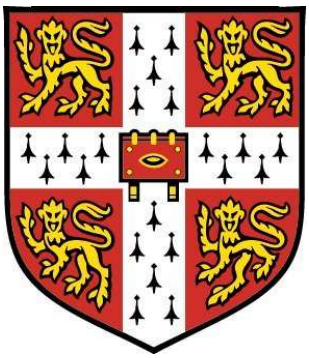
Lattice Rescoring Methods for Statistical Machine Translation

Graeme Blackwood

Cambridge University Engineering Department
and
Clare College



Dissertation submitted to the University of Cambridge
for the degree of Doctor of Philosophy

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text. It has not been submitted in whole or in part for a degree at any other university. Some of the work has been published previously in conference proceedings (Blackwood et al., 2008a; Blackwood et al., 2008b; Blackwood et al., 2009; Kurimo et al. (2009)) and a journal article (de Gispert et al., 2010), or accepted for publication in forthcoming conference proceedings (Blackwood and Byrne, 2010). The length of this thesis including appendices, references, footnotes, tables and equations is approximately 53,000 words and it contains 56 figures and 58 tables.

Summary

Modern statistical machine translation (SMT) systems include multiple interrelated components, statistical models, and processes. Translation is often factored as a cascaded series of modules such that the output of one module serves as the input to the next; this is the SMT pipeline. Simplifying assumptions, limited training data, and pruning during search mean that the maximum likelihood hypothesis may not represent the best translation. Since any errors will be propagated through the SMT pipeline, it is better to avoid hard decisions by passing on as much information as possible to subsequent modules. The focus, then, is less on finding the single-best translation and more on being able to generate a rich space of likely translations that can be exploited through subsequent rescoring and combination techniques. The large size of the search space in SMT means that it is not always possible to apply more complex models in translation decoding; such models are normally applied to a translation lattice, a space efficient representation of many translation alternatives with scores.

This thesis develops a robust inventory of large-scale lattice rescoring methods that improve the quality of statistical machine translation. These rescoring methods include (i) sentence-specific, high-order language models estimated over multi-billion word corpora, (ii) stochastic segmentation transducers that model the phrasal segmentation process in phrase-based SMT, (iii) efficient large-scale lattice minimum Bayes-risk decoding procedures based on weighted path counting transducers, (iv) multi-input and multi-source lattice combination techniques that synthesise multiple sources of translation knowledge, and (v) a novel decoding framework based on segmentation of a word lattice into regions of high and low confidence that supports targeted application of modelling techniques intended to address particular deficiencies in translation. Efficient realisations of these lattice rescoring methods are described in terms of general purpose weighted finite state transducer operations.

A second theme of this thesis concerns the exploitation of monolingual corpora. Although monolingual data is much more widely available than parallel data, in SMT it is typically only used for building word-based language models. However, there are other complementary ways in which this data can be used to improve translation quality. Two novel lattice rescoring methods for exploiting monolingual corpora - phrasal segmentation models that learn the segmentation of sequences of words into sequences of translatable phrases, and monolingual coverage constraints that address the often overlooked issue of machine translation fluency - are proposed in this thesis.

Keywords: statistical machine translation, statistical language modelling, lattice rescoring, minimum Bayes-risk decoding, exploiting monolingual data

Acknowledgements

First and foremost, I owe my deepest gratitude to my supervisor Dr. William Byrne for his invaluable guidance and assistance throughout the course of my studies. His focus, passion and drive have shaped my approach to research and will continue to influence me in my future career. I would also like to thank my advisor Dr. Mark Gales, Professor Phil Woodland, and the other members of staff in the Department of Engineering for their kind support.

It is an honour to have worked with Dr. Adrià de Gispert over the last three years and I have learned a great deal from his scientific rigour and infectious enthusiasm. I am also grateful to my colleagues Jamie Brunning, Gonzalo Iglesias, Juan Pino, Rory Waite and other members of the Speech Group of the Machine Intelligence laboratory for illuminating discussions and valuable feedback.

I am particularly grateful to the computer officers Patrick Gosling and Anna Langley for maintaining to a consistently high standard the excellent computing infrastructure that supported my numerous experiments.

I would like to show my gratitude to Dr. Matt Gibson for his considerable help in setting up the web application, and for organising and administering the manual evaluation. I would also like to thank the participants of the manual evaluation who generously gave their time to provide valuable judgements of machine translation fluency.

I would like to thank Cyril Allauzen, Michael Riley and the other developers of OpenFst at Google for providing and maintaining such an excellent toolkit, without which much of this work would have been impossible.

Finally, I am eternally indebted to my parents Victor and Jean, my brother Iain, his fiancée Lucy, and my wonderful wife Minobu for their endless support and encouragement whenever it was most needed; this thesis could never have been written without them.

The research described in this thesis was supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022.

Acronyms

AER	Alignment Error Rate
AGILE	Autonomous Global Integrated Language Exploitation
ASR	Automatic Speech Recognition
BLEU	Bilingual Evaluation Understudy
BP	Brevity Penalty
CFG	Context Free Grammar
CNF	Chomsky Normal Form
CUED	Cambridge University Engineering Department
DAG	Directed Acyclic Graph
EM	Expectation Maximisation
FSA	Finite-State Acceptor
FST	Finite-State Transducer
GALE	Global Autonomous Language Exploitation
GIS	Generalised Iterative Scaling
LDC	Linguistic Data Consortium
LM	Language Model
LMBR	Lattice Minimum Bayes-Risk
LMBR-SC	Lattice Minimum Bayes-Risk System Combination
MAP	Maximum A Posteriori
MBR	Minimum Bayes-Risk
MERT	Minimum Error Rate Training
METEOR	Metric for Evaluation of Translation with Explicit Reordering
MGPSM	Multi gram Phrasal Segmentation Model
MIRA	Margin Infused Relaxed Algorithm
ML	Maximum Likelihood
MT	Machine Translation
NIST	National Institute of Standards and Technology
NLG	Natural Language Generation
NLP	Natural Language Processing
OCR	Optical Character Recognition
OOV	Out-of-Vocabulary
PER	Position-Independent Error Rate
PP	Perplexity
PSM	Phrasal Segmentation Model
ROVER	Recogniser Output Voting Error Reduction
SCFG	Synchronous Context Free Grammar

STG	Stochastic Text Generation
SMT	Statistical Machine Translation
TER	Translation Edit Rate
TTM	Translation Transducer Model
WFSA	Weighted Finite-State Acceptor
WFST	Weighted Finite-State Transducer
WMT	Workshop on Statistical Machine Translation

Notation

These are the terms and notation used throughout this work.

Variables, Symbols and Operations

$\approx$	approximately equal to
$\propto$	proportional to
$\underset{x}{\operatorname{argmax}} f(x)$	the value of x that maximises the value of $f(x)$
$\underset{x}{\operatorname{argmin}} f(x)$	the value of x that minimises the value of $f(x)$
$\log(x)$	logarithm base e of x
$\exp(x)$	exponential of x
$\mathbb{E}[f(x)]$	the expected value of $f(x)$, where x is a random variable
$\sum_{n=1}^{N} a_n$	summation of terms from $n = 1$ to N – that is, $a_1 + a_2 + \cdots + a_N$
$\prod_{n=1}^{N} a_n$	product of terms from $n = 1$ to N – that is, $a_1 \times a_2 \times \cdots \times a_N$
$\min(x, y)$	minimum value of x and y
$\max(x, y)$	maximum value of x and y
$\#_x(y)$	count of x in y

Weighted Finite State Transducers

$\mathbb{K}$	semiring weight set
$\mathbb{R}_+$	set of positive real numbers
$T(a, b)$	weight associated by transducer T to string pair (a, b)
$A(a)$	weight associated by acceptor A to string a
$T_1 \circ T_2$	composition of two transducers T_1 and T_2
$A_1 \cap A_2$	intersection of two acceptors A_1 and A_2
$T_1 \otimes T_2$	concatenation of two transducers T_1 and T_2
$T_1 \oplus T_2$	union of two transducers T_1 and T_2
$\Pi_1[T]$	input label projection of transducer T
$\Pi_2[T]$	output label projection of transducer T
Q	finite set of states
I	finite set of initial states: $I \subseteq Q$
F	finite set of final states: $F \subseteq Q$
E	finite set of transitions – that is, edges
$\otimes$	binary operator for combining weights along a path
$\oplus$	binary operator for combining weights of identically labelled paths
$\bar{0}$	designated value: for all $x \in \mathbb{K}$, $x \oplus \bar{0} = x$ and $x \otimes \bar{0} = \bar{0} \otimes x = \bar{0}$
$\bar{1}$	designated value: for all $x \in \mathbb{K}$, $x \otimes \bar{1} = x$
$\oplus_{\log}$	log semiring weight combination: $x \oplus_{\log} y = -\log(e^{-x} + e^{-y})$
$E[q]$	set of all transitions from state q
$\lambda[q]$	initial state weight function $\lambda : I \to \mathbb{K}$
$\rho[q]$	final state weight function $\rho : F \to \mathbb{K}$
π	complete path denoted by the series of transitions $e_1 \cdots e_K$
$\otimes_{n=1}^{N} x_n$	operator $\otimes$ applied to N terms: $x_1 \otimes x_2 \otimes \cdots \otimes x_N$
$\oplus_{n=1}^{N} x_n$	operator $\oplus$ applied to N terms: $x_1 \oplus x_2 \oplus \cdots \oplus x_N$
σ	consuming composition symbol (matches all)
ρ	consuming composition symbol (matches rest)
ϵ	non-consuming composition symbol (matches all)
ϕ	non-consuming composition symbol (matches rest)

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Machine translation (MT) is the process of automatically translating written text or speech in one language (the *source*) into a different language (the *target*). One possible future application is a multilingual, real-time speech-to-speech translation device such as the Babelfish in The Hitchhiker's Guide to the Galaxy (Adams, 1979). Such a sophisticated translation device, however, is still very much a long-term goal; machine translation, particularly speech translation, is a highly complex task with many unresolved difficulties. Differences in lexical choice, word order, and grammatical structure, the use of idiomatic expressions and non-literal translations, and the presence or absence of particular cultural conventions all combine to make high quality automatic machine translation extremely challenging.

The statistical approach to machine translation (Koehn, 2010), driven largely by the increased availability of parallel training corpora and widespread acceptance of automatic quality metrics such as BLEU (Papineni et al., 2002b), addresses many of these issues by learning correspondences between the source and target languages from a large collection of translation examples. Statistical machine translation (SMT) chooses from the space of all possible translations of the source language sentence, the most likely target language translation given the source sentence and trained parameters of the statistical model.

Rapid progress has been made in statistical machine translation since the original word-based formulation of Brown et al. (1990). Significant advances include the move to translation models based on phrases (Och, 2002; Koehn, 2010), the incorporation of discriminative training and parameter optimisation (Och and Ney, 2002; Och, 2003), and the introduction of synchronous context-free grammars capable of supporting sophisticated reordering and movement of phrases (Chiang, 2005, 2007). Depending on the genre and nature of the translation task, however, both fluency and adequacy are still often lacking in translations produced using SMT. There is certainly a great deal of room for improvement.

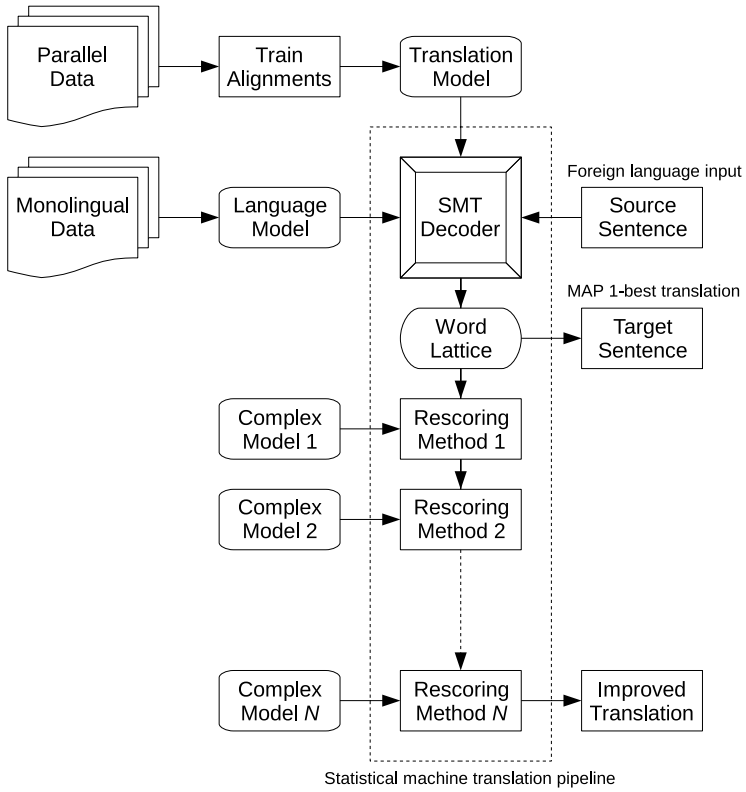


Figure 1.1: Cascaded module implementation of the statistical machine translation pipeline.

1.1 Lattice Rescoring Methods

Modern SMT systems are highly complex and include a number of interrelated components, statistical models, and processes. Translation is normally factored into a cascaded series of modules such that each module generates output for consumption by subsequent modules; this series of modules forms the SMT processing pipeline illustrated in Figure 1.1.

Incorrect assumptions, insufficient training data, and pruning during search mean that the *maximum a posteriori* (MAP) hypothesis may not represent the best possible translation of the source sentence. Since errors in the output of one module are propagated to subsequent modules, it is better to avoid making hard decisions and instead pass on as much information as possible to subsequent modules. The focus, then, is less on producing the single best translation and more on being able to generate a rich space of possible translations that can be effectively exploited by subsequent post-processing and combination techniques.

The large size of the search space in SMT decoding means that it is sometimes impossible to apply the most sophisticated models to the full space of translation hypotheses. For this reason, it is common practice to generate a large subset of the most likely translations according to the first-pass decoder, and then re-rank the hypotheses with more sophisticated models. A weighted word lattice (Ueffing et al., 2002; Kumar and Byrne, 2003) is a space-efficient representation of a large number of ranked translation alternatives and scores. The best translation in the lattice (the *oracle* translation) is usually significantly better than the MAP 1-best translation produced by the first-pass SMT system. The goal of lattice rescoring is to re-rank translation hypotheses so that their ranking better reflects their quality.

This thesis develops a practical and robust inventory of large-scale lattice rescoring methods that are demonstrated to result in significant improvements in the quality of statistical machine translation. Several rescoring strategies are empirically investigated. Efficient realisations of these rescoring methods and algorithms are described in terms of general purpose weighted finite-state transducer (WFST) operations (Mohri et al., 2008).

Translation lattices can be rescored with a more powerful language model (LM) than is normally possible in first-pass translation. This thesis shows how WFSTs can be used for efficient SMT lattice rescoring with sentence-specific n-gram LMs (Chen and Goodman, 1998; Huang et al., 2001) estimated over multi-billion word training corpora; significant improvements in BLEU score are observed with respect to the baseline system (Blackwood et al., 2009). Phrasal segmentation models based on stochastic segmentation transducers are demonstrated to improve the quality of phrase-based SMT (Blackwood et al., 2008b). An implementation of minimum Bayes-risk (MBR) decoding for large SMT lattices (Tromble et al., 2008) is described in terms of general purpose operations and algorithms on weighted finite-state acceptors (WFSA). An improved lattice MBR decoder based on efficient path counting transducers allows for fast and exact computation of the required statistics (Blackwood and Byrne, 2010). The lattice MBR decoding framework is then extended to the task of combining multiple lattices generated from alternative analyses of the source language sentence (Kurimo et al., 2009; de Gispert et al., 2010). This lattice rescoring method combines multiple sources of translation knowledge and leads to significant improvements in translation quality as measured by the BLEU score (Papineni et al., 2002b).

Another contribution of this thesis is a novel lattice rescoring framework for improving the quality of statistical machine translation. The motivation is to use n-gram posterior probabilities (Zens and Ney, 2006) as a confidence measure to identify portions of the lattice that are suspected to be of low quality. Starting from the best available SMT system, general purpose WFST operations and algorithms can be applied to segment a word lattice into regions of high and low confidence. The high confidence regions are trusted and left unmodified. Specialised models can then be applied to particular problems in the regions of low confidence. Lattice segmentation simplifies the problem of improving SMT quality by making it easier to integrate new modelling approaches into good baseline systems.

1.2 Exploiting Monolingual Data

A second theme of this thesis concerns the exploitation of large monolingual corpora to improve the quality of statistical machine translation. Although parallel text collections such as the proceedings of the United Nations (Graff, 1994), Canadian Hansard (Germann, 2001), and European Parliament (Koehn, 2005) are of paramount importance in training the parameters of statistical translation models, parallel data is expensive to produce and therefore usually only available in limited quantities. Large Arabic↔English and Chinese↔English parallel corpora exist, but less data is available for other language pairs. This issue of data sparsity is a serious problem for statistical approaches to machine translation.

