transaltion. A joint search is conducted for the optimal translation candidate and its corresponding translation in the source language. In Chapter 2, we formulated a novel joint source-channel model for the translation of speech was formulated as a direct extension of the text translation system. The speech is translated via a series of transformations at the word and phrase level; conditional distributions underlying the generative model describe the process of transforming a source language string into a target language speech utterance. We used WFSTs to model the conditional distributions and discussed how translation can be implemented using composition and best path operations under this framework. We presented an efficient strategy for extracting

phrases from a word lattice. In addition, we introduced a novel phrase based pruning strategy to retain only high confidence phrases (high posterior probability under the target acoustic model and language model probability) during translation. In Chapter 6, translating from lattices was shown to significantly outperform translating the ASR 1-best.

In Chapter 4, we studied discriminative training of the SMT parameters in detail. Three different objective functions for training the SMT parameters were investigated - BLEU, expected BLEU and MMI. Furthermore, we introduced a novel growth transformation based update procedure to discriminatively train the SMT parameters. We also derived growth transformation based updates for the MMI and expected BLEU training objective functions. Furthermore, we investigated the problem of training with multiple references and proposed two different MMI training criteria for the task. In Chapter 5, growth transformation based discriminative training was shown to be comparable to the standard minimum error training approach.

In Chapter 8, we studied the problem of automatically generating transcripts from non-reliable transcripts. We also presented a lightly supervised discriminative training approach to train the acoustic models from these automatically generated transcripts. Specifically, a novel procedure for identifying reliable acoustic frames was presented. Also, a modified MMI procedure was used to update the acoustic model parameters using a frame based filtering strategy: the parameter counts in MMI were accumulated only over the reliable portions of the acoustic frame. Finally, recognition experiments in Chapter 9, showed significant gains from applying this approach.

10.2 Future Work

10.2.1 Speech Translation

The generative model formulated for speech translation is a relatively simple and straightforward extension of the text translation system. The ASR component forms one of the components in the translation pipeline. Instead of using the ASR lattice,

we could explore the use of word confusion networks as input to the decoder [26]. Confusion networks [69, 27] are a compacted representation of word lattices that have strictly ordered word hypothesis slots, and each word as an associated posterior probability. In addition to retaining the paths from the original lattice, confusion networks also introduce new paths. As a result phrase extraction from confusion network can result in a richer phrase pair inventory than the original word lattice. Confusion networks can be obtained by using lattice cutting procedures [70], to segment the lattice into sub-lattices and then concatenating these sub-lattices. Since, we are interested in translating phrases, another extension would be to build confusion networks at the phrase level (each arc in each segment corresponds to a sequence of words).

10.2.2 Discriminative Training for Machine Translation

Growth transformations were presented as the optimization procedure for MMI and Expected BLEU training. Growth transformation are an attractive optimization procedure as it neatly sidesteps the issue of choosing an appropriate step size in the other gradient based methods (for example, gradient descent). Furthermore, convergence is observed in practice within a few iterations. One direction for future research we would like to explore is optimizing for a significantly larger feature set. Another extension, is extending the discriminative training approach from N-best lists to lattices. Appendix A, outlines a procedure for lattice based discriminative training under the MMI training criterion.

10.2.3 Lightly supervised training for speech recognition

We presented a novel frame filtering based approach to selectively train the acoustic models in a discriminative fashion. However, in the current implementation, during MMI training we simply skip the count accumulation for the unreliable frames. Instead of simply skipping the unreliable frames, one possibility is to explore data re-weighting. For each frame we can associate a weight that reflect the reliability of that frame. During the count accumulation the MMI training algorithm simply puts more weight on the reliable portions while still using some of the acoustic evidence in the unreliable portions. In generating the partially reliable training transcripts, only the 1-best output of the ASR recognition pass is aligned with the document gram-

mar. Instead we can align confusion networks with the document grammar. The intuition is that paths other than the ASR 1-best might align better with the document grammar under the edit distance metric, thereby resulting in more accurate training transcripts.