Much larger monolingual text collections are available. In most SMT systems, monolingual data is only used to train the parameters of an n-gram language model (LM) (Chen and Goodman, 1998). SMT has been shown to continue to benefit from increasing quantities of language model training data; the largest experiments reported in the literature use a 5-gram LM estimated over approximately 1.8 trillion tokens of English text (Brants et al., 2007). SMT

research usually views increasing monolingual data as simply facilitating higher order n-gram language models and better parameter estimation. However, there are other complementary ways in which this data can be used to improve the quality of machine translation. Two novel methods for exploiting monolingual data – phrasal segmentation models and monolingual coverage constraints – are proposed in this thesis.

A phrasal segmentation model (PSM) can be used to rescore lattices produced by a phrase-based statistical machine translation decoder (Blackwood et al., 2008a). The model defines a mapping from the words of a sentence to a sequence of translatable phrases, where the space of possible segmentations is determined by the inventory of phrase pairs extracted from word-aligned parallel data. This thesis shows how phrasal segmentation model parameters can be estimated from a large monolingual corpus and applied in lattice rescoring. A first-order phrasal segmentation model implemented using WFSTs is demonstrated to result in improved translation quality as measured by the BLEU score.

Monolingual coverage constraints are another way in which monolingual data can be used to improve the quality of statistical machine translation. These constraints address a sometimes overlooked aspect of machine translation: hypothesis fluency. Monolingual coverage constraints based on high-order n-gram coverage in a large monolingual text collection are used to filter hypotheses believed to be disfluent from the hypothesis space of an MBR decoder. A human evaluation of the translation output shows that performing MBR search in the filtered hypothesis space leads to improved overall fluency.

1.3 Original Contributions

The original contributions of this thesis are summarised below:

1. A comprehensive inventory of large-scale SMT lattice rescoring methods are developed, leading to robust and significant improvements in the quality of translation. These rescoring methods remain in continuous use at CUED for SMT system development and research, and have contributed significant gains to highly ranked state-of-the-art submissions in recent blind evaluations of statistical machine translation quality.

2. A fast and exact method for linearised lattice minimum Bayes-risk decoding (Tromble et al., 2008) based on efficient path counting transducers is proposed (Blackwood and Byrne, 2010); this method is shown to perform well even for large SMT lattices. An original multiple lattice generalisation of the MBR decoder framework is extended to the task of multi-input (de Gispert et al., 2009) and multi-source (Och and Ney, 2001) translation. These methods support efficient combination of multiple SMT lattices and lead to large gains in translation quality as measured by the BLEU score.

3. Two lattice rescoring methods are proposed for improving the quality of SMT through the exploitation of abundantly available monolingual data: (i) phrasal segmentation models that can be used to improve the quality of phrase-based SMT, and (ii) monolingual coverage constraints for addressing the issue of poor fluency in SMT. These rescoring methods both provide improvements in quality that are complementary to the improvements obtained using higher-order second-pass n-gram language models.

4. A novel lattice rescoring framework is proposed for improving SMT quality through separation of the hypothesis space and evidence space of a minimum Bayes-risk decoder. Segmenting translation lattices into regions of high and low confidence allows the low confidence regions to be refined through targeted application of modelling approaches and procedures intended to address particular deficiencies in first-pass decoding. MBR decoding in the refined hypothesis space is demonstrated to result in improved translation fluency.

1.4 Organisation of Thesis

The remainder of this thesis is organised as follows. The weighted finite-state transducer operations and algorithms used throughout this work are described in Chapter 2. An example, the stochastic transformation of strings under a generative model, illustrates the practical application of these techniques. Chapter 3 presents an overview of n-gram language models, focusing in particular on the backoff and smoothing methods required for state-of-the-art performance. Chapter 3 includes a summary of recent approaches to large-scale distributed language modelling, and a discussion of the finite-state implementation of a backoff n-gram language model. The statistical approach to machine translation is reviewed in Chapter 4, together with a detailed description of the hierarchical phrase-based decoder used to generate the lattices for many of the subsequent rescoring experiments.

Chapter 5 describes the large language modelling experiments that serve as the baseline for many of the rescoring methods developed throughout this work. Phrasal segmentation models are defined and evaluated in Chapter 6. Lattice minimum Bayes-risk (LMBR) decoding is described in Chapter 7; a fast implementation of LMBR based on efficient path counting transducers is proposed and evaluated in the context of Arabic→English and Chinese→English translation experiments. Significant improvements in decoding efficiency are demonstrated. Lattice MBR decoding is extended to the task of multiple lattice combination for multi-input and multi-source translation in Chapter 8. Chapter 9 proposes a novel framework for improving statistical machine translation quality based on segmenting translation lattices using n-gram posterior probabilities. An application of this framework, monolingual coverage constraints, is shown to improve the fluency of Arabic→English machine translation. Chapter 10 reviews the original contributions and suggests possible areas for future research that build upon the ideas proposed in this thesis.

Chapter 2

Natural Language Processing with Weighted Finite-State Transducers

Weighted finite-state transducers (WFSTs) have been found useful in a variety of natural language processing (NLP) tasks including automatic speech recognition, speech synthesis, morphology, optical character recognition, part-of-speech tagging, and in other fields such as biological sequence processing. This chapter introduces the general purpose WFST operations and algorithms used throughout this work. The presentation derives from material in Mohri (1997), Mohri et al. (2000), Mohri (2002), Allauzen et al. (2003), Allauzen et al. (2007), and Mohri et al. (2008).[1]

2.1 Introduction

Transitions between states in a finite state transducer are labelled with both input and output labels. Paths through a finite-state transducer thus define a mapping from input label sequences to output label sequences. If each transition also has a weight, then the accumulation

[1] See also the OpenFst documentation: http://www.openfst.org/

of weights along a path through the WFST determines the weight of the mapping. In NLP applications, weights often encode probabilities or, for numerical stability, log probabilities. The weighted mapping between string pairs defined by a WFST makes it an appropriate representation for the probabilistic finite-state models common in NLP tasks.

2.2 Semiring Definitions

A semiring $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ is defined by a set of values $\mathbb{K}$, two binary operators $\oplus$ and $\otimes$, and designated values $\bar{0}$ and $\bar{1}$. The $\otimes$ operator is used to combine weights along a path or when matching paths in composition or intersection. The $\oplus$ operator is used to combine the weights of identically labelled paths. Four commonly used semirings are shown in Figure 2.1.

Semiring	Weight Set	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
boolean	$\{0, 1\}$	$\vee$	$\wedge$	0	1
probability	$\mathbb{R}_+$	$+$	$\times$	0	1
log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{\log}$	$+$	$+\infty$	0
tropical	$\mathbb{R} \cup \{-\infty, +\infty\}$	min	$+$	$+\infty$	0

Figure 2.1: Semirings often used for natural language processing (Mohri et al., 2008).

The *probability* (or *real*) semiring is appropriate when the transition weights represent probabilities. The *log* semiring (isomorphic to the probability semiring under $-\log$) is frequently used in automatic speech recognition and machine translation since it offers greater numerical stability. The *tropical* semiring is derived from the log semiring under the Viterbi approximation (Huang et al., 2001). It is appropriate when there is a need for shortest path algorithms, e.g. to apply the argmax or argmin operations in a decoder decision rule. The tropical and log semirings differ only in their interpretation of the $\oplus$ operator. In the tropical semiring, two weights x and y are combined as $x \oplus y = \min(x, y)$. In the log semiring, the weights are combined as $x \oplus y = -\log(e^{-x} + e^{-y})$. This is sometimes denoted $x \oplus_{\log} y$.

2.3 Transducers, Paths, and Weights

Formally, a weighted finite-state transducer $T = (\mathcal{A}, \mathcal{B}, Q, I, F, E, \lambda, \rho)$ over weight set $\mathbb{K}$ is defined by an input alphabet $\mathcal{A}$, an output alphabet $\mathcal{B}$, a set of states Q, a set of initial states $I \subseteq Q$, a set of final states $F \subseteq Q$, a set of weighted transitions E, an initial state weight assignment $\lambda : I \to \mathbb{K}$, and a final state weight assignment $\rho : F \to \mathbb{K}$ (Mohri et al., 2008). The sets $\mathcal{A}$, $\mathcal{B}$, Q, I, F, and E are all of finite size. For each state $q \in Q$, let $E[q]$ denote the set of all transitions (i.e. edges) leaving state q. The weighted transitions of T form the set

$$E \subseteq Q \times (\mathcal{A} \cup \{\epsilon\}) \times (\mathcal{B} \cup \{\epsilon\}) \times \mathbb{K} \times Q, \tag{2.1}$$

where each transition includes an origin or source state from Q, an input symbol from $\mathcal{A} \cup \{\epsilon\}$, an output symbol from $\mathcal{B} \cup \{\epsilon\}$, a cost from $\mathbb{K}$, and a destination or target state from Q.

Weighted finite-state acceptors are a special case of weighted finite-state transducers in which the input or output labels are omitted. An acceptor A for the input strings or output strings of transducer T is created by projecting on the input or output labels. This operation is denoted by $A = \Pi_1(T)$ for input projection and $A = \Pi_2(T)$ for output projection.

For transition $e \in E$, let $p[e]$ denote its source state, $n[e]$ its target state, $i[e]$ its input label, $o[e]$ its output label, and $w[e]$ its weight. Let $\pi = e_1 \cdots e_K$ denote a complete path in T from initial state $p[e_1]$ to final state $n[e_K]$, so that $n[e_{k-1}] = p[e_k]$ for $k = 2, \ldots, K$. The weight of the path π is the $\otimes$-product of the weights of the transitions:

$$w[\pi] = \bigotimes_{k=1}^{K} w[e_k] = w[e_1] \otimes \cdots \otimes w[e_K] \tag{2.2}$$

Let $p[\pi] = p[e_1]$ and $n[\pi] = n[e_K]$. If $\mathcal{P}(I, a, b, F)$ denotes the set of all paths in T starting from an initial state in I with input label sequence $a \in \mathcal{A}^*$ and output label sequence $b \in \mathcal{B}^*$ and ending in a final state in F, then the weight $T(a, b)$ associated by transducer T to any pair of input-output strings (a, b) is obtained as the $\oplus$-sum over all matching paths:

$$T(a, b) = \bigoplus_{\pi \in \mathcal{P}(I,a,b,F)} \lambda[p[\pi]] \otimes w[\pi] \otimes \rho[n[\pi]] \tag{2.3}$$

The weighted finite-state transducer T thus defines a weighted relation between strings $a \in \mathcal{A}^*$ in the input alphabet and strings $b \in \mathcal{B}^*$ in the output alphabet. The weight $T(a, b) = \bar{0}$ is associated to string pairs (a, b) not in T. The weight $A(a)$ associated by an acceptor to string a is computed as the $\oplus$-sum over paths $\pi \in \mathcal{P}(I, a, F)$.

2.4 Operations and Algorithms

This section gives a brief overview of the general purpose WFST operations and algorithms that will be used in this thesis (see Mohri et al. (2008) for details). The operations and algorithms manipulate the set of strings and weights in transducers and acceptors in accordance with the semiring definition $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$. WFST operations include concatenation and union of string sequences, probabilistic model combination through weighted composition, time and space optimisation through determinization and minimisation, and weight pushing for distributing transition weights appropriately for particular tasks.

The *union* of two transducers T_1 and T_2 contains the union of the string pairs in T_1 and the string pairs in T_2. The weight associated by the union is

$$(T_1 \oplus T_2)(a, b) = T_1(a, b) \oplus T_2(a, b) \tag{2.4}$$

The *concatenation* of two transducers T_1 and T_2 contains the concatenation of the string pairs in T_1 and the string pairs in T_2. For each string pair (a, b) formed from the concatenation of substring pairs (a_1, b_1) in T_1 and (a_2, b_2) in T_2, the weight associated by concatenation is

$$(T_1 \otimes T_2)(a, b) = \bigoplus_{a=a_1a_2, b=b_1b_2} T_1(a_1, b_1) \otimes T_2(a_2, b_2) \tag{2.5}$$

Subsequent chapters will rely heavily on *weighted composition*. The weight associated to the string pair (a, b) by the composition of two transducers T_1 and T_2 with matching respective output and input alphabets $\mathcal{C}$ is defined as

$$(T_1 \circ T_2)(a, b) = \bigoplus_{c \in \mathcal{C}^*} \{ T_1(a, c) \otimes T_2(c, b) \} \tag{2.6}$$

The finite-state acceptor equivalent of composition is weighted *intersection*. The weight associated to string a by the intersection of two acceptors is $(A_1 \cap A_2)(a) = A_1(a) \otimes A_2(a)$. Other operations that manipulate the language of strings represented by a WFST include closure, reverse, invert, and difference (Allauzen et al., 2007).

2.4.1 Optimisation and Search Procedures

General purpose operations and algorithms are available for optimising WFSTs with respect to time and memory. Optimisation does not affect the language of accepted strings or associated weights – only the transducer topology and distribution of weights is modified.

The *connect* operation removes unreachable states and arcs. The *rmepsilon* operation removes transitions that have both input and output label ϵ. The *determinize* and *minimise* operations can be used to create an equivalent, minimal, deterministic transducer with the property that no state has more than one transition with the same input label. These operations can significantly reduce the number of states and arcs.

The *shortest-path* algorithm in the tropical semiring can be used to find the k lowest cost paths in a transducer (Mohri, 2002). This allows the best string(s) to be efficiently extracted using the Viterbi approximation and provides a generic implementation of the argmax and argmin operations in probabilistic models. The *prune* operation discards paths based on a cost threshold relative to the cost of the shortest path in the transducer.

The *push* operation redistributes transition weights in a way that does not affect the weight associated to complete paths. Pushing weights towards the initial state results in a *stochastic* machine with the property that for each state $q \in Q$, the $\oplus$-sum of outgoing transition weights and final state weight $\{\bigoplus_{e \in E[q]} w[e]\} \oplus \rho[q]$ is $\bar{1}$. If weights are instead pushed towards the final states, then for each state $q \in Q$ the $\oplus$-sum of incoming transition weights $\bigoplus_{e \in E:n[e]=q} w[e]$ is $\bar{1}$. Weight pushing is useful for optimising search procedures and for converting path weights to normalised probabilities.

2.4.2 Special Label Matching

Special symbol matching (Allauzen et al., 2007) can be used in weighted composition. Special symbols act as transition filters in composition and enable more compact topologies and faster matching. The special symbols are ϵ (epsilon), σ (all), ρ (rest), and ϕ (fail). Their behaviour is summarised in Table 2.2. Transitions labelled σ match and consume any arc in composition. Transitions labelled ρ match and consume any arc without an explicit transition from the state. Non-consuming ϕ-transitions are similar to ϵ transitions but can only be taken when no regular symbol match is possible. ϕ-transitions are required for the exact implementation of backoff in the WFST representation of an n-gram language model (Allauzen et al., 2003).

Matches	Consuming	
	Y	N
All	σ	ϵ
Rest	ρ	ϕ

Figure 2.2: Special label matching in weighted composition.

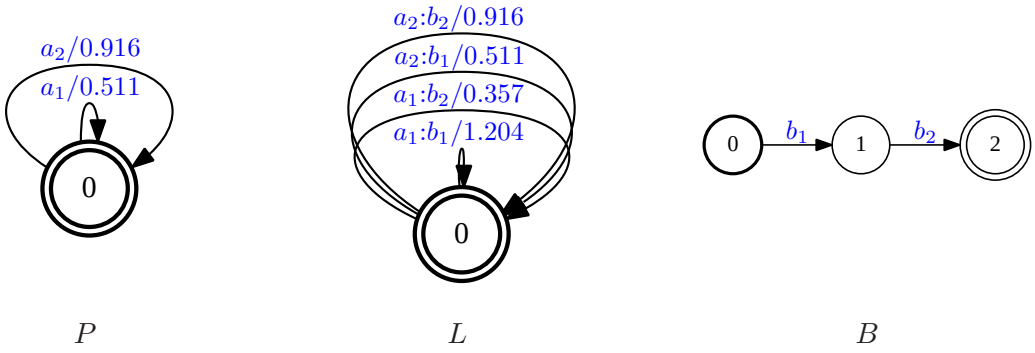


Figure 2.3: WFST implementation of a simple decoder that models the stochastic transformation of strings. Acceptor P encodes the prior distribution $P(\mathbf{a})$, transducer L encodes the conditional distribution $P(\mathbf{b}|\mathbf{a})$, and acceptor B encodes the observed string $\mathbf{b} = b_1 b_2$.