Appendix A

Lattice MMI Training under the TTM

A.1 Growth Transforms

The goal is to obtain parameter updates for MMI training over lattices. We will use the growth transformation based update procedure described in Chapter 4 to calculate the updates. Furthermore, we will show how the objective function and derivative calculations can be implemented efficiently using WFSTs.

We begin with the TTM [3] joint distribution defined as

$$p_\theta(\mathbf{f}_s, \mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s, \mathbf{e}_s) \;=\; P(\mathbf{f}_s|\mathbf{v}_s)\; P(\mathbf{v}_s|\mathbf{y}_s)^{\theta_1}\; P(\mathbf{y}_s|\mathbf{x}_s)^{\theta_2}\; P(\mathbf{x}_s|\mathbf{u}_s)^{\theta_3}\; P(\mathbf{u}_s|\mathbf{e}_s)\; P(\mathbf{e}_s)^{\theta_4} \tag{A.1}$$

where,

- $\mathbf{f}_s$ is the target sentence in target language order
- $\mathbf{e}_s$ is the source sentence in source language order
- $\mathbf{v}_s$ is the target phrase in target language order after insertions
- $\mathbf{y}_s$ is the target phrase in target language order after reordering $\mathbf{x}_s$
- $\mathbf{x}_s$ is the target phrase in source language order
- $\mathbf{u}_s$ is the souce phrase in source language order

Also,

$$\sum_{q=1}^{4} \theta_q = 1,\ \theta_q \geq 0$$

For simplicity let us assume for each sentence $\mathbf{f}_s$ to be translated there is a single reference translation $\mathbf{e}_s^+$ available. Let the parameter set we are interested in estimating be defined as $\theta = \{\theta_1, \theta_2, \theta_3, \theta_4\}$. We can then define the MMI training criteria as:

$$\mathcal{F}(\theta) = \sum_{s=1}^{S} \log \frac{\sum_{\mathbf{v}_s,\mathbf{y}_s,\mathbf{x}_s,\mathbf{u}_s} p_\theta(\mathbf{f}_s, \mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s, \mathbf{e}_s^+)}{\sum_{\mathbf{e}'_s,\mathbf{v}'_s,\mathbf{y}'_s,\mathbf{x}'_s,\mathbf{u}'_s} p_\theta(\mathbf{f}'_s, \mathbf{v}'_s, \mathbf{y}'_s, \mathbf{x}'_s, \mathbf{u}'_s, \mathbf{e}'_s)} \tag{A.2}$$

This objective function is differentiable and so we can apply the growth transform update procedure to obtain the parameter updates:

$$\hat{\theta}_i = \frac{\theta_i \left(\nabla_\theta \mathcal{F}(\theta) \Big|_{\theta=\theta_i} + C \right)}{\sum_{j=1}^{4} \theta_j \left(\nabla_\theta \mathcal{F}(\theta) \Big|_{\theta=\theta_j} + C \right)}, \quad 1 \leq i \leq 4 \tag{A.3}$$

We will now derive the derivative for each of the component model parameters

Phrase Transduction Model

$$\begin{aligned} \nabla_\theta \mathcal{F}(\theta) \Big|_{\theta=\theta_3} = \sum_{s=1}^{S} \Bigg[& \sum_{\mathbf{v}_s,\mathbf{y}_s,\mathbf{x}_s,\mathbf{u}_s} \frac{p_\theta(\mathbf{f}_s, \mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s, \mathbf{e}_s^+)}{\sum_{\mathbf{v}_s,\mathbf{y}_s,\mathbf{x}_s,\mathbf{u}_s} p_\theta(\mathbf{f}_s, \mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s, \mathbf{e}_s^+)} \log p(\mathbf{x}_s|\mathbf{u}_s) \\ & - \sum_{\mathbf{v}_s,\mathbf{y}_s,\mathbf{x}_s,\mathbf{u}_s,\mathbf{e}_s} \frac{p_\theta(\mathbf{f}_s, \mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s, \mathbf{e}_s)}{\sum_{\mathbf{v}_s,\mathbf{y}_s,\mathbf{x}_s,\mathbf{u}_s,\mathbf{e}_s} p_\theta(\mathbf{f}_s, \mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s, \mathbf{e}_s)} \log p(\mathbf{x}_s|\mathbf{u}_s) \Bigg] \end{aligned} \tag{A.4}$$

$$= \sum_{s=1}^{S} \left[\sum_{\mathbf{h}_s} p_\theta(\mathbf{h}_s|\mathbf{f}_s, \mathbf{e}_s^+) \log p(\mathbf{x}_s|\mathbf{u}_s) - \sum_{\mathbf{e}_s,\mathbf{h}_s} p_\theta(\mathbf{e}_s, \mathbf{h}_s|\mathbf{f}_s) \log p(\mathbf{x}_s|\mathbf{u}_s) \right] \tag{A.5}$$

where, $\mathbf{h}_s = \{\mathbf{v}_s, \mathbf{y}_s, \mathbf{x}_s, \mathbf{u}_s\}$ is the set of hidden variables. Observe that

$$\log p(\mathbf{x}_s|\mathbf{u}_s) = \sum_{x,u} \#_{x,u}(\mathbf{x}_s, \mathbf{u}_s) \log p(x|u)$$

where, $\#_{x,u}(\mathbf{x}_s, \mathbf{u}_s)$ is defined as the number of times the pair (x, u) was observed in the sequence pair $(\mathbf{x}_s, \mathbf{u}_s)$.

$$\nabla_\theta \mathcal{F}(\theta)\Big|_{\theta=\theta_3} = \sum_{s=1}^{S}\Bigg[\sum_{x,u}\log p(x|u)\Bigg(\sum_{\mathbf{h}_s} p_\theta(\mathbf{h}_s|\mathbf{f}_s,\mathbf{e}_s^+)\#_{x,u}(\mathbf{x}_s,\mathbf{u}_s)\Bigg) - \sum_{x,u}\log p(x|u)\Bigg(\sum_{\mathbf{h}_s,\mathbf{e}_s} p_\theta(\mathbf{e}_s,\mathbf{h}_s|\mathbf{f}_s)\#_{x,u}(\mathbf{x}_s,\mathbf{u}_s)\Bigg)\Bigg] \tag{A.6}$$

Similarly derivatives can be calculated for the other components

Target Phrase Insertion Model

$$\nabla_\theta \mathcal{F}(\theta)\Big|_{\theta=\theta_1} = \sum_{s=1}^{S}\Bigg[\sum_{v,y}\log p(v|y)\Bigg(\sum_{\mathbf{h}_s} p_\theta(\mathbf{h}_s|\mathbf{f}_s,\mathbf{e}_s^+)\#_{v,y}(\mathbf{v}_s,\mathbf{y}_s)\Bigg) - \sum_{v,y}\log p(v|y)\Bigg(\sum_{\mathbf{h}_s,\mathbf{e}_s} p_\theta(\mathbf{e}_s,\mathbf{h}_s|\mathbf{f}_s)\#_{v,y}(\mathbf{v}_s,\mathbf{y}_s)\Bigg)\Bigg] \tag{A.7}$$

Target Phrase Reordering Model

$$\nabla_\theta \mathcal{F}(\theta)\Big|_{\theta=\theta_2} = \sum_{s=1}^{S}\Bigg[\sum_{y,x}\log p(y|x)\Bigg(\sum_{\mathbf{h}_s} p_\theta(\mathbf{h}_s|\mathbf{f}_s,\mathbf{e}_s^+)\#_{y,x}(\mathbf{y}_s|\mathbf{x}_s)\Bigg) - \sum_{y,x}\log p(y|x)\Bigg(\sum_{\mathbf{h}_s,\mathbf{e}_s} p_\theta(\mathbf{e}_s,\mathbf{h}_s|\mathbf{f}_s)\#_{y,x}(\mathbf{y}_s|\mathbf{x}_s)\Bigg)\Bigg] \tag{A.8}$$