2.4.3 Stochastic Decoding with WFSTs

This section provides examples of the main WFST operations and algorithms used throughout this work. Suppose the transformation of strings $\mathbf{a} \in \mathcal{A}^*$ to strings $\mathbf{b} \in \mathcal{B}^*$ is modelled as a generative stochastic process. The goal is to find the most likely source string $\mathbf{a}$ given the observed string $\mathbf{b}$. This is the string $\hat{\mathbf{a}}$ that maximises the conditional probability $P(\mathbf{a}|\mathbf{b})$. Using Bayes' rule, the maximum likelihood decoder decision rule has the form:

$$\hat{\mathbf{a}} = \underset{\mathbf{a}}{\operatorname{argmax}}\, P(\mathbf{a}|\mathbf{b}) = \underset{\mathbf{a}}{\operatorname{argmax}}\, P(\mathbf{b}|\mathbf{a})P(\mathbf{a}) \tag{2.7}$$

For a string of length I, let $P(\mathbf{a}) = \prod_{i=1}^{I} P(a_i)$ and $P(\mathbf{b}|\mathbf{a}) = \prod_{i=1}^{I} P(b_i|a_i)$. Let the alphabet $\mathcal{A} = \{a_1, a_2\}$ and $\mathcal{B} = \{b_1, b_2\}$. Suppose the model parameters $P(a_i)$ and $P(b_i|a_i)$ for all $a_i \in \mathcal{A}$ and $b_i \in \mathcal{B}$ have been estimated as follows:

	$P(a)$	$P(b_1\|a)$	$P(b_2\|a)$
a_1	0.6	0.3	0.7
a_2	0.4	0.6	0.4

The most likely source string $\hat{\mathbf{a}}$ for observed string $\mathbf{b}$ can be found using the WFSTs shown in Figure 2.3. Weights are shown for the tropical semiring so that probability p is represented as weight $-\log p$. The prior distribution $P(\mathbf{a})$ is implemented by acceptor P and assigns $P(\mathbf{a}) = \prod_{i=1}^{I} P(a_i)$ to any sequence $\mathbf{a} \in \mathcal{A}^*$. The conditional distribution $P(\mathbf{b}|\mathbf{a})$ is implemented by transducer L which transduces source sequences $\mathbf{a}$ to observed sequences $\mathbf{b}$ with probability $P(\mathbf{b}|\mathbf{a}) = \prod_{i=1}^{I} P(b_i|a_i)$. The product $P(\mathbf{b}|\mathbf{a})P(\mathbf{a})$ is found by weighted composition. The search space of the maximum likelihood decoder of Equation (2.7) can be found by the composition chain

$$A = P \circ L \circ B, \tag{2.8}$$

where B accepts the observed string $\mathbf{b}$ and input label sequences in A are source strings $\mathbf{a}$ that might have generated $\mathbf{b}$, each with weight $-\log P(\mathbf{b}|\mathbf{a})P(\mathbf{a})$ according to the model. The

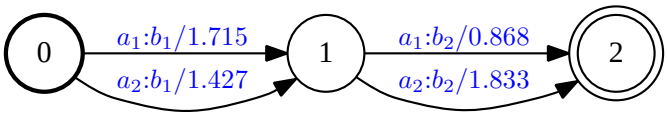


Figure 2.4: Decoder search space A for the input sequence $\mathbf{b} = b_1 b_2$.

transducer A resulting from the composition of Equation (2.8) is shown in Figure 2.4. The most likely source string $\hat{\mathbf{a}}$ is the input label sequence in A with least cost:

$$\hat{\mathbf{a}} = \text{shortestpath}(\Pi_1(A)) \tag{2.9}$$

For input string $\mathbf{b} = b_1 b_2$ encoded by acceptor B in Figure 2.3, the shortest path in A is the sequence $\hat{\mathbf{a}} = a_2 a_1$ with joint probability $P(a_2 a_1, b_1 b_2) = P(b_1|a_2) \times P(b_2|a_1) \times P(a_2) \times P(a_1) = 0.6 \times 0.7 \times 0.4 \times 0.6 = 0.1008$. This corresponds to weight $-\log P(a_2 a_1, b_1 b_2) = 2.295$ in the log semiring. The joint probabilities of all source strings $\mathbf{a}$ generating $\mathbf{b}$ are as follows:

$\mathbf{a}$	$-\log P(\mathbf{a}, \mathbf{b})$	$P(\mathbf{a}, \mathbf{b})$
$a_2 a_1$	2.29461670	0.1008
$a_1 a_1$	2.58229899	0.0756
$a_2 a_2$	3.25969768	0.0384
$a_1 a_2$	3.54737997	0.0288

The marginal probability $P(\mathbf{b}) = \sum_{\mathbf{a}} P(\mathbf{a}, \mathbf{b})$ can be computed efficiently from the decoder search space transducer A by projecting on the input labels and then pushing weights towards the final state in the log semiring. The resulting acceptor is shown in Figure 2.5.

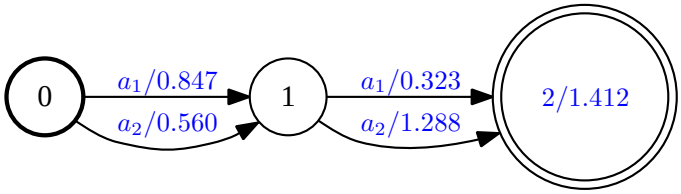


Figure 2.5: Source strings acceptor after pushing weights towards the final state.

The final state cost is $-\log P(\mathbf{b}) = -\log \sum_{\mathbf{a}} P(\mathbf{a}, \mathbf{b})$. Summing $P(\mathbf{a}, \mathbf{b})$ in the table of joint probabilities gives $P(\mathbf{b}) = 0.2436$ and $-\log 0.2436 = 1.412$ which agrees with the final state cost. Removing the final state cost normalises the path probabilities. A then defines the posterior probability distribution $P(\mathbf{a}|\mathbf{b})$ over strings $\mathbf{a} \in A$ given the observed string $\mathbf{b}$ so that $\sum_{\mathbf{a} \in A} P(\mathbf{a}|\mathbf{b}) = 1$. The cost of each $\mathbf{a} \in A$ is then $-\log P(\mathbf{a}|\mathbf{b})$. The cost of each path in A is the negative log of

$$P(\mathbf{a}|\mathbf{b}) = \frac{P(\mathbf{a}, \mathbf{b})}{\sum_{\mathbf{a}' \in A} P(\mathbf{a}', \mathbf{b})}. \tag{2.10}$$

Weight pushing in the log semiring thus provides an efficient method for marginalisation over identically labelled sequences, and for the conversion of likelihoods or model scores to a normalised posterior probability distribution. These marginalisation and normalisation operations are used extensively throughout this work.

Chapter 3
Statistical Language Modelling

Sentences in a natural language consist of an ordered sequence of words and punctuation symbols. A statistical language model (LM) trained from a large corpus of monolingual training text can be used to assign a likelihood to a sequence of words, or to predict the word most likely to follow a given history (or context) of preceding words. Language models have applications in any field where the goal is to produce fluent natural language as the output. Statistical language models are particularly important in automatic speech recognition (ASR) (Huang et al., 2001) and statistical machine translation (SMT) (Koehn, 2010) since they are used to guide the search procedure of the decoder and ensure that the resulting output is of high quality (Jelinek, 1998).

3.1 Introduction

This chapter first reviews the n-gram approach to statistical language modelling that is integral to many of the lattice rescoring methods described in this thesis. The n-gram language model is described and defined in Section 3.2. Sections 3.2.1 and 3.2.2 summarise techniques for addressing the issue of data sparsity: frequency discounting, interpolation with lower-order distributions, and backing off to more reliable probability estimates. Section 3.2.3 summarises the main n-gram smoothing methods used in this thesis. Large-scale statistical language models trained using multi-billion word corpora are described in Section 3.3. The chapter concludes with a description of the finite-state representation of a backoff n-gram language model in Section 3.4.

3.2 N-gram Language Models

In automatic speech recognition and statistical machine translation, n-gram language models (as summarised in Jelinek (1998), Huang et al. (2001), Jurafsky and Martin (2008), and Koehn (2010)) can be used to assign a likelihood to a sequence of words. This likelihood is related to the *a priori* probability of the sequence of words in the language. Intuitively, word sequences that are grammatical and express sensible semantic relationships should be assigned a high likelihood by the language model; other sequences should be assigned a low likelihood. The language model probability is usually combined with a conditional probability (e.g. of acoustic observations in ASR or foreign words in SMT) to calculate the posterior probability of hypotheses during maximum likelihood decoding under the source-channel model of information processing.

The n-gram approach to language modelling is effective because (i) the models capture both syntax and semantics, (ii) they focus on important local grammatical relationships, and (iii) they have a simple dependency structure that allows for efficient training and integration in ASR and SMT decoding. The main disadvantages are that they ignore the structure of natural language, and that limited training data can result in unreliable estimates, particularly for those word sequences that were not observed in the training data.

An n-gram language model defines a probability distribution over sequences of words, where the probability assigned to each word sequence is related to the likelihood of occurrence of the sequence of words as a sentence in the language. Formally, the probability assigned to the word sequence $\mathbf{w} = w_1 w_2 \dots w_{|\mathbf{w}|}$ of length $|\mathbf{w}|$ is computed using the chain rule as the product of the conditional probability of each word in the sequence given the history of preceding words:

$$P(\mathbf{w}) = \prod_{i=1}^{|\mathbf{w}|} P(w_i | w_1 w_2 \dots w_{i-1}) \tag{3.1}$$

Since there are a potentially infinite number of possible word sequences in natural language there will never be enough data to reliably train a model conditioned on the entire history of words $w_1 \dots w_{i-1}$. The model must be able to generalise to provide good predictions of word probabilities even for sequences that were not present in the training data. The Markov assumption that only local context is relevant when predicting which word follows a given history defines a set of equivalence classes between strings that share the same initial sequence of words. This reduces the number of possible histories to a much more manageable level. The length of the sequence is the n-gram order. The Markov independence assumption for an n-gram model of order n approximates the probability of a sequence of words $\mathbf{w}$ as

$$P(\mathbf{w}) \approx \prod_{i=1}^{|\mathbf{w}|} P(w_i | w_{i-n+1}^{i-1}) \tag{3.2}$$

Maximum likelihood (ML) estimation can be used to train the parameters of an n-gram language model from the relative frequency of n-gram word sequences in a large training corpus. For a model of order n, the conditional probability of word w_i given the preceding history of words w_{i-n+1}^{i-1} is computed by relative frequency as

$$P(w_i | w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})}, \tag{3.3}$$

where the function $c(\cdot)$ simply counts the frequency of the specified n-gram in the language model training data. These parameter estimates maximise the training data likelihood.

Although the Markov assumption in Equation (3.2) significantly reduces the number of model parameters, the problem of data sparsity means that the majority of higher-order n-gram word sequences will not be observed in the training data. The ML probability assigned by Equation (3.3) to such n-grams is zero, regardless of how likely the sequence of words might be. This is clearly undesirable since the n-gram may be a perfectly grammatical and quite probable sequence of natural language that just happened to be missing from the training data. Sections 3.2.1, 3.2.2 and 3.2.3 discuss standard strategies for addressing the data sparsity issue.

3.2.1 Language Model Discounting

The sum of conditional probabilities $P(w_i|w_{i-n+1}^{i-1})$ taken over all n-grams with the same history must be 1 for Equation (3.3) to be a valid probability distribution. In order to assign probability mass to unseen events it is necessary to discount the probabilities of seen events. The discounted n-gram probability estimate is

$$P(w_i|w_{i-n+1}^{i-1}) = d(r)\frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})}, \tag{3.4}$$

where $d(r)$ is a discount coefficient that specifies the amount of the discount and is usually a function of the frequency r of the n-gram being predicted. Simple discounting schemes such as absolute discounting or linear discounting subtract a small fixed constant or scale the observed counts so that the discounted probability mass can be reassigned to unobserved n-grams. Witten-Bell discounting (Bell et al., 1990) computes discount coefficients that are proportional to the number of distinct words that follow the n-gram history.

Good-Turing discounting (Good, 1953) adjusts the observed frequencies such that an n-gram that occurs r times in the training data is treated as if it had occurred r^* times. The modified counts are computed from the observed counts as

$$r^* = (r+1)\frac{n_{r+1}}{n_r}, \tag{3.5}$$

where n_r denotes the number of n-grams that occur r times in the training data. Only the counts of low frequency n-grams are adjusted in this way since the counts of high frequency n-grams are assumed to be reliable. Good-Turing discounting reserves a proportion n_1/N of the total probability mass for unseen n-grams, where N is the total number of tokens in the training corpus. Estimating the parameters of a language model using Good-Turing discounting requires computing both the regular n-gram counts and also the count-of-counts n_r for $r < k$ where k is the maximum order at which discounting should be applied.

3.2.2 Language Model Interpolation and Backoff

Interpolation and backoff are two common strategies for improving the reliability of word predictions in an n-gram language model. Both techniques compensate for the problem of data sparsity through the use of lower-order probability distributions. The use of lower-order distributions in backoff and interpolated models differs since the interpolated model always smoothes estimates using the lower-order distribution, while the backoff model does so only for n-grams with counts lower than the cutoff threshold.

3.2.2.1 Backoff Models

Most n-gram language models used in ASR and SMT are backoff models (Katz, 1987). The general form of a backoff n-gram language model defines the conditional probability $P_{\text{BO}}(w_i|w_{i-n+1}^{i-1})$ of word w_i given history w_{i-n+1}^{i-1} recursively as

$$P_{\text{BO}}(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \alpha(w_i|w_{i-n+1}^{i-1}) & \text{if } c(w_{i-n+1}^{i}) > k \\ \gamma(w_{i-n+1}^{i-1})P_{\text{BO}}(w_i|w_{i-n+2}^{i-1}) & \text{otherwise} \end{cases} \tag{3.6}$$

where α is the discounted probability distribution that allows mass to be reassigned to unseen n-grams, γ is a backoff weight specific to the n-gram history w_{i-n+1}^{i-1} (this is required for normalisation), and k is the n-gram frequency cutoff point that determines the counts for which the backed-off $(n-1)$-gram probability is used. The cutoff frequency is often $k = 0$ so that the lower-order distribution is only used for unseen n-grams. Intuitively, this model continues backing off until an n-gram with sufficient frequency for a reliable estimate of the word probability is found. The recursion ends at the unigram distribution.

3.2.2.2 Interpolated Models

The n-gram probabilities in an interpolated language model are computed from a linear interpolation of higher-order and lower-order distributions. Effectively, the lower-order distributions are used to smooth the higher-order sparser distributions, resulting in more reliable parameter estimates. The interpolated probability is computed recursively as follows:

$$P_{\text{INTERP}}(w_i|w_{i-n+1}^{i-1}) = \lambda P_{\text{ML}}(w_i|w_{i-n+1}^{i-1}) + (1-\lambda)P_{\text{INTERP}}(w_i|w_{i-n+2}^{i-1}) \tag{3.7}$$

Interpolation is thus a weighted sum of probabilities computed from n-grams of different orders. The weights of the interpolated model can be optimised on a corpus of representative held-out data using deleted interpolation (Bahl et al., 1990). The interpolation weights can also be conditioned on the context. The optimised weights then indicate the reliability of the distribution at each order, given the history.

3.2.3 Language Model Smoothing

Language model smoothing combines discounting, interpolation, and backoff to obtain estimates of word sequence probabilities that are closer to the true distribution of words in the language. There are several smoothing strategies of varying complexity and effectiveness.

3.2.3.1 Additive Smoothing

One of the simplest methods of language model smoothing consists of adding a small fixed constant to the counts of each n-gram observed in the training data:

$$P_{\text{ADD}}(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i}) + \delta}{c(w_{i-n+1}^{i-1}) + \delta|V|} \tag{3.8}$$

An example of this form of additive smoothing is to choose $\delta = 1$ such that each n-gram is assumed to have occurred once more than is actually found to be the case in the training data. This ensures that no word sequences are assigned a probability of zero. However, this technique is known to perform poorly since it significantly overestimates the probability of unseen n-grams (Gale and Church, 1994).

3.2.3.2 Katz Smoothing

Katz smoothing (Katz, 1987) combines Good-Turing discounting (Good, 1953) of low frequency unreliable n-gram counts with backing off to lower-order distributions for unseen n-grams. The conditional probability of word w_i given history w_{i-n+1}^{i-1} is

$$P_{\text{KATZ}}(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})} & \text{if } r > k \\ d(r)\frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})} & \text{if } 0 < r \leq k \\ \gamma(w_{i-n+1}^{i-1})P_{\text{KATZ}}(w_i|w_{i-n+2}^{i-1}) & \text{if } r = 0 \end{cases}, \tag{3.9}$$

where k determines the range of count frequencies $r = c(w_{i-n+1}^{i})$ which should be discounted in order to reserve probability mass for unseen n-grams. Those n-grams with frequency $r > k$ are assumed to be reliable and assigned conditional probability $P_{\text{ML}}(w_i|w_{i-n+1}^{i-1})$. The probabilities of n-grams with frequencies $0 < r \leq k$ are discounted according to the discount coefficient $d(r)$:

$$d(r) = \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \tag{3.10}$$

The probability assigned to n-grams that are not observed in the training data is computed using the lower order $(n-1)$-gram distribution. The context-specific backoff weight $\gamma(w_{i-n+1}^{i-1})$ ensures that the distribution satisfies the probability constraint $\sum_{w_i} P_{\text{KATZ}}(w_i|w_{i-n+1}^{i-1}) = 1$. The backoff weights $\gamma(w_{i-n+1}^{i-1})$ are computed as follows. Let $\beta(w_{i-n+1}^{i-1})$ denote the probability mass that remains after discounting the probabilities of n-grams with frequencies $0 < r \leq k$. This probability is computed by subtracting from 1 the probabilities of all n-grams with non-zero counts:

$$\beta(w_{i-n+1}^{i-1}) = 1 - \sum_{w_i:r>0} P_{\text{KATZ}}(w_i|w_{i-n+1}^{i-1}) \tag{3.11}$$

This is the total discounted probability mass for context w_{i-n+1}^{i-1} that will be distributed evenly amongst the lower-order backed-off $(n-1)$-grams. The backoff weight $\gamma(w_{i-n+1}^{i-1})$ is obtained by normalising the discounted probability mass $\beta(w_{i-n+1}^{i-1})$ by the total probability of all $(n-1)$-grams w_{i-n+2}^{i} that begin backed-off n-grams w_{i-n+1}^{i} of frequency $r = 0$. The backoff weight is

$$\gamma(w_{i-n+1}^{i-1}) = \frac{\beta(w_{i-n+1}^{i-1})}{\sum_{w_i:r=0} P_{\text{KATZ}}(w_i|w_{i-n+2}^{i-1})} = \frac{1 - \sum_{w_i:r>0} P_{\text{KATZ}}(w_i|w_{i-n+1}^{i-1})}{1 - \sum_{w_i:r>0} P_{\text{KATZ}}(w_i|w_{i-n+2}^{i-1})}, \tag{3.12}$$

where the sum over $w_i : r = 0$ in the denominator is rewritten in terms of the sum over $w_i : r > 0$ since, for higher-order n, it is much more efficient to sum over the observed n-grams than the more numerous unobserved n-grams.

3.2.3.3 Kneser-Ney Smoothing

Kneser-Ney smoothing (Kneser and Ney, 1995) is the most commonly used smoothing method in modern ASR and SMT systems. A modified version that includes interpolation with lower-order distributions has been demonstrated to obtain better perplexities than any other smoothing method (Chen and Goodman, 1998).

The motivation for Kneser-Ney smoothing is that some words occur almost exclusively with certain other words and therefore any smoothing scheme that ignores the context (e.g. the backed-off unigram distribution in Katz smoothing of a bigram language model) will result in artificially high smoothed counts for word pairs that almost never co-occur. The canonical example is the bigram *san francisco* (Chen and Goodman, 1998). Since this bigram occurs frequently in the training data, the unigram probabilities of *san* and *francisco* are both relatively high. However, if it is necessary to back off to the unigram distribution then *francisco* should have low probability for any context other than *san*.

In a Kneser-Ney smoothed language model, the smoothed lower-order distribution is not computed from counts of n-grams, but instead depends on the number of unique words that precede the backed off n-gram. The Kneser-Ney smoothed probability is computed as

$$P_{\text{KN}}(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \frac{\max\{c(w_{i-n+1}^{i})-D,0\}}{c(w_{i-n+1}^{i-1})} & \text{if } c(w_{i-n+1}^{i}) > 0 \\ \gamma(w_{i-n+1}^{i-1})P_{\text{KN}}(w_i|w_{i-n+2}^{i-1}) & \text{otherwise} \end{cases}, \tag{3.13}$$

where the context-specific backoff weight $\gamma(w_{i-n+1}^{i-1})$ ensures the distribution is properly normalised. The lower-order distribution is computed from the counts of unique histories as

$$P_{\text{KN}}(w_i|w_{i-n+2}^{i-1}) = \frac{\mathbb{C}(\bullet\, w_{i-n+2}^{i})}{\sum_{w_i}\mathbb{C}(\bullet\, w_{i-n+2}^{i})} \tag{3.14}$$

where $\mathbb{C}(\bullet\, w_{i-n+2}^{i}) = |\{w_{i-n+1} : c(w_{i-n+1}^{i}) > 0\}|$ denotes the number of unique words that precede the backed off n-gram w_{i-n+2}^{i}. The modified version of Kneser-Ney smoothing replaces the single discount parameter D with separate discount parameters D_1, D_2, and D_{3+} for discounting n-grams with counts of 1, 2 and 3 or more respectively. The best language model performance is obtained by interpolating with the lower order distributions as in Equation (3.7). The interpolated modified Kneser-Ney smoothed probability is

$$P_{\text{MKN}}(w_i|w_{1-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i}) - D(c(w_{i-n+1}^{i}))}{c(w_{i-n+1}^{i-1})} + \gamma(w_{i-n+1}^{i-1})P_{\text{MKN}}(w_i|w_{i-n+2}^{i-1}) \tag{3.15}$$

The count-specific discount parameters are defined as

$$D(c) = \begin{cases} 0 & \text{if } c = 0 \\ D_1 & \text{if } c = 1 \\ D_2 & \text{if } c = 2 \\ D_{3+} & \text{if } c \geq 3 \end{cases} \tag{3.16}$$

The distribution must sum to 1, so the context specific term $\gamma(w_{i-n+1}^{i-1})$ is defined as

$$\gamma(w_{i-n+1}^{i-1}) = \frac{D_1\mathbb{C}_1(w_{i-n+1}^{i-1}\bullet) + D_2\mathbb{C}_2(w_{i-n+1}^{i-1}\bullet) + D_{3+}\mathbb{C}_{3+}(w_{i-n+1}^{i-1}\bullet)}{c(w_{i-n+1}^{i-1})}, \tag{3.17}$$

where $\mathbb{C}_x(w_{i-n+1}^{i-1}\bullet)$ is the number of unique words following history w_{i-n+1}^{i-1} amongst n-grams that are found to occur x times in the training data. The optimised discount parameters are computed from the counts-of-counts n_r in the training corpus as follows:

$$\begin{aligned} Y &= \tfrac{n_1}{n_1+2n_2} \\ D_1 &= 1 - 2Y(\tfrac{n_2}{n_1}) \\ D_2 &= 2 - 3Y(\tfrac{n_3}{n_2}) \\ D_{3+} &= 3 - 4Y(\tfrac{n_4}{n_3}) \end{aligned} \tag{3.18}$$

To compute the smoothed probabilities of a Kneser-Ney n-gram language model requires the regular counts c, the count-of-counts n_r, and the continuation counts $\mathbb{C}$. This makes estimating the parameters more expensive than other smoothing schemes.

3.3 Large-Scale Statistical Language Models

The size of the training data used to estimate the parameters of an n-gram language model is continually increasing. This is particularly true in English where multi-billion token corpora are increasingly the norm. The challenge is how to most effectively exploit this vast quantity of data. This section describes some of the problems associated with large-scale statistical language models, and surveys some recent approaches that seek to exploit the full set of available training data.

3.3.1 Distributed Language Models

The predictive power of n-gram language models usually increases at higher orders since there is a longer context on which to condition each word prediction. It is therefore desirable to use the highest possible order that can be reliably estimated from a given quantity of training data. However, higher order n-gram language models estimated over large corpora result in a huge number of model parameters; it is often impossible to store all of these probabilities in memory during decoding. Count frequency cutoffs (Stolcke, 2002), probability quantisation, entropy-based pruning (Stolcke, 1998), and Bloom filters (Talbot and Osborne, 2007) can be used to reduce the memory requirements of a language model, but these techniques discard potentially useful information that may degrade the quality of the model.

One solution to the problem of large-scale language models is to use distributed computing based on the client-server paradigm. In this framework, decoder clients connect to one or more remote language model servers and request n-gram probabilities or counts. Requests are typically batched for efficiency. For example, a stack-based machine translation decoder (see Chapter 4, Section 4.2) iteratively extends partial hypotheses by translating a single source language word or phrase. The n-grams required to compute the language model score of each partial hypothesis extension can be batched together and requested in a single remote procedure call. This substantially reduces the amount of network traffic during decoding.

Distributed language modelling is used for re-ranking k-best lists produced by a machine translation decoder in Zhang et al. (2006). The lists are re-ranked using a combination of n-gram language model probabilities and sentence likelihood features computed on demand from raw counts. The training corpus (3 billion tokens) is split into chunks and n-gram counts (orders $n = 1 \ldots 4$) for a single chunk are loaded into each server in the form of a suffix array. To obtain the n-gram probabilities required to re-rank each k-best list requires a separate query to each server in order to aggregate counts over the full corpus. The aggregate counts are then used to compute the required probabilities.

A similar n-gram counts server approach to implementing a distributed language model is integrated into a statistical machine translation decoder in Emami et al. (2007); experiments using a 5-gram language model estimated over 4 billion tokens show improved SMT quality. Again, n-gram probabilities are computed on demand by aggregating the counts obtained from each individual language model server.

An alternative distributed architecture for SMT decoding is described in Brants et al. (2007). Clients are served smoothed language model probabilities instead of counts so that only a single server needs to be contacted per n-gram request. This avoids the aggregation of counts over multiple servers. They describe a context-independent backoff scheme that considerably simplifies the parameter estimation and run-time complexity of the language model. The distributed architecture and simplified backoff implementation allows a 5-gram LM estimated over 1.8 trillion tokens to be efficiently integrated directly in SMT decoding.

3.3.2 Stupid Backoff Smoothing

Stupid backoff (Brants et al., 2007) is a simple form of language model smoothing that replaces the n-gram conditional probabilities $P(w_i|w_{i-n+1}^{i-1})$ in Equation (3.6) with non-normalised scores based on relative frequencies. The motivation for stupid backoff smoothing is that (i) it is inexpensive to calculate in a distributed environment and (ii) the quality approaches that of Kneser-Ney smoothing (Kneser and Ney, 1995) for very large training corpora. Zero-cutoff stupid backoff language model scores are defined recursively as

$$S(w_i|w_{i-n+1}^{i-1}) = \begin{cases} P(w_i|w_{i-n+1}^{i-1}) & \text{if } c(w_{i-n+1}^{i}) > 0 \\ \gamma(n) S(w_i|w_{i-n+2}^{i-1}) & \text{otherwise} \end{cases} \tag{3.19}$$

where the backoff weight $\gamma(n)$ depends only on the order n and is independent of the n-gram context. The conditional probability $P(w_i|w_{i-n+1}^{i-1})$ of n-grams observed in the training data is computed by maximum likelihood estimation from the relative frequency of counts:

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})} \tag{3.20}$$

There is no discounting of the maximum likelihood estimates. The recursion in Equation (3.19) ends with the definition for unigrams

$$S(w_i) = \frac{c(w_i)}{\sum_j c(w_j)} = \frac{c(w_i)}{N}, \tag{3.21}$$

where N is the total number of tokens in the training corpus. Only n-gram frequencies are required to compute the parameters of a stupid backoff language model. There is no need to compute the count-of-counts required by most other smoothing methods.

The simplicity of stupid backoff smoothing means that the language model parameters may be distributed in a way that allows the model to be scaled up to a very large size, while still allowing for efficient integration in a real-time translation decoder. In Brants et al. (2007), the effectiveness of stupid backoff smoothing is compared with Kneser-Ney smoothing at a range of different training corpora sizes; they show that translation quality continues to improve with larger training corpora and that there is no significant difference in quality between Kneser-Ney smoothing and stupid backoff smoothing once the size of the training data exceeds several billion tokens.

3.4 Finite-State Acceptor Language Models

One of the advantages of n-gram language models is that they have a very simple dependency structure. The parameters of a backoff n-gram language model can be encoded in a space efficient representation as a weighted finite-state acceptor (Allauzen et al., 2003). In the WFSA representation, each state encodes a word history. Arcs from states encode the conditional probability $P(w|w_{i-n+1}^{i-1})$ of target word w given the preceding history of words w_{i-n+1}^{i-1}.

Let $\mathcal{G}$ denote the WFSA representation of a language model. Let the strings in acceptor $\mathcal{L}$ denote word sequences $\mathbf{w}$ that are to be scored by the language model. The language model can be easily applied using weighted composition: $\mathcal{L} \circ \mathcal{G}$. After composition, each string in $\mathcal{L}$ has probability $P(\mathbf{w}) = \prod_{i=1}^{|\mathbf{w}|} P(w_i|w_{i-n+1}^{i-1})$. Figure 3.1 shows the subset of states and arcs in $\mathcal{G}$ that encode the conditional probabilities and backoff weights for the trigram $P(w_i|w_{i-2}^{i-1})$, as defined by the n-gram backoff language model of Equation (3.6).

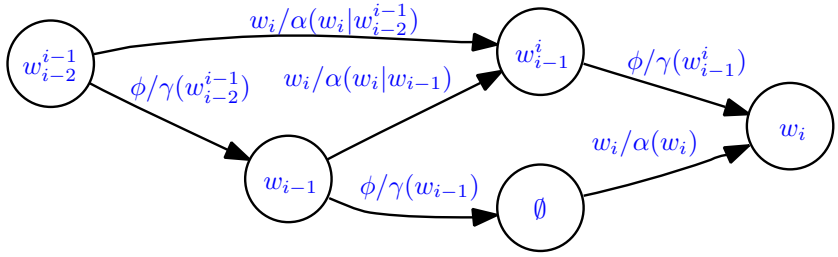


Figure 3.1: Finite-state acceptor representation of a trigram backoff language model. Only states and arcs that implement the conditional probability $P(w_i|w_{i-2}^{i-1})$ are shown for clarity.

The probability $P(w_i|w_{i-2}^{i-1})$ assigned to word w_i preceded by history w_{i-2}^{i-1} in the composition $\mathcal{L} \circ \mathcal{G}$ depends on whether or not it is necessary to backoff to a lower-order n-gram. If the trigram word sequence w_{i-2}^{i} did not occur in the language model training data, then its probability is computed by backing off to the lower-order bigram distribution. The probability is the $\otimes$-product of the backoff weight and bigram probability:

$$P(w_i|w_{i-2}^{i-1}) = \gamma(w_{i-2}^{i-1}) \otimes \alpha(w_i|w_{i-1}) \tag{3.22}$$

To ensure that the composition $\mathcal{L} \circ \mathcal{G}$ assigns the correct n-gram language model probabilities to strings, failure ϕ-transitions (Chapter 2, Section 2.4.3) must be used instead of ϵ-transitions for backoff arcs. ϕ-transitions ensure that the highest possible order of n-grams is used for each word prediction by allowing backoff arcs to be taken only if there are no regular word transition matches. This avoids the problem of assigning incorrect language model probabilities whenever $\gamma(w_{i-2}^{i-1}) \otimes \alpha(w_i|w_{i-1}) > \alpha(w_i|w_{i-2}^{i-1})$.

This way of encoding the parameters of a conditional distribution as an automaton is used extensively throughout this work. In Chapter 5, general purpose WFSA operations allow for efficient rescoring of large statistical machine translation lattices. A similar topology is used to encode the parameters of first-order phrasal segmentation models in Chapter 6. In Chapter 9, a variant of the WFSA n-gram language model is described that applies a fixed penalty to strings in proportion to the number of times backoff arcs were used. These penalties are the basis for monolingual coverage constraints that can be used to improve the fluency of machine translation output.

Chapter 4
Statistical Machine Translation

The mass availability of large quantities of electronic text in multiple languages and ready access to powerful and inexpensive computer hardware has made possible a data-driven, statistical approach to the problem of translating between natural languages. Statistical machine translation (SMT) combines the fields of natural language processing, computational linguistics, pattern recognition, and machine learning. One of the main motivations for statistical machine translation is that it consistently achieves state-of-the-art quality in evaluations such as those conducted by the National Institute of Standards and Technology (NIST)[1].

4.1 Introduction to Statistical Machine Translation

Classical approaches to machine translation (as summarised by Jurafsky and Martin (2008)) use linguistically motivated transfer rules or an intermediate representation known as an *interlingua*. The transfer approach in *rule-based* MT systems requires specialised linguistic knowledge to formulate rules that specify how different language features are mapped between the source and target language. In the interlingua approach, the aim is to extract semantic information and syntactic relationships from the source language sentence. The extracted information can then be mapped to any target language in order to render a translation of the input sentence. However, extracting meaning from a sentence requires a deep analysis

[1] http://www.itl.nist.gov/iad/mig/tests/mt

and sophisticated knowledge representation, together with some degree of world or domain-specific knowledge. High levels of linguistic expertise in the source and target languages are also required; this expertise is not always directly transferable to other language pairs.