Source Phase Language Model

$$\nabla_\theta \mathcal{F}(\theta)\Big|_{\theta=\theta_4} = \sum_{s=1}^{S}\Bigg[\sum_{h_e,e}\log p(e|h_e)\Bigg(\sum_{\mathbf{h}_s} p_\theta(\mathbf{h}_s|\mathbf{f}_s,\mathbf{e}_s^+)\#_{h_e,e}(\mathbf{e}_s^+)\Bigg) - \sum_{h_e,e}\log p(e|h_e)\Bigg(\sum_{\mathbf{h}_s,\mathbf{e}_s} p_\theta(\mathbf{e}_s,\mathbf{h}_s|\mathbf{f}_s)\#_{h_e,e}(\mathbf{e}_s)\Bigg)\Bigg] \tag{A.9}$$

where, h_e is the history for the word e. For example, if we have a trigram language model with probability $p(e|e',e'')$ then the history is $h_e = (e',e'')$.

A.2 WFST Implementation

We will now show how the gradient and the objective function can be calculated using WFST operations similar to the work in [71]. Following the discussion in

Section 2.6, we will define some additional WFST/WFSA operations over the log semiring. We assume without loss of generality that the weights $w[\pi]$ are negative log probabilities.

1. $Encode(T)$

   This operation converts a WFST T to its equivalent WFSA A by encoding the input output label pair as a single string. For a WFST T, this operation yields a WFSA A such that

   $$\Pi_A = \{\pi_A \big| \forall \pi_T \in \Pi_T, \exists\ w[\pi_A] = w[\pi_T] \wedge\ l[e_A] = i[e_T] \cdot o[e_T] \forall e_T \in \pi_T\}$$

2. $PathSum(A)$ Given a WFA A, this yields the total log probability of the sum of all paths in the lattice.

   $$PathSum(A) = \log\left(\sum_{\pi \in \Pi_A} \exp(w[\pi])\right)$$

3. $Norm(A)$

   Given a WFA A, this operation yields a WFA B such that $L_A = L_B$ and

   $$\Pi_B = \left\{\pi_B \big|\ \forall \pi_A \in \Pi_A,\ \exists\ l[\pi_A] = l[\pi_B] \wedge\ w[\pi_B] = w[\pi_A] - PathSum(A)\right\}$$

4. $ExpCount(A, s_1^k)$

   Given a WFA A and the string s_1^k, the expected count of w_1^k in A is defined as

   $$ExpCount(A, s_1^k) = \sum_{\pi \in \Pi_A} C(s_1^k|\pi)\ [[Norm(A)]](s_1^k)$$

   where, $C(s_1^k|\pi)$ is the frequency of occurrence of s_1^k in π.

In Equation A.2, let us represent the joint distribution in the numerator by $\mathcal{N}_s$ and the joint distribution in the denominator by $\mathcal{D}_s$. Then the objective function is simply

$$\mathcal{F}(\theta) = \sum_{s=1}^{S} \log(PathSum(\mathcal{N}_s)) - \log(PathSum(\mathcal{D}_s)) \tag{A.10}$$

The derivative for the phrase transduction model can be calculated as:

$$\begin{aligned}\nabla_\theta \mathcal{F}(\theta)\Big|_{\theta=\theta_3} &= \sum_{s=1}^{S} \sum_{x,u} \log p(x|u) ExpCount(Norm(N_s, xu)) \\ &\quad - \sum_{s=1}^{S} \sum_{x,u} \log p(x|u) ExpCount(Norm(D_s, xu))\end{aligned} \tag{A.11}$$

We can calculate the other derivatives in a similar fashion.

Bibliography

[1] F. Jelinek, "Continuous speech recognition by statisical methods," in *IEEE Proceedings*, 1976, vol. 64.

[2] L.R. Bahl, F. Jelinek, and R.L. Mercer, "A maximum likelihood approach to continuous speech recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 1983, IEEE.

[3] S. Kumar, Y. Deng, and W. Byrne, "A weighted finite state transducer translation template model for statistical machine translation," *Journal of Natural Language Engineering*, vol. 12, no. 1, pp. 35–75, 2006.