Machine translation using transfer rules or an interlingua focuses on the process of translation. The statistical approach focuses on the result by framing translation as a generative stochastic process for which parameters can be estimated from a large corpus of example translations. This corpus is known as a *parallel text* or *bitext* and contains sentences with the same meaning in two (or more) languages. Popular parallel texts include the proceedings of the United Nations (Graff, 1994), Canadian Parliament (Germann, 2001), and European Parliament (Koehn, 2005). Large collections of Arabic↔English and Chinese↔English newswire parallel text are also available. Although parallel text collections are much larger now than when they first became available, they are still small in comparison with the volume of monolingual data available for estimating the parameters of a statistical language model.

The main advantages of statistical machine translation are (i) almost no specialised linguistic knowledge is required, (ii) the modelling procedures are largely language independent, (iii) there is the promise that natural and fluent translations can be learned directly from real training data, and (iv) idiomatic translations can be captured in context from observed examples. The ability of SMT systems to learn idiomatic translations is a significant advantage. In classical MT, these translations must be encoded manually and the exact circumstances under which they can be appropriately employed are very difficult to formalise. A popular example of a phrase-based SMT system is available as a free online service from Google.[1]

This chapter follows the convention of describing translation as the process of transforming a foreign input sentence $\mathbf{f}$ into an English output sentence $\mathbf{e}$. The language independent nature of statistical machine translation is one of its main benefits and $\mathbf{e}$ and $\mathbf{f}$ should be taken to represent sentences in any two natural languages.

4.1.1 The Source-Channel Model of Statistical Machine Translation

The first influential framework for statistical machine translation described the process of translating between two languages in terms of the source-channel model (Brown et al., 1990, 1993). Foreign sentences are considered to be English sentences that have passed through a noisy communication channel corrupting their surface form. The task of translation is to recover the hidden English sentence that generated the observed foreign sentence.

In the source-channel model, the goal is to recover the source sentence $\mathbf{e}$ generating target sentence $\mathbf{f}$ that maximises the conditional probability of translation $P(\mathbf{e}|\mathbf{f})$. Inspired by the use of the source-channel model in automatic speech recognition (ASR) (Huang et al., 2001), the conditional probability $P(\mathbf{e}|\mathbf{f})$ can be decomposed using Bayes' rule as follows:

$$P(\mathbf{e}|\mathbf{f}) = \frac{P(\mathbf{e},\mathbf{f})}{P(\mathbf{f})} = \frac{P(\mathbf{f}|\mathbf{e})P(\mathbf{e})}{P(\mathbf{f})} \tag{4.1}$$

Since the denominator $P(\mathbf{f})$ is constant for any given input sentence, it can be ignored during decoding. This leads to the simplified maximum likelihood (ML) decision rule

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}}\, P(\mathbf{f}|\mathbf{e})P(\mathbf{e}), \tag{4.2}$$

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

where $P(\mathbf{f}|\mathbf{e})$ represents the translation probability, $P(\mathbf{e})$ represents the language model probability, and the argmax operation denotes the search over possible translations of $\mathbf{f}$ for the best translation $\hat{\mathbf{e}}$. At an abstract level, the translation model favours translations that capture the semantic content of the foreign language sentence, whilst the language model favours translations that respect the grammaticality and fluency of the source language.

The translation model $P(\mathbf{f}|\mathbf{e})$ defines a probability distribution over sentence pairs $(\mathbf{e}, \mathbf{f})$ in the source and target language giving the probability that $\mathbf{e}$ generates $\mathbf{f}$. One of the main difficulties in estimating the probability that $\mathbf{e}$ generates $\mathbf{f}$ is the difference in word order between the source and target languages.

4.1.2 Word Alignments for Statistical Machine Translation

Word alignments define a mapping between the words of a source language sentence and a target language sentence known to be its translation. Links between words correspond to syntactic functions or semantic relationships shared by the words of the source and target sentences. One possible word alignment for a Spanish→English sentence pair is shown in Figure 4.1. The links show which English word in the source sentence generated each Spanish word in the target sentence. Some words must be reordered: the adjective and noun in the English noun-phrase 'green witch' must be reordered as 'bruja verde' in Spanish. Differences in word order are the main reason why high quality automatic machine translation between languages is so difficult.

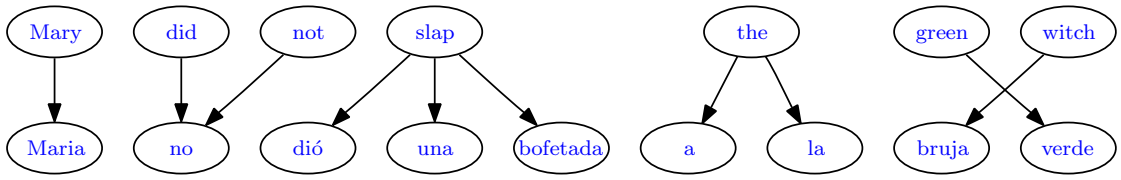


Figure 4.1: Word alignment example showing the one-to-many links between source and target words for a Spanish→English sentence pair (Jurafsky and Martin, 2008).

Brown et al. (1993) describes a series of five translation models of increasing sophistication known as IBM Model 1 to IBM Model 5. These translation models capture various features of the word alignment process. They describe algorithms for the unsupervised estimation of model parameters using a corpus of aligned parallel text, and a training procedure in which the parameters of each model serve as the initialisation for the next and more sophisticated model. Alignments are modelled by a hidden variable that specifies the source word to which each target word is aligned. Even though the word alignment between source and target sentences in the parallel data is not explicit, the alignment probabilities can still be learned using the expectation-maximisation algorithm (Dempster et al., 1977).

4.1.3 Phrase-Based Statistical Machine Translation

In the word-based generative model of statistical machine translation (Brown et al., 1993), words are inserted, deleted, translated and reordered according to distributions learned from the alignments. Phrase-based statistical machine translation (Koehn et al., 2003), developed from the Alignment Template approach of Och and Ney (2004), uses phrases instead of single words as the fundamental unit of translation.

In phrase-based translation, phrases are defined as any contiguous sequence of words and therefore have no syntactic or semantic significance other than that implied by their natural grouping as a phrase in the training corpus. The only requirement is that a translation for each phrase can be learned from parallel data. Although such an interpretation of phrases is somewhat unusual, it is a simplification that leads to significantly improved performance without requiring detailed knowledge of source and target language grammars.

The main advantages of phrase-based statistical machine translation are (i) a large phrase-pair lexicon can be induced from parallel data with high precision, (ii) phrases incorporate reorderings, insertions, and deletions that are sensitive to local context, (iii) semantic collocations that are useful for resolving translation ambiguities can be captured, and (iv) the words within phrases are sequences learned from real data resulting in more fluent translation output. Decoding with phrases also enables longer distance movement of words and can be less computationally demanding than word-based translation since there are fewer units to be translated. It is for these reasons that phrase-based methods have become the dominant paradigm in statistical machine translation research and evaluations.

Phrase-based statistical machine translation starts with the *segmentation* of foreign sentence $\mathbf{f}$ into a sequence of I phrases: $\bar{f}_1, \ldots, \bar{f}_I$. The segmentation process is not usually explicitly modelled so all segmentations are considered equally likely. Alternatives to the uniform phrasal segmentation distribution are the subject of Chapter 6. Each foreign phrase $\bar{f}_i$ is translated as English phrase $\bar{e}_i$ with phrase-to-phrase translation probability $\phi(\bar{f}_i|\bar{e}_i)$ estimated from parallel data. In decoding, the translation probability $P(\mathbf{f}|\mathbf{e})$ of Equation (4.2) is decomposed as the product of phrase-to-phrase translation probabilities so that

$$P(\mathbf{f}|\mathbf{e}) = P(\bar{f}_1^I|\bar{e}_1^I) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i) d(\text{start}_i - \text{end}_{i-1} - 1), \tag{4.3}$$

where the reordering distribution $d(\text{start}_i - \text{end}_{i-1} - 1)$ is a function of the relative number of words skipped forwards or backwards during decoding (Koehn, 2010), with start_i as the start position of the foreign phrase translated as phrase $\bar{e}_i$ and end_{i-1} as the end position of the foreign phrase translated as the preceding English phrase $\bar{e}_{i-1}$. This distribution can be estimated from parallel data or modelled as a simple exponential decay that penalises longer distance reorderings (Koehn et al., 2003).

Phrase-based statistical machine translation requires a lexicon of phrase-to-phrase translation probabilities extracted from the parallel data. A popular approach is the *phrase-extract* algorithm of Och (2002). This algorithm starts by generating IBM Model 4 word alignments in each direction (Brown et al., 1993) (Section 4.1.2). Finding alignments in both directions compensates for the asymmetric 1-to-1 and 1-to-n word alignment limitation of the IBM models. The two sets of alignments are then combined to form their union – a process known as *symmetrisation*. Although the union does contain most of the alignments of interest, it does so with relatively low precision and can include many spuriously aligned words. The second stage of the algorithm finds a subset of phrase-pairs that are well aligned according to a set of heuristics and such that words within a phrase-pair are not aligned to any words outside the pair. The resulting set of *consistent* bilingual phrase-pairs have high alignment precision. However, some foreign phrases will not be included in the phrase lexicon because word alignment errors failed to align them with suitable English phrases. The phrase lexicon can be expanded using heuristics that check the alignments for additional phrases that are consistent expansions of existing phrases.

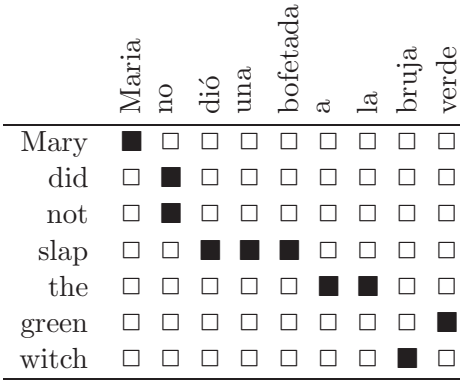


Figure 4.2: A word alignment matrix for a Spanish→English sentence pair that shows the alignment between words (Jurafsky and Martin, 2008).

Figure 4.2 shows a word alignment matrix for a Spanish→English sentence pair. Phrase-pairs identified by the phrase extraction algorithm might include ⟨Maria, Mary⟩, ⟨no dió una bofetada, did not slap⟩, and ⟨a la bruje verde, the green witch⟩. A word-based model must correctly reorder the Spanish source words 'bruja verde' as the English target words 'green witch'. In a phrase-based model, this local reordering is learned from the parallel data and encoded directly in the phrase-pair. Phrase-based models, therefore, reduce the need for explicit reordering in translation. For this example, a monotone translation decoder is able to generate the correct English word order without explicit reordering.

Usually, alignment probabilities are ignored when building the phrase translation model – only the presence or absence of alignment links is considered when computing phrase translation probabilities. Although the phrase extraction heuristics may have little theoretical justification, they have been found to work very well in practice.

For the set of phrase-pairs defined by the lexicon, a phrase translation table containing phrase-to-phrase translation probabilities can be estimated by relative frequency from counts in the aligned parallel corpus. These probabilities define the set of all possible translations of each foreign language input phrase, weighted with a probability distribution learned from the alignments. The maximum likelihood estimate of the probability of translating phrase $\bar{f}_i$ given phrase $\bar{e}_i$ is

$$\phi(\bar{f}_i|\bar{e}_i) = \frac{\text{count}(\bar{e}_i, \bar{f}_i)}{\sum_{\bar{f}_j} \text{count}(\bar{e}_i, \bar{f}_j)} \tag{4.4}$$

where $\text{count}(\bar{x}, \bar{y})$ is simply a count of the number of times the phrase-pair $(\bar{x}, \bar{y})$ is aligned in the parallel training corpus. These conditional probability distributions define the translation model for phrase-based statistical machine translation.

Phrase-based statistical machine translation systems are often modelled as a log-linear combination of features (Section 4.1.4) and typically include both $P(\mathbf{e}|\mathbf{f})$ (source→target) and $P(\mathbf{f}|\mathbf{e})$ (target→source) translation probabilities. The translation model can also be improved through the addition of lexical translation probabilities in each direction (Koehn et al., 2003). These features model how well the words in each phrase-pair align with one another and act to smooth the phrase translation model using the richer statistics of the word-to-word alignments. Lexical translation probabilities are particularly useful for rare phrase pairs that may occur only once or twice in the training data and are therefore usually assigned too much probability mass by the maximum likelihood estimate of Equation (4.4).

4.1.4 Maximum Entropy Models and Direct Translation

The statistical approach to machine translation is based on the IBM word alignment models of Brown et al. (1993). In a similar manner to automatic speech recognition, the process of translation is formulated in terms of the source-channel model: the source sentence is considered to have passed through a noisy-channel that corrupts its surface form into the words of the foreign language. The decoding task is to recover the source sentence.

Och and Ney (2002) proposed a direct translation modelling framework that extends and generalises the source-channel model, citing the following limitations as their motivation: (i) the source-channel decision rule is optimal only if the true translation and language model probability distributions are known, and this is never true in practice since it would require an infinite quantity of training data; (ii) it is difficult to integrate additional sources of knowledge; and, (iii) alternative decision rules may simplify translation decoding.

Instead of inverting and decomposing via Bayes' rule, the posterior probability of a candidate translation can be modelled directly using maximum entropy (Berger et al., 1996; Papineni et al., 1998). The maximum entropy model is defined by a set of M feature functions $h_m(\mathbf{e}, \mathbf{f})$ and associated feature weights λ_m for $m = 1, \ldots, M$. The direct translation probability is given by

$$P(\mathbf{e}|\mathbf{f}) = \frac{\exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}))}{Z(\mathbf{f})}, \tag{4.5}$$

where the normalisation $Z(\mathbf{f})$ in the denominator is required to satisfy the constraint that $P(\mathbf{e}|\mathbf{f})$ is a valid probability distribution for all $\mathbf{f}$. The normalisation factor

$$Z(\mathbf{f}) = \sum_{\mathbf{e}} \exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f})) \tag{4.6}$$

is constant for all hypothesised translations and can therefore be ignored during search. The direct translation decision rule therefore simplifies as follows:

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \left\{ P(\mathbf{e}|\mathbf{f}) \right\} = \underset{\mathbf{e}}{\operatorname{argmax}} \left\{ \sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}) \right\} \tag{4.7}$$

Decoding under the maximum entropy model thus selects the translation $\mathbf{e}$ that maximises the dot product of feature value and feature weight vectors. More advanced features, such as those that are a function of the alignment between source and target sentences[1], can be incorporated by decomposing the translation probability using hidden variables and extending the feature functions and decision rule to include the hidden variable dependencies. For features that include alignments, the feature function has the form $h_m(\mathbf{e}, \mathbf{f}, \mathbf{a})$. Since the sum over alignments is expensive to compute, the Viterbi approximation is normally used.

The flexibility of the model allows for arbitrary features that are a function of the source and target sentences, although for decoding in a large search space it must be possible to compute the feature scores efficiently. Typical features include translation and alignment model features (e.g. phrase-to-phrase translation probabilities and lexical weights in each translation direction), language model features (such as a high-order n-gram, and class-based or part-of-speech features), a sentence length feature, a conventional lexicon score, lexical

[1] Under the direct translation model the input sentence is usually termed the *source* and the output sentence the *target*. This is precisely the opposite interpretation of these terms under the source-channel model.

relationship features (Och and Ney, 2002), and phrase-pair count features (Bender et al., 2007). More sophisticated linguistic features, such as parse-tree probabilities, can also be included. Note that if the model is restricted to just two feature functions $h_1(\mathbf{e}, \mathbf{f}) = \log P(\mathbf{f}|\mathbf{e})$ and $h_2(\mathbf{e}, \mathbf{f}) = \log P(\mathbf{e})$, and the feature weights are $\lambda_1 = \lambda_2 = 1$, then the direct translation maximum entropy model is equivalent to the source-channel model (Section 4.1.1).

The goal in training is to find the set of model parameters $\hat{\lambda}_1^M$ that maximise the training data likelihood. Och and Ney (2002) train the model parameters λ_m to maximise the class posterior probability criterion using the Generalised Iterative Scaling (GIS) algorithm (Darroch and Ratcliff, 1972) and convergence to a global optimum is guaranteed. Given training corpus $\mathbf{S} = \{(\mathbf{f}_s, \mathbf{e}_s)\}$ for $s = 1, \ldots, S$ the weights are optimised by

$$\hat{\lambda}_1^M = \underset{\lambda_1^M}{\operatorname{argmax}} \left\{ \sum_{s=1}^{S} \log P_{\lambda_1^M}(\mathbf{e}_s, \mathbf{f}_s) \right\} \tag{4.8}$$

This is equivalent to maximising the likelihood of the direct translation. However, given that the ultimate aim is to produce good translations of unseen testing sentences, there are better discriminative training criteria that directly maximise a translation metric over the training data (Och, 2003) (Section 4.5).

During parameter optimisation, the expensive normalisation required by the denominator of Equation (4.5) is approximated by an k-best list of highly probable translations. Pruning during search means that the k-best list might not contain the required reference translation, so the k-best hypothesis that has minimum word error with respect to the reference translations is chosen as the *pseudo-reference* for parameter optimisation.

4.2 Statistical Machine Translation Decoding

The goal in statistical machine translation decoding is to find the most likely target language translation $\hat{\mathbf{e}}$ given source language sentence $\mathbf{f}$. The name 'decoding' comes from cryptography and the original formulation of SMT as a generative stochastic process in which an English sentence $\mathbf{e}$ is passed through a noisy-channel corrupting or encoding its surface form as foreign sentence $\mathbf{f}$. In this framework, translation is the task of decoding the encoded sentence $\mathbf{f}$ to find the most likely original sentence $\hat{\mathbf{e}}$.

This section describes two approaches to phrase-based SMT decoding. An alternative decoding architecture, hierarchical phrase-based decoding, is described in Section 4.3.3.

4.2.1 Stack-Based Decoding and Pruning

The task of the decoder is to search the space of possible translations of a given input sentence for the most likely translation according to the model. This search can be defined as finding the English sentence $\hat{\mathbf{e}}$ for foreign sentence $\mathbf{f}$ that maximises the conditional probability of translation $P(\mathbf{e}|\mathbf{f})$. In a log-linear model of direct translation (Och and Ney, 2002) (Section 4.1.4), the decision rule chooses the sentence $\hat{\mathbf{e}}$ that maximises the dot product of feature weight and feature value vectors:

$$\begin{aligned} \hat{\mathbf{e}} &= \underset{\mathbf{e}}{\operatorname{argmax}}\, P(\mathbf{e}|\mathbf{f}) \\ &= \underset{\mathbf{e}}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}) \end{aligned} \tag{4.9}$$

Translation models usually include additional dependencies on hidden variables representing alignments and distortion, e.g. by including in Equation (4.9) features of the form $h_m(\mathbf{e}, \mathbf{f}, \mathbf{a})$ where $\mathbf{a}$ specifies the alignment between $\mathbf{e}$ and $\mathbf{f}$. These dependencies significantly increase decoding complexity such that an efficient search guaranteed to find the most likely hypothesis under the model cannot be implemented. In practice, approximations are used that render the search tractable at the expense of search errors.

Most phrase-based statistical machine translation decoders such as Pharaoh (Koehn, 2004), Moses (Koehn et al., 2007), and the decoder of Och and Ney (2004) generate translation hypotheses from left-to-right in target language word order. The search space has the form of a directed acyclic graph where states encode partial and complete target language translation hypotheses. This style of decoder is similar in form to an ASR beam search decoder (Huang et al., 2001).

Each state in the graph represents a partial or complete translation hypothesis. States are defined by (i) a coverage vector listing the words of the source language sentence translated by paths leading to the state, (ii) a language model history consisting of the $n-1$ previously generated target language words (this history allows the n-gram language model probability of each single word extension from the state to be computed), and (iii) a score representing the likelihood of the partial translation encoded by the state (i.e. the weighted sum of feature values in a log-linear model). Each state also includes a heuristic-based future cost estimate associated with translating the remaining source words not listed in the coverage vector. The most likely translation $\hat{\mathbf{e}}$ for source sentence $\mathbf{f}$ is the sequence of target language words along the path with least cost in the search graph. In addition to finding the 1-best translation, a lattice or k-best list of the top translation hypotheses can be generated by recording at each state back-pointers to previous partial hypotheses (Koehn, 2010).

Exact decoding can be implemented using an A^* search heuristic (Cormen et al., 2001). In practice, however, this is prohibitively slow so a beam search is used instead. In order to apply pruning fairly, the states in the search space are organised into stacks or priority queues so that states that cover the same source words are stored in the same queue. This ensures that the heuristic used to estimate the future cost is only used for comparing and pruning partial hypotheses that cover the same source words. Queues are usually pruned using a likelihood threshold relative to the highest scoring hypothesis, or by fixing the maximum size of the k-best translation hypotheses stored in each priority queue. The threshold can be tuned to balance speed and accuracy. If the beam is narrow, then decoding will be fast but many search errors will be made and the translation hypothesis selected by the decoder will probably not be the hypothesis with maximum score under the model.

4.2.2 Word Lattices for Statistical Machine Translation

Section 4.2.1 described how a statistical machine translation decoder can be used to generate a ranked k-best list or word lattice of the most likely hypotheses according to the translation model. Word lattices (Ueffing et al., 2002; Kumar and Byrne, 2003) are a much more space efficient representation than k-best lists and allow for astronomical numbers of translation hypotheses to be compactly encoded. Some of the largest lattices reported in the literature are estimated to encode on the order of 10^{80} hypotheses (Tromble et al., 2008). This section describes the representation of machine translation lattices as weighted finite-state acceptors (Mohri et al., 2008).

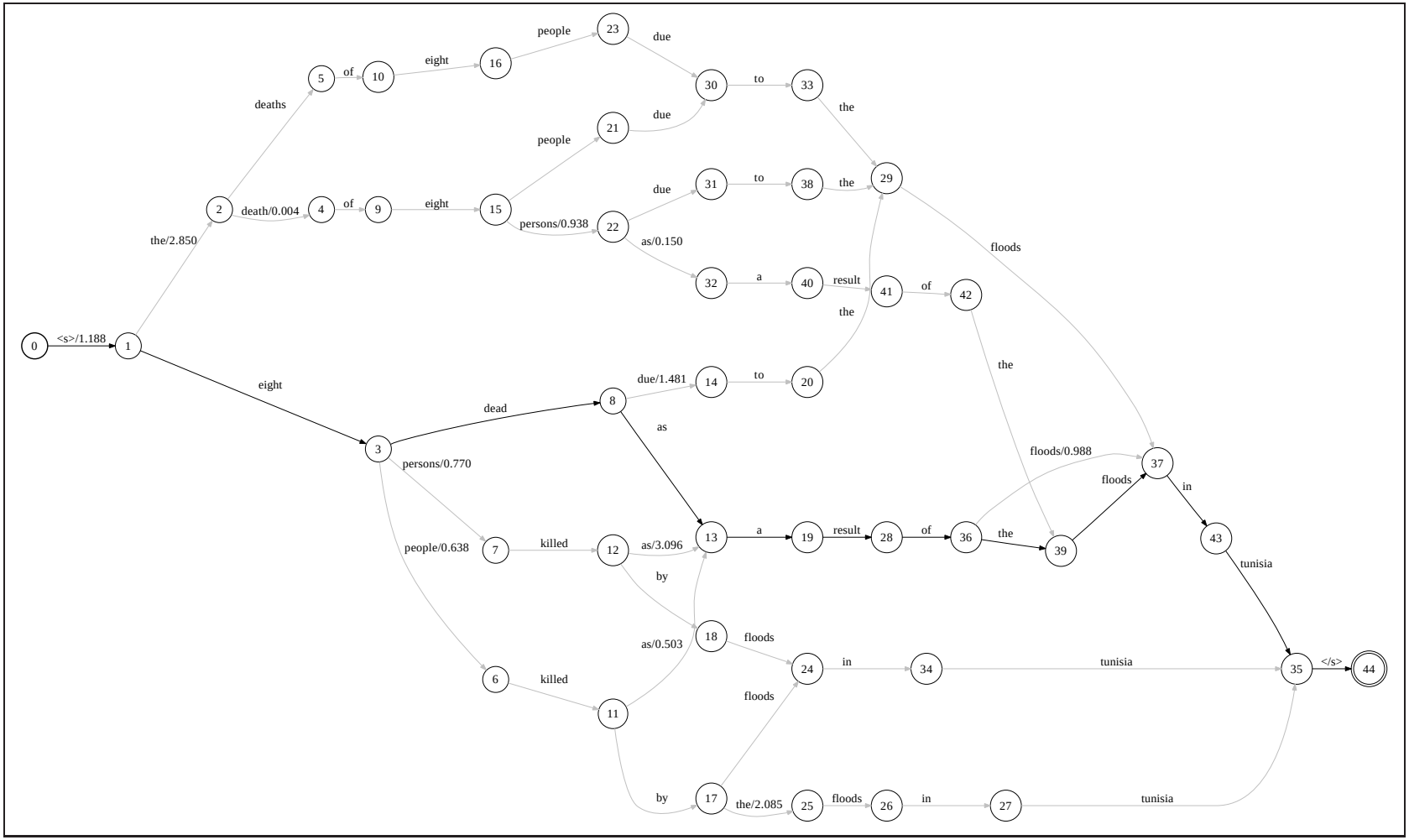


Figure 4.3: Optimised weighted word lattice example encoding multiple Arabic→English translation hypotheses. The maximum likelihood translation hypothesis $\hat{\mathbf{e}}$ is the sentence "*eight dead as a result of the floods in tunisia*" corresponding to the path through the lattice marked with bold transitions. This translation lattice encodes 14 distinct hypotheses. Weights are shown as tropical semiring negative log probabilities and the distribution has been normalised so that the probability of each path is $P(\mathbf{e}|\mathbf{f})$ and $\sum_{\mathbf{e}} P(\mathbf{e}|\mathbf{f}) = 1$.

Formally, a machine translation word lattice is a weighted directed acyclic graph (DAG) (Cormen et al., 2001). The sequence of state transitions on each complete path through the lattice from the initial state to a final state defines a translation hypothesis and its cost. The total cost of the hypothesis is obtained by aggregating the costs of the individual transitions that define the path. In the weighted finite-state acceptor representation of a word lattice, the cost is obtained as the generalised $\otimes$-product of individual transition costs.

One advantage of representing word lattices as WFSAs is that general purpose optimisation operations (Mohri et al., 2008) (Chapter 2, Section 2.4.1) exist to determinize and minimise the lattice for space efficiency. Transition and path weights are also easily manipulated: hypothesis scores can be converted to a normalised probability distribution or the weights redistributed optimally for efficient second-pass rescoring and search procedures.

Figure 4.3 on page 29 shows the WFSA representation of a lattice encoding multiple hypotheses generated by an Arabic→English machine translation decoder. The cost of each path $\mathbf{e}$ has been normalised so that the lattice defines the posterior distribution $P(\mathbf{e}|\mathbf{f})$, and the sum over all paths $\sum_{\mathbf{e}} P(\mathbf{e}|\mathbf{f}) = 1$. Costs in the figure are shown as tropical semiring negative log probabilities. The maximum likelihood translation hypothesis $\hat{\mathbf{e}}$ corresponds to the sequence of input labels on the transitions of the path marked in bold.

It may not always be possible to apply complex rescoring procedures to the full space of hypotheses encoded in a first-pass translation lattice. Likelihood pruning (Mohri, 2002) can be used to prune the most unlikely hypotheses from the lattice. Pruning is performed with respect to the shortest path in the lattice (i.e. the most likely translation) using a likelihood pruning threshold p. If the shortest path has cost c then any paths with cost greater than $c \otimes p$ are pruned from the lattice, where the greater than comparison is evaluated with respect to the natural semiring order (Allauzen et al., 2007).

4.2.3 Decoding with Weighted Finite-State Transducers

This section describes an alternative decoding architecture for SMT that exploits efficient general purpose operations and algorithms and avoids the need for a custom decoder implementation. The Transducer Translation Model (TTM) (Kumar et al., 2006) is a generative model of translation that applies a series of transformations specified by conditional probability distributions and encoded as weighted finite-state transducers (Mohri, 1997; Mohri et al., 2008). The TTM is based on the generative source-channel model of SMT (Brown et al., 1990) so in the following discussion the 'target language sentence' refers to the input sentence in the foreign language and the 'source language sentence' refers to the goal of decoding, i.e. the most likely English translation selected by the decoder.

In the TTM, the generation of a target language sentence $\mathbf{f} = f_1^J$ starts with the source language sentence $\mathbf{e} = e_1^I$ generated by the source language model distribution $P_G(e_1^I)$. Next, the source sentence is segmented into a series of K phrases according to the phrasal segmentation distribution $P_W(u_1^K, K|e_1^I)$. This distribution is the subject of Chapter 6. The phrase translation and reordering model $P_\Phi(v_1^R|u_1^K)$ generates the reordered sequence of target language phrases v_1^R (Kumar and Byrne, 2005). In order to avoid an exponential explosion in the size of the search space, reordering is usually limited to a window of one or two phrases. Finally, the reordered target language phrases are transformed to word sequences f_1^J under the target segmentation model $P_\Omega(f_1^J|v_1^R)$.

These component distributions together form a joint distribution over the source and target language sentences and their possible intermediate phrase sequence representations as

$P(f_1^J, v_1^R, u_1^K, e_1^I)$. The probability of generating foreign sentence f_1^J from source sentence e_1^I is found as the product of the probabilities of each component TTM model:

$$P(f_1^J|e_1^I) = \overset{f_1^J}{\underset{\Omega}{\longleftarrow}} P_\Omega(f_1^J|v_1^R) \times \overset{v_1^R}{\underset{\Phi}{\longleftarrow}} P_\Phi(v_1^R|u_1^K) \times \overset{u_1^K}{\underset{W}{\longleftarrow}} P_W(u_1^K|e_1^I) \times \overset{e_1^I}{\underset{G}{}} P_G(e_1^I)$$

Translation under the generative model starts with the target sentence f_1^J in the foreign language and searches for the best source sentence $\hat{e}_1^I$. The decision rule is as follows:

$$\hat{e}_i^I = \underset{e_1^I, u_1^K, v_1^R}{\operatorname{argmax}} P(f_1^J, v_1^R, u_1^K, e_1^I) \tag{4.10}$$

Encoding each conditional distribution as a WFST leads to a model of statistical machine translation as the series of weighted compositions

$$\mathcal{L} = G \circ W \circ \Phi \circ \Omega \circ F, \tag{4.11}$$

where F is an acceptor for the target language sentence and $\mathcal{L}$ is the word lattice of translations obtained by decoding. The maximum likelihood translation $\hat{e}_1^I$ according to the decision rule of Equation (4.10) is the path in $\mathcal{L}$ with least cost; this can be easily and efficiently found using the shortest distance algorithm in the tropical semiring (Mohri, 2002).

The use of compact word lattices in the TTM allows general purpose WFST decoding procedures to be applied to a very large space of alternative hypotheses, and for the direct generation of translation lattices. The large lattice rescoring methods described in Chapters 5, 6, 7, 8 and 9 show the importance and benefits of being able to generate a large space of high quality translations.

4.3 Hierarchical Phrase-Based Machine Translation

Hierarchical phrase-based machine translation is an induced form of syntax-based translation first introduced by Chiang (2005). The main goal is to combine the advantages of phrase-based SMT with the fundamental idea from syntax that natural language has a hierarchical structure. Just as for phrases in phrase-based SMT, the constituents of hierarchical phrase-based SMT do not necessarily have a linguistic motivation, although a translation must be learnable from parallel data. The main advantages of hierarchical models are that they capture the recursive relationship of natural language constituents and allow linguistic annotation through the use of categories for non-terminals in the grammar.

4.3.1 Context-Free Grammars and Chart Parsing

Context-free grammars, also known as phrase-structure grammars, define a formal language of strings and associated hierarchical structure in terms of non-overlapping constituents (Manning and Schutze, 1999; Jurafsky and Martin, 2008). The rules or productions of the grammar define a relation that specifies how the single (i.e. context-free) non-terminal left-hand side of a rule can be rewritten as a mixed string of non-terminals and terminals on the right-hand side. A series of rule applications that expands the start symbol to a sequence containing only terminal symbols represents a complete derivation or parse. Stochastic context-free grammars

assign a probability to each rule; these probabilities can be estimated from training data using the inside-outside algorithm (Lari and Young, 1990). The probability of a complete parse is the product of the probabilities of rules used in the derivation. Multiple derivations may yield the same string; the probability of a string is the sum of the probabilities of all derivations that yield the string.

Context-free grammars can be transformed into a convenient form such that the right-hand side of each rule consists of either two non-terminal symbols or a single terminal symbol. This process is known as binarization and the resulting grammar is said to be in Chomsky Normal Form (CNF). Binarization may require the introduction of additional intermediate rules, but the grammar still defines the same formal language as the original grammar. CNF grammars have a binary branching structure that can be parsed efficiently with simple algorithms.

The CYK algorithm (Jurafsky and Martin, 2008) is a bottom-up chart parsing algorithm for finding all derivations that yield a given input string. The parsing procedure is simple when applied to the binary branching structure of a Chomsky Normal Form grammar. The binary tree structure means that each non-terminal has exactly two child nodes and this can be conveniently represented as a two-dimensional matrix or chart (parsing of non-CNF grammars requires a higher-dimensional chart). Each chart cell contains the non-terminal symbols associated with a partial derivation yielding the words that span the positions $[x, y]$ of the input string. The chart is filled from left-to-right and bottom-to-top so that all child nodes that might be part of a derivation in a cell have already been parsed before the cell is visited. Each non-terminal maintains back-pointers to the chart entries from which it was derived. When all cells have been processed, the parse tree can be generated from the start node by following back-pointers to recursively retrieve embedded constituents.