[4] K. Knight and D. Marcu, "Machine translation in the year 2004," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 2005.

[5] P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer, "The mathematics of machine translation: Parameter estimation," *Computational Linguistics*, vol. 19, pp. 263–312, 1993.

[6] U. Germann, M. Jahr, K. Knight, D. Marcu, and K. Yamada, "Fast decoding and optimal decoding for machine translation.," in *Association for Computational Linguistics*, 2001.

[7] P. Koehn, F. Och, and D. Marcu, "Statistical phrase-based translation.," in *Proceedings of the Human Language Technology Conference of the NAACL*, 2003.

[8] F. Och, C. Tillmann, and H. Ney, "Improved alignment models for statistical machine translation," in *Proceedings of the Joint Conf. of Empirical Methods in Natural Language Processing and Very Large Corpora*, College Park, MD, USA, 1999, pp. 20–28.

[9] M. Mohri, F. Pereira, and M. Riley, "Weighted automata in text and speech processing," in *European Conference on Artificial Intelligence*, 1996.

[10] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "BLEU: a method for automatic evaluation of machine translation," Tech. Rep. RC22176(W0109-022), IBM Research, 2001.

[11] G. Doddington, "Automatic evaluation of machine translation quality using ngram co-occurrence statistics," in *Proceedings of HLT*, 2002.

[12] I. D. Melamed, R. Green, and J. P. Turian, "Precision and recall of machine translation," in *Proceedings of the Human Language Technology Conference of the NAACL*, 2003.

[13] F. Och, *Statistical Machine Translation: From Single Word Models to Alignment Templates*, Ph.D. thesis, RWTH Aachen, Germany, 2002.

[14] Mathew Snover, Bonnie Dorr, Richard Schwartz, John Makhoul, Linnea Micciula, and Ralph Weischedel, "Study of translation error rate with targeted human annotation," in *Machine Translation Workshop*, 2005.

[15] NIST, "Nist machine translation evaluations," .

[16] H. Ney, "Speech translation: Coupling of recognition and translation," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 1999.

[17] Alon Lavie, Alex Waibel, Lori Levin, Michael Finke, Donna Gates, Marsal Gavaldà, Torsten Zeppenfeld, and Puming Zhan, "Janus-iii: Speech-to-speech translation in multiple languages," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 1997.

[18] Alan W Black, Ralf D. Brown, Robert Frederking, Kevin Lenzo, John Moody, and Eric Steinbrecher Alexander Rudnicky, Rita Singh, "Rapid development of speech to speech translation systems," in *Proceedings International Conference on Speech and Language Processing*, 2002.

[19] Fu Hua Liu, Yuqing Gao, Liang Gu, and Michael Picheny, "Noise robustness in speech to speech translation," in *IBM Tech. Report RC22874*, 2003.

[20] E. Vidal, "Finite-state speech-to-speech translation," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 1997.

[21] F. Casacuberta, E. Vidal, and J. Vilar, "Architectures for speech-to-speech translation using finite-state models," in *Proceedings Workshop on Speech-to-Speech Translation*, 2002.

[22] Ruiqiang Zhang, Genichiro Kikui, Hirofumi Yamamoto, Taro Watanabe, Frank Soong, and Wai Kit Lo, "A unified approach in speech-to-speech translation: integrating features of speech recognition and machine translation," in *Proceedings COLING*, 2004.

[23] V.H. Quan, M. Federico, and M.Cettolo, "Integrated N-best re-ranking for spoken language translation," in *In EuroSpeech*, 2005.

[24] S.Saleem, S. C. Jou, S. Vogel, and T. Schultz, "Using word lattice information for a tighter coupling in speech translation systems," in *Proceedings International Conference on Speech and Language Processing*, 2004.

[25] E. Matusov, S.Kanthak, and H. Ney, "On the integration of speech recognition and statistical machine translation," in *Proceedings InterSpeech*, 2005.

[26] N. Bertoldi and M. Federico, "A new decoder for spoken language translation based on confusion networks," in *IEEE ASRU Workshop*, 2005.