4.3.2 Synchronous Context-Free Grammars

Hierarchical phrase-based machine translation is based on the theory of synchronous context-free grammars, also known as syntax-directed transduction grammars (Lewis and Stearns, 1968). Rules in a synchronous context-free grammar consist of a non-terminal left-hand side and two sequences of terminals and non-terminals on the right-hand side, one in the source language and one in the target language. Productions have the form

$$X \rightarrow \langle \gamma,\ \alpha,\ \sim \rangle, \tag{4.12}$$

where X is a non-terminal and $\gamma, \alpha \in (\{X\} \cup \mathbf{T})^+$ are sequences of non-terminals and terminals in the source or target language. The relation $\sim$ defines a bijective mapping, i.e. a one-to-one alignment of non-terminals in the source and target sides of the rule. The presence of non-terminals on the right-hand side of the rule gives the grammar its hierarchical structure. Rules are usually lexicalised so that the sequence of symbols γ and α each contain at least one terminal; this ensures that rules are only applied when there is supporting evidence in the form of the source and target lexical contexts. In addition to the regular rules of Equation (4.12), a synchronous context-free grammar includes 'simple' rules consisting only of terminals (analogous to phrase-pairs in phrase-based SMT), and special 'glue' rules $S \rightarrow \langle X, X \rangle$ and $S \rightarrow \langle S\ X, S\ X \rangle$ that allow monotone translation by direct concatenation of constituents.

Weighted synchronous context-free grammars specify a weight function $w(X \rightarrow \langle \gamma, \alpha \rangle)$ that distributes weight over the rules of the grammar. The weight of a complete derivation

$w(\mathcal{D})$ is obtained as the product of the weights of all rules used in the derivation:

$$w(\mathcal{D}) = \prod_{(X \to \langle \gamma, \alpha \rangle) \in \mathcal{D}} w(X \to \langle \gamma, \alpha \rangle) \tag{4.13}$$

Hierarchical rules are extracted from word-aligned parallel data using heuristics similar to those of the phrase extraction procedure in phrase-based SMT. Starting from a set of initial phrase-pairs, hierarchical rules are found by identifying phrases containing sub-phrases and replacing the sub-phrases with non-terminal symbols. Since search space complexity and decoding time depend on the number of hierarchical rules, the rule set is usually filtered (Chiang, 2005; Iglesias et al., 2009a) by restricting the number of source words covered by each rule application or constraining the number and positions of production non-terminals. This also reduces the problem of spurious ambiguity (Chiang, 2007). Maximum likelihood estimates of rule probabilities can be computed from the counts using relative frequency.

Hierarchical rules capture local context and reordering in a similar way to phrases in phrase-based SMT. For example, a Chinese→English hierarchical translation rule that allows reordering of its two non-terminal constituents X_1 and X_2 might have the following form:

$$X \to \langle X_1 \text{ de } X_2,\ X_2 \text{ of } X_1 \rangle \tag{4.14}$$

This rule encodes language-specific translation knowledge that the possessor is marked to the left in Chinese but to the right in English. Since the reordered arguments are specified as non-terminals, this rule has great generality and can even be used to reorder translations of constituent pairs not observed in the parallel data. Such generality of rule applications is one of the great strengths of hierarchical phrase-based machine translation.

4.3.3 Hierarchical Phrase-Based Decoding

Chiang (2005, 2007) implements hierarchical phrase-based translation decoding using k-best lists. The space of derivations that can be generated from even a modestly sized rule set is very large so hypotheses must be pruned during search. Chiang (2007) describes an algorithm, *cube pruning*, that can be used to efficiently find the k-best hypotheses in each cell of the CYK grid as a target language model is applied.

This section describes HiFST (Iglesias et al., 2009b; de Gispert et al., 2010), a lattice-based hierarchical decoder implemented using weighted finite-state transducers (Mohri, 1997; Mohri et al., 2008). Translation in HiFST is performed in two stages. In the first stage, the source language sentence is parsed according to a variant of the CYK algorithm (Chappelier and Rajman, 1998). In the second stage, the parse tree drives the generation of a target language word lattice containing all possible translations and derivations of the source sentence. The following description of HiFST is derived from the presentation in Iglesias et al. (2009b) and de Gispert et al. (2010).

Let $\mathcal{L}(N, x, y)$ denote the target language translation lattice associated with the source language sentence span s_x^{x+y-1} in cell (N, x, y) headed by non-terminal N. The goal of decoding is the lattice $\mathcal{L}(S, 1, J)$ at the top of the chart corresponding to complete translations of the full source sentence s_1^J. $\mathcal{L}(S, 1, J)$ is formed by the concatenation and union of sublattices in lower-level cells of the CYK grid according to back-pointers established during parsing.

Let $R(N, x, y)$ denote the set of rule indices used in cell (N, x, y). Each rule corresponds to a partial derivation headed by N and spans words s_x^{x+y-1} of the source sentence. The

lattice representing the application of rule $R^r : N \rightarrow \langle \gamma^r, \alpha^r \rangle / p^r$ is formed by concatenating lattices for each terminal and non-terminal element in the target language side of the rule $\alpha^r = \alpha^r_1, \ldots, \alpha^r_{|\alpha^r|}$. The lattice $\mathcal{L}(N, x, y, r)$ is built for each rule $r \in R(N, x, y)$ as follows:

$$\mathcal{L}(N, x, y, r) = \bigotimes_{i=1 \ldots |\alpha^r|} \mathcal{L}(N, x, y, r, i) \tag{4.15}$$

If α^r_i is a terminal $t \in \mathbf{T}$ then $\mathcal{L}(N, x, y, r, i)$ is a single-arc acceptor $\mathcal{A}(t)$ for target word t; if α^r_i is a non-terminal, then it refers to a lower-level lattice of partial translations in cell (N', x', y') identified by a back-pointer:

$$\mathcal{L}(N, x, y, r, i) = \begin{cases} \mathcal{A}(\alpha_i) & \text{if } \alpha_i \in \mathbf{T} \\ \mathcal{L}(N', x', y') & \text{otherwise} \end{cases} \tag{4.16}$$

The target language translation lattice for cell (N, x, y) is obtained as the union of lattices $\mathcal{L}(N, x, y, r)$ over all rules $r \in R(N, x, y)$ applied in the cell:

$$\mathcal{L}(N, x, y) = \bigoplus_{r \in R(N,x,y)} \mathcal{L}(N, x, y, r) \tag{4.17}$$

The probability of rule R^r is applied as cost $c^r = -\log p^r$ in the exit state of the lattice $\mathcal{L}(N, x, y, r)$. The concatenation and union of Equations (4.15) and (4.17) are applied using general purpose WFST operations. Non terminals in derivations are represented by special pointer arcs to lower-level lattices and only expanded to target language words when the language model is applied. This improves memory efficiency considerably.

The main advantage of hierarchical decoding with HiFST is the use of lattices instead of k-best lists. Chiang (2005) describes the problem of spurious ambiguity in which multiple derivations with the same target language translation can result in impoverished k-best lists with little variation. This lack of variation results from cube pruning with a fixed depth and can cause problems for minimum error rate training (Section 4.5) and subsequent rescoring procedures. The use of lattices in HiFST allows a much larger and richer space of partial translation hypotheses and derivations to be represented in each cell. As a result, less pruning is required during decoding and search errors are reduced.

Another advantage of HiFST is that the use of general purpose WFST operations results in a much simpler decoder architecture that supports the direct generation of target language translation lattices. This allows better integration with large lattice rescoring and transformation procedures such as the application of higher-order n-gram language models (Chapter 5), phrasal segmentation models (Chapter 6), lattice minimum Bayes-risk decoding (Chapter 7), multiple-lattice combination procedures (Chapter 8), and monolingual fluency constraints (Chapter 9). All of these large lattice rescoring methods benefit from the rich space of translation hypotheses encoded in HiFST lattices.

4.4 Machine Translation Evaluation Metrics

Machine translation quality is a difficult thing to measure since there are often several alternative and equally valid ways of translating a foreign sentence. Although human experts may be good at judging translation quality, they are expensive to employ, time consuming, and

can sometimes be inconsistent. For these reasons, a low-cost automatic means of scoring machine translation output is highly desirable. It can be argued that one of the main reasons for the rapid progress in statistical machine translation over the last ten years is the widespread acceptance and adoption of automatic metrics, particularly the BLEU score (Papineni et al., 2002b). This section describes the most commonly used metrics.

4.4.1 BLEU Score

The BLEU score is the most widely used automatic machine translation quality metric and is motivated by the view that "the closer a machine translation is to a professional human translation, the better it is" (Papineni et al., 2002b). How close system output is to a human translation can be evaluated by comparing it with a known set of good reference translations. BLEU is popular since it is quick, language-independent, correlates well with human judgements of quality, and accounts for the variation allowed in translation. The BLEU score is a measure of precision computed from a weighted average of variable-length, position-independent n-gram co-occurrences between the system output and one or more reference translations. Translations that match reference unigrams satisfy adequacy; higher-order matches account for fluency.

The precision at order n is the proportion of n-grams in the candidate translation that are matched in the references. The matched count of each n-gram u is first *clipped* by truncation to the maximum number of times it occurs in any of the individual references. Such count clipping ensures that precision is not artificially inflated by spurious repetition of high probability n-grams. For a single sentence S, the precision is computed by summing the matching clipped counts of each n-gram $u \in S$ and dividing by the total number of n-grams in S. For multi-sentence corpus C, the corpus-level modified n-gram precision p_n is obtained by summing over each sentence:

$$p_n = \frac{\sum_{S \in C} \sum_{u \in S} count_{matched}(u)}{\sum_{S \in C} \sum_{u \in S} count(u)} \tag{4.18}$$

Candidate translations should be similar in length to those of human translators. If the translation is too long then it will have poor precision. However, high precision could be obtained simply by producing shorter output. To compensate for this problem, the BLEU score includes an exponentially harsh multiplicative brevity penalty (BP) to ensure that high scoring translations cannot be much shorter than the references. The brevity penalty is computed from the total length of the corpus of candidate translations c, and the effective reference length r obtained by summing the *best match length* of each candidate translation:

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases} \tag{4.19}$$

The brevity penalty is multiplied by the weighted geometric mean of modified n-gram precisions p_n at each order $n = 1, \ldots, N$ to give the following definition of the BLEU score:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} \lambda_n \log p_n\right) \tag{4.20}$$

The weight parameters λ are positive and sum to 1. BLEU is normally evaluated using a maximum order of 4 and uniform weights so the definition in the log domain simplifies as

$$\log \text{BLEU} = \min(1 - \frac{r}{c}, 0) + \frac{1}{4}\sum_{n=1}^{4} \log p_n \tag{4.21}$$

BLEU scores range from 0 to 1 but even human translators will not attain perfect precision at the corpus-level unless they happen to match the references in both word choice and word order. BLEU score precisions are computed with respect to n-grams in the union of the references; since the precision increases when there are more references it is difficult to compare scores computed with respect to a different number of references.

Papineni et al. (2002a,b) show a strong correlation between the BLEU score and human assessments of translation quality. Doddington (2002) shows BLEU to be sensitive to differences between systems of similar quality, and that it ranks them consistently regardless of the selection of documents and references used for scoring. Burch et al. (2006) argue that the coarse model of variation in translation allowed by BLEU is such that there is no guarantee that higher scores reflect real improvements in translation quality. In particular, BLEU is unable to distinguish between content and function words, fails to match synonyms and paraphrases (unless they happen to be present in the references), and settles for a crude brevity penalty because of the difficulty of computing recall with respect to multiple references.

4.4.2 NIST Score

BLEU makes no distinction between n-grams of the same order when computing precisions, even though some words and phrases are obviously more important than others. This is particularly true when considering translation adequacy. The NIST score (Doddington, 2002) uses information weights to distinguish between informative and uninformative n-gram matches. These weights are computed for each n-gram $u = w_1 \ldots w_n$ using counts in the reference translations:

$$\text{Info}(u) = \log_2 \frac{c(w_1 \ldots w_{n-1})}{c(w_1 \ldots w_n)} \tag{4.22}$$

These information weights replace the clipped counts of the BLEU score. Let $\mathcal{N}_n$ denote the set of n-grams in the translation output and $\mathcal{R}_n$ the set of n-grams in the references. The NIST score is defined as

$$\text{NIST} = \sum_{n=1}^{N} \left\{ \frac{\sum_{u \in \{\mathcal{N}_n \cap \mathcal{R}_n\}} \text{Info}(u)}{|\mathcal{N}_n|} \right\} \cdot \exp\left\{ \beta \log_2 \left[\min\left(\frac{c}{r}, 1 \right) \right] \right\}, \tag{4.23}$$

where N specifies the maximum order (usually 5), c is the candidate translation length, r is the average reference length, and β controls the harshness of the brevity penalty. Note that the NIST brevity penalty differs from BLEU by using the average reference length instead of the closest reference length. Doddington (2002) shows NIST to have greater stability than BLEU and to correlate better with human assessments of adequacy for a range of different languages and test sets.

4.4.3 METEOR

METEOR (Metric for Evaluation of Translation with Explicit Ordering) (Banerjee and Lavie, 2005) compensates for weaknesses in BLEU by combining both precision and recall computed with respect to an explicit one-to-one word alignment of the system output with each available reference. One of its main advantages over BLEU is that it is valid at the segment level; BLEU is unreliable since it assigns segments a score of zero if any of the order-specific precisions are zero. METEOR has been shown to have higher correlation with human judges than BLEU and NIST for Arabic→English and Chinese→English translation (Lavie and Agarwal, 2007).

Word matching in METEOR is performed incrementally, starting from exact matches. Morphological variants using the Snowball stemmer[1] and synonyms via WordNet[2] are also allowed as matches. Each word-to-word alignment is scored as the product of the weighted harmonic mean of unigram precision P and unigram recall R,

$$F_{mean} = \frac{P \cdot R}{\alpha \cdot P + (1 - \alpha) \cdot R}, \tag{4.24}$$

with weight α set to emphasise recall. A tunable penalty ρ based on chunk fragmentation favours the grouping of adjacent words into longer spanning chunks. The score is then:

$$\text{METEOR} = (1 - \rho) \cdot F_{mean} \tag{4.25}$$

The score for each sentence is the highest of the scores computed with respect to each individual reference. METEOR has recently been extended to include support for the matching of paraphrases (Lavie and Denkowski, 2009).

4.4.4 TER - Translation Edit Rate

Translation edit rate (TER) (Snover et al., 2006) is the minimum number of edits required to modify a translation hypothesis such that it exactly matches one of the references. If there are multiple references, TER is the smallest number of edits to any one of the references, i.e. the score of the closest reference. A single edit is an insertion, deletion, substitution of a single word, or a shift of a contiguous word sequence to an alternate position in the hypothesis. The total number of edits is then normalised by the average length of the references so that for a single segment

$$\text{TER} = \frac{\#\text{ of edits}}{\text{average }\#\text{ of reference words}}. \tag{4.26}$$

Finding the minimum number of edits is NP-complete so dynamic programming with a greedy search is used to find the minimum number of edits relative to any one of the reference translations. The higher correlation with human judgements and reduced sensitivity to the number of references in TER is partly due to the low cost assigned to phrasal shifts. A shifted phrase in BLEU is much more harshly penalised since it results in a loss of precision over multiple n-grams and orders. TERP (TER Plus) (Snover et al., 2009) extends TER to incorporate edits based on morphology, synonymy, and probabilistically weighted paraphrases, with the cost of each type of edit optimised for maximum correlation with human judgements of adequacy and fluency.

[1] http://snowball.tartarus.org
[2] http://wordnet.princeton.edu

4.5 Minimum Error Rate Training

Discriminative training has been shown to provide significant improvements in a wide range of automatic speech recognition and natural language processing tasks. This section describes one popular application of discriminative training: the optimisation of feature weights in a log-linear model of statistical machine translation.

Och and Ney (2002) model the posterior probability of translation directly using maximum entropy (Section 4.1.4). With this model, for feature functions $h_m(\mathbf{e}, \mathbf{f})$ and feature weights λ_m, $m = 1, ..., M$, the direct translation probability is computed as a weighted log-linear combination of feature scores and feature weights:

$$P(\mathbf{e}|\mathbf{f}) = \frac{\exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f}))}{\sum_{\mathbf{e}'} \exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}', \mathbf{f}))} \tag{4.27}$$

Discriminative training can be used to find the feature weights $\hat{\lambda}_1^M$ that optimise the maximum class posterior probability criterion over a reference set of translations $\mathbf{S} = \{(\mathbf{e}_s, \mathbf{f}_s)\}$:

$$\hat{\lambda}_1^M = \underset{\lambda_1^M}{\operatorname{argmax}} \left\{ \sum_{s=1}^{S} \log P_{\lambda_1^M}(\mathbf{e}_s, \mathbf{f}_s) \right\} \tag{4.28}$$

Since the normalisation in Equation (4.27) is the same for each hypothesised translation it can be ignored during decoding. For feature weight optimisation, the normalisation is approximated by a large set of the most likely translations in the form of an k-best list.