[27] E. Brill L. Mangu and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," *Computer, Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[28] M. Mohri, "Finite state transducers in language and speech processing," *Computational Linguistics*, vol. 23, no. 2, pp. 269–311, 1997.

[29] M. Mohri, F. Pereira, and M. Riley, "Weighted finite state transducers in speech processing," *Computer Speech and Language*, vol. 16, no. 1, pp. 69–88, 2002.

[30] Srinivas Bangalore and Giuseppe Riccardi, "Finite-state models for lexical reordering in spoken language translation," in *Proceedings International Conference on Speech and Language Processing*, 2000, vol. 4, pp. 422–425.

[31] S. Vogel, H. Ney, and C. Tillmann, "HMM based word alignment in statistical translation," in *Proceedings of the COLING*, 1996.

[32] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Computational Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[33] Y. Deng and W. Byrne, "HMM word and phrase alignment for statistical machine translation," in *Proceedings of HLT-EMNLP*, 2005.

[34] S. Kumar and W. Byrne, "Local phrase reordering models for statistical machine translation," in *Proceedings of HLT-EMNLP*, 2005.

[35] D. et al Dechelotte, "Investigating translation of parliament speeches," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2005.

[36] C. Allauzen, M. Mohri, and B. Roark, "Generalized algorithms for constructing statistical language models," in *Proceedings of the Association for Computational Linguistics*, July 2003.

[37] L.R. Bahl, P.F. Brown, P.V. deSouza, and R.L. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, Tokyo, 1986, pp. 49–52.

[38] A. Lavie and A. Agarwal, "Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments," in *Proceedings of the Association for Computational Linguistics*, 2007.

[39] Franz Josef Och, "Minimum error rate training in statistical machine translation," in *Proceedings of the Association for Computational Linguistics*, 2003.

[40] Franz Josef Och and Hermann Ney, "Discriminative training and maximum entropy models for statistical machine translation," in *Proceedings of the Association for Computational Linguistics*, Morristown, NJ, USA, 2002, pp. 295–302.

[41] D.A. Smith and Jason Eisner, "Minimum risk annealing for training log-linear models," in *Proceedings of the Association for Computational Linguistics*, 2006.

[42] Richard Zens, Hasan Sasa, and Hermann. Ney, "A systematic comparison of training criteria for statistical machine translation," in *EMNLP*, 2007, pp. 524–532.

[43] Libin Shen, Anoop Sarkar, and Franz Josef Och, "Discriminative reranking for machine translation," in *Proceedings of the Human Language Technology Conference of the NAACL*, 2004.

[44] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian O. Flannery, *Numerical Recipes in C++*, Cambridge University Press, 2002.

[45] L.E Baum and J.A Eagon, "An inequality with applications to statistical prediction for functions of markov processes and to a model of ecology," in *Bull. Amer. Math Soc.*, 1967, vol. 73, pp. 360–363.

[46] P.S Gopalkrishnan, Dimitri Kanevsky, A. Nadas, and D. Nahamoo, "An inequality for rational functions with applications to some statistical estimation problems," *IEEE Trans. Inform Theory*, vol. 37, no. 1, 1991.

[47] Dimitri Kanevsky, "A generalization of the baum algorithm to functions on non-linear manifolds," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 1995, vol. 1, pp. 473 – 476.

[48] P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar, "An end-to-end discriminative approach to machine translation," in *Proceedings of the Association for Computational Linguistics*, 2006.

[49] Yonggang Deng and William Byrne, "Mttk: an alignment toolkit for statistical machine translation," in *Proceedings of the Human Language Technology Conference of the NAACL*, 2006.

[50] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proceedings International Conference on Speech and Language Processing*, 2002.

[51] Steve Young, Gunnar Evermann, Thomas Hain, Dan Kershaw, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Valtcho Valtchev, and Phil Woodland, *The HTK Book, Version 3.1*, Dec. 2001.

[52] L. Chen, L. Lamib, and J.-L. Gauvain, "Transcribing mandarin broadcast news," in *IEEE ASRU Workshop*, 2003.

[53] Mehryar Mohri, Fernando Pereira, and Michael Riley, "The design principles of a weighted finite-state transducer library," *Theoretical Computer Science*, vol. 231, no. 1, pp. 17–32, 2000.

[54] V. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics - Doklady*, vol. 10, no. 10, pp. 707–710, 1966.

[55] Shankar Kumar, Yonggang Deng, and William Byrne, "A weighted finite state transducer translation template model for statistical machine translation," Clsp tech report 48, Johns Hopkins University, 2004.

[56] L. R. Rabiner and B. H. Juang, "An introduction to hidden markov models," *IEEE ASSP Magazine*, pp. 4–16, 1986.

[57] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the em algorithm (with discussion)," *Journal of the Royal Statistical Society*, vol. 39, 1977.

[58] A. Wald, "Note on the consistency of the maximum likelihood estimate," in *Annals of Mathematical Statistics*, 1949.

[59] A. Nádas, "A decision theoretic formulation of a training problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood," *IEEE Transactions on Acoustics, Speech and Signal Processing,*, vol. 31, no. 4, 1983.

[60] Y.-L. Chow, "Maximum mutual information estimation of hmm parameters for continuous speech recognition using the n-best algorithm," in *International IEEE International Conference on Acoustics, Speech, and Signal Processing Conference on Acoustics, Speech, and Signal Processing*, 1990.

[61] Phil Woodland and Daniel Povey, "Large scale discriminative training for speech recognition," in *Proceedings ITW ASR, ISCA*, 2000.

[62] Phil Woodland and Daniel Povey, "Large Scale Discriminative Training of Hidden Markov Models for Speech Recognition," *Computer Speech and Language*, vol. 16, no. 1, pp. 25–47, 2002.

[63] Y. Normandin, *Hidden Markov Models, Maximum Mutual Information Estimation and the Speech Recognition Problem*, Ph.D. thesis, McGill University, Montreal, 1991.

[64] Asela Gunawardana, "Maximum mutual information estimation of acoustic hmm emission densities," Tech. Rep., Johns Hopkins Univerisity, 3400 N. Charles St., Baltimore, MD 21218, 2001.

[65] H.Y. Chan and Phil Woodland, "Improving broadcast news transcription by lightly supervised discriminative training," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 2004, vol. 1, pp. 737–740.

[66] Chen Langzhou, Lori Lamel, and Jean-Luc Gauvain, "Lightly supervised acoustic model training using consensus networks," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 2004, vol. 1, pp. 189–192.

[67] V. Valtchev, J.J. Odell, P.C. Woodland, and S.J. Young, "Mmie training of large vocabulary recognition systems," *Speech Communication*, vol. 22, 1997.

[68] Lambert Mathias, Girija Yegnanarayanan, and Juergen Fritsch, "Discriminative training of acoustic models applied to domains with unreliable transcripts. in," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing*, 2005.

[69] S. Kumar V. Goel and W. Byrne, "Segmental minimum bayes-risk decoding for automatic speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 12, no. 3, pp. 287–333, 2004.

[70] Shankar Kumar and William Byrne, "Risk based lattice cutting for segmental minimum bayes risk decoding," in *Proceedings International Conference on Speech and Language Processing*, 2002.

[71] Brian Roark, Murat Saraclar, and Michael Collins, "Discriminative n-gram language modeling," in *Computer Speech and Language*, 2007, vol. 21.

Vita

Lambert Mathias was born in Mumbai, India. He graduated from the University of Mumbai in 2000, with a B.E. in Electronics. After receiving a M.S. in Electrical Engineering from Michigan State University in 2002, he joined the PhD program in the Department of Electrical and Computer Engineering, the Johns Hopkins University. Since then, he has been pursing his Ph.D. in Electrical and Computer Engineering at the Center for Language and Speech Processing at the Johns Hopkins University.

His research interests include statistical modeling and machine learning, and their applications in machine translation, speech recognition, information retrieval and other problems in speech and language processing.