The maximisation of Equation (4.28) is equivalent to maximising the likelihood of the direct translation model. However, since the correlation between maximum likelihood and real translation performance is not perfect, it is better to directly optimise the evaluation metric of interest, e.g. the BLEU score (Papineni et al., 2002b) or some other automatic measure of translation quality. Optimising the decoder feature weights for a specific error metric is known as *minimum error rate training* (MERT) (Och, 2003).

Let the number of errors in translation hypothesis $\mathbf{e}$ with respect to reference translation $\mathbf{r}$ be defined as $E(\mathbf{r}, \mathbf{e})$. The error count over all sentences in an evaluation corpus is

$$E(\mathbf{r}_1^S, \mathbf{e}_1^S) = \sum_{s=1}^{S} E(\mathbf{r}_s, \mathbf{e}_s) \tag{4.29}$$

The goal of minimum error rate training is to minimise the number of errors on a development corpus $\mathbf{f}_1^S$ given reference translations $\hat{\mathbf{e}}_1^S$ and K candidate translations $\mathbf{C}_s = \{\mathbf{e}_{s,1}, ..., \mathbf{e}_{s,K}\}$ for each input sentence $\mathbf{f}_s$. The optimal feature weights are those satisfying

$$\hat{\lambda}_1^M = \underset{\lambda_1^M}{\operatorname{argmin}} \left\{ \sum_{s=1}^{S} E(\mathbf{r}_s, \hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M)) \right\} \tag{4.30}$$

$$= \underset{\lambda_1^M}{\operatorname{argmin}} \left\{ \sum_{s=1}^{S} \sum_{k=1}^{K} E(\mathbf{r}_s, \mathbf{e}_{s,k}) \delta(\hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M), \mathbf{e}_{s,k}) \right\}, \tag{4.31}$$

where $\hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M)$ is the highest probability translation in $\mathbf{C}_s$ according to the log-linear model of Equation (4.27) given the foreign sentence $\mathbf{f}_s$ and feature weights λ_1^M:

$$\hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M) = \underset{\mathbf{e} \in \mathbf{C}_s}{\operatorname{argmax}} P(\mathbf{e}|\mathbf{f}_s) \tag{4.32}$$

The error function of Equation (4.30) contains an argmax operation and has many local optima so gradient descent cannot be used. In Och (2003), a smoothed error count amenable to gradient-based optimisation is defined, but it is shown to perform no better than the unsmoothed error count.

The standard way of optimising the unsmoothed error criterion is to use Powell's algorithm with a grid-based line search (Press et al., 2002). However, it is difficult to balance both speed and performance since the fine-grained grid and many random restarts required to find the true optimum with high probability is expensive to evaluate. In Och (2003), an algorithm for efficient optimisation of the unsmoothed error count that is guaranteed to find the optimal solution with greater speed and stability is proposed. The algorithm works by computing an ordered sequence of intervals $\gamma_1^{\mathbf{f}} < \gamma_2^{\mathbf{f}} < \ldots < \gamma_{N_{\mathbf{f}}}^{\mathbf{f}}$ along a line $\lambda_1^M + \gamma \cdot d_1^M$ in parameter space for each input sentence $\mathbf{f}$ in the corpus together with the change in error count $\Delta E_1^{\mathbf{f}}, \Delta E_2^{\mathbf{f}}, \ldots, \Delta E_{N_{\mathbf{f}}}^{\mathbf{f}}$ relative to the previous interval. The interval boundaries define the values of γ at which a different candidate becomes the most likely translation, so the error function only needs to be evaluated once per interval. The sentence-specific interval boundaries and changes in error for each sentence are merged to obtain sequences over the whole corpus. The optimal γ can then be efficiently computed by traversing the relatively small number of boundaries while updating error counts. Since the parameters λ_1^M are optimised with respect to a k-best list, they might be biased. To avoid such a bias, the parameters are iteratively re-estimated until convergence, with merging of k-best lists at each iteration to ensure the error rate cannot increase.

Och (2003) investigates the effect of minimum error rate training on the TIDES 2002 Chinese→English machine translation task[1] using a variety of different error criteria. The conclusion is that the metric chosen for MERT is usually the best performing metric in evaluation. The optimised parameters have a clear bias towards the metric used in the error criterion. In Och et al. (2004), minimum error training under the BLEU score is applied to the task of re-ranking k-best list with a large number of feature functions. The use of k-best lists permits a much larger and richer set of features including shallow syntactic functions and tree-based feature functions.

[1] http://projects.ldc.upenn.edu/TIDES/index.html

Chapter 5

Large Language Model Lattice Rescoring for Statistical Machine Translation

Statistical machine translation makes use of n-gram language models to guide the decoder search procedure and to select the most fluent target language hypothesis from the large space of possible translations of the source language sentence. Large monolingual corpora are available for training the parameters of a statistical language model. The main challenge is in effectively exploiting all of the available data.

The language model lattice rescoring framework described in this chapter serves as the baseline for more sophisticated lattice rescoring methods presented in Chapters 6, 7, 8 and 9. Section 5.1 motivates the use of higher-order language models and large monolingual corpora in second-pass lattice rescoring. Sentence-specific counts extraction and language model parameter estimation procedures are described in Section 5.2. Section 5.3 defines the machine translation lattice rescoring decision rule. An empirical study of Arabic→English and Chinese→English lattice rescoring using 5-gram and 6-gram language models estimated over approximately ten billion words of data is presented in Section 5.4. French→English and Spanish→English lattice rescoring experiments are also briefly reported. A summary and conclusions are presented in Section 5.5.

5.1 Introduction and Motivation

Increasing quantities of language model training data continue to improve SMT performance (Brants et al., 2007). It is therefore important to use all of the available training data whenever possible. However, large monolingual corpora and vocabularies create two problems for SMT language models: parameter estimation and translation decoding.

Smoothing methods such as Kneser-Ney (Kneser and Ney, 1995) require large amounts of memory during parameter estimation since it is necessary to compute the continuation counts for each history as well as the regular n-gram counts. Even supposing the model can be estimated, it may be difficult to apply in decoding since there are too many parameters to load into memory. Distributed language models (Zhang et al., 2006; Emami et al., 2007; Brants et al., 2007) (Chapter 3, Section 3.3) are one possible solution to this problem. An alternative approach is to perform first-pass translation with a lower-order language model in order to generate large word lattices (Chapter 4, Section 4.2.2) for subsequent rescoring with more powerful higher-order language models. Lattice rescoring considerably simplifies the problem of estimating and applying higher-order models. It is not necessary to load the full LM into memory since sentence-specific language models can be constructed for each individual lattice. These sentence-specific language models contain only the subset of n-gram probabilities required to rescore the hypotheses contained in the lattice.

5.2 Large Language Model Estimation

This section describes the counting and estimation procedures for second-pass language model rescoring. The goal is to be able to efficiently build powerful, higher-order language models over large corpora that can be used for offline rescoring of first-pass translation lattices. The techniques described in this chapter are needed to provide strong baselines for the subsequent investigations into the use of monolingual data.

5.2.1 Counts Extraction

Counts of relevant n-grams are first extracted from the monolingual corpus. This process is driven by the task-specific language model vocabulary formed from the list of all words in the target language side of the parallel text. Counts are extracted by dividing the training data into chunks and extracting substrings consisting only of words in the LM vocabulary with length less than or equal to the maximum n-gram order. Each chunk is processed in parallel and then merged to form a single task-specific counts file that can be used to rescore any set of lattices generated by the first-pass SMT system. Processing the monolingual corpus in chunks ensures that even higher-order LMs have relatively low memory requirements when extracting counts. Depending on the quantity of available training data, the counts extraction process can be slow but it only needs to be performed once per language pair.

5.2.2 Counts Filtering

The task-specific n-gram counts file is filtered to obtain the sentence-specific subset of counts required to rescore each individual lattice. The n-grams in each first-pass translation lattice can be extracted using a WFST counting transducer (Allauzen et al., 2003). An example

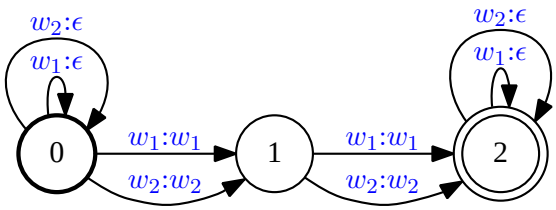


Figure 5.1: Example counting transducer for extracting bigrams from a lattice with vocabulary $\Sigma = \{w_1, w_2\}$. The bigrams are extracted by composing the transducer with the lattice.

```
FILTER-COUNTS(C, S, L_1 ... L_S, n)
1  for s ← 1 ... S
2      do N_s ← EXTRACT-NGRAMS(L_s, n)
3         for each u ∈ N_s
4             do R[u] ← {R[u] ∪ s}
5  for each (u, c) ∈ C : |u| ≤ n
6      do for each s ∈ R[u]
7             do WRITE-COUNT(C_s, (u, c))
```

Figure 5.2: Stream-based counts filtering algorithm for second-pass LM rescoring.

bigram counting transducer for the vocabulary $\Sigma = \{w_1, w_2\}$ is shown in Figure 5.1. Lattice n-grams are extracted by composing the counting transducer for order n with the lattice, projecting on the output, removing ϵ-arcs, determinizing and minimising. The resulting acceptor compactly encodes the lattice n-grams of order n. This process is repeated for each order; the resulting acceptors are then unioned to obtain the full set of lattice n-grams.

Stupid backoff smoothing (Brants et al., 2007) (Chapter 3, Section 3.3.2) uses context-independent backoff weights $\gamma(n)$ that depend only on the n-gram order. This allows the probabilities required to rescore each lattice to be estimated using only the counts of n-grams found in the lattice. Lists of lattice n-grams can be used to filter the task-specific counts file into subsets in order to estimate sentence-specific language models. The filtering process can be performed with low memory requirements using the efficient streaming algorithm shown in Figure 5.2. The input parameters are the stream of counts $\mathcal{C}$, testset size S, first-pass translation lattices $\mathcal{L}_1 \ldots \mathcal{L}_S$, and maximum n-gram order n. The algorithm generates for each sentence s the subset $\mathcal{C}_s \subseteq \mathcal{C}$ of n-gram counts (orders $1 \ldots n$) required to rescore the first-pass lattice $\mathcal{L}_s$. Firstly, the n-grams $\mathcal{N}_s$ in each lattice $\mathcal{L}_s$ are extracted using counting transducers (line 2). These n-grams are used to initialise a relevancy list $\mathcal{R}$ (lines 3 to 4) that indicates for each n-gram u the set of lattices $\mathcal{R}[u]$ containing the n-gram. As each n-gram and count pair $(u,\ c)$ is read from the task-specific counts file (line 5), a single hash lookup (line 6) identifies the list of lattices $\mathcal{R}[u]$ containing the n-gram; the count is then written to the sentence-specific counts file $\mathcal{C}_s$ of each relevant sentence (line 7). This algorithm has very low memory requirements since only the relevancy list $\mathcal{R}$ needs to be stored in memory while filtering the task-specific counts file.

5.2.3 Parameter Estimation

Stupid backoff language models (Brants et al., 2007) (Chapter 3, Section 3.3.2) use maximum likelihood probability estimates for observed n-grams and back off recursively to lower-orders for unobserved n-grams:

$$S(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})} & \text{if } c(w_{i-n+1}^{i}) > 0 \\ \gamma(n)S(w_i|w_{i-n+2}^{i-1}) & \text{otherwise} \end{cases} \tag{5.1}$$

The backoff weight $\gamma(n)$ depends only on the order n and is independent of the n-gram context. The recursion in Equation (5.1) ends with the definition for unigrams

$$S(w_i) = \frac{c(w_i)}{\sum_j c(w_j)} = \frac{c(w_i)}{N}, \tag{5.2}$$

where N is the total number of tokens in the training corpus. This simple form of backoff allows n-gram probabilities for each lattice to be computed directly from the filtered sentence-specific counts. The only statistic not available in the filtered counts is the total number of unigrams N. This has the same value for each sentence-specific LM and only needs to be computed once per language pair from the task-specific counts file.

The sentence-specific filtered counts are used to estimate the parameters of the second pass language model in accordance with Equations (5.1) and (5.2). The n-gram probabilities are encoded in the form of a weighted finite-state acceptor language model (Allauzen et al., 2003) (Chapter 3, Section 3.4). Each backoff arc has order-specific cost $-\log\gamma(n)$. Hypotheses are rescored by composing the language model acceptor with the lattice.

5.2.3.1 Out-of-Vocabulary Words

There may be some words in the testset that did not occur in the parallel data and are therefore not part of the language model vocabulary. These out-of-vocabulary (OOV) words are normally passed through the first-pass translation system unmodified. If they are not included in the WFSA representation of the language model, then paths on which they occur will be deleted when the language model is composed with the lattice in second-pass rescoring. The out-of-vocabulary words are therefore assigned a small probability. The denominator in Equation 5.2 is incremented by 1 and the additional probability mass $\frac{1}{N+1}$ shared equally amongst the out-of-vocabulary words.

5.3 Large Language Model Lattice Rescoring

The lattice rescoring experiments in Section 5.4 will show that optimal performance is obtained by rescoring lattices according to a combination of first-pass and second-pass language model scores. The language model rescoring decision rule is

$$\hat{E} = \underset{E\in\mathcal{E}}{\text{argmax}}\left\{ P(F|E) \times P_{LM_1}(E)^{\alpha_1} \times P_{LM_2}(E)^{\alpha_2} \times \beta^{|E|} \right\} \tag{5.3}$$

where $P(F|E)$ is the translation model probability, $P_{LM_1}(E)$ is the first-pass language model probability, $P_{LM_2}(E)$ is the second-pass language model probability, and β is a fixed word

AR→EN Testset	Genre	Sentences	Length	OOVs (%)
tune.text.nw	Newswire	2963	101317	0.22
test.text.nw		2242	80679	0.28
tune.text.web	Web	4589	139955	0.41
test.text.web		2697	86358	0.23

Table 5.1: Arabic→English testsets, average tokenised reference length, and average out-of-vocabulary word rate (%). Each testset sentence has either 1 or 4 reference(s).

ZH→EN Testset	Genre	Sentences	Length	OOVs (%)
tune.text.nw	Newswire	3085	105,375	0.38
test.text.nw		3001	102,632	0.41
tune.text.web	Web	4221	119,640	0.25
test.text.web		4285	117,883	0.25

Table 5.2: Chinese→English testsets, average tokenised reference length, and average out-of-vocabulary word rate (%). Each testset sentence has either 1 or 4 reference(s).

penalty that can be tuned to adjust the length of the output. The search is carried out over the full space of translation hypotheses encoded in the first-pass lattice $\mathcal{E}$. The exponential scaling parameters α_1 and α_2 smooth the first- and second-pass language model distributions; when $\alpha_1 = 0$, only the second-pass language model score influences the decision rule. The word penalty β can be used to tune the translation output length. These three parameters are optimised for BLEU score (Papineni et al., 2002b) on the tuning set. The translation model $P(F|E)$ is the weighted log-linear sum of first-pass feature scores (see Chapter 4, Section 4.1.4), excluding the contribution of the first-pass LM.

The second-pass lattice rescoring procedures and decision rule of Equation (5.3) are applied as operations on weighted finite-state acceptors (Mohri et al., 2008), implemented using OpenFst (Allauzen et al., 2007). In order to ensure that weighted composition applies the exact LM score to hypotheses in the lattice, failure transitions (Allauzen et al., 2003) are used to encode backoff arcs in the WFSA representation of the second-pass LM. This is particularly important for stupid backoff smoothing (Brants et al., 2007).

5.4 Large Language Model Rescoring Experiments

The following experiments investigate the use of higher-order 5-gram and 6-gram language models estimated over multi-billion token corpora for rescoring large-scale statistical machine translation lattices. Arabic→English and Chinese→English rescoring experiments are carried out within the framework of the GALE P4 evaluation. Language model rescoring results are also reported for French→English and Spanish→English translation lattices, based on the CUED submission to the WMT 2010 constrained data track translation task.[1]

The language model vocabulary is defined as the list of all words in the target language side of the parallel text. The size of the LM vocabulary is 476,346 words for Arabic→English translation and 417,410 words for Chinese→English translation. A large proportion of the words in these vocabularies occur only once in the parallel text.

[1] http://www.statmt.org/wmt10

Corpus	# Lines	# Tokens	OOVs (%)		Epoch
			AR	ZH	
Yemen Times	256,787	6,389,176	0.68	1.07	2006/02 – 2009/06
PBS	342,811	8,383,761	0.19	0.25	2003/10 – 2009/07
Ahram	832,855	22,322,749	0.35	0.66	1998 – 2009/04
Chinese FBIS	278,470	9,607,434	0.35	0.23	2003 – 2006
Arabic FBIS	749,434	19,670,578	0.53	0.98	2003 – 2006
Chinese OSC	246,189	8,387,221	0.23	0.09	2006/01 – 2009/03
Arabic OSC	347,529	8,643,053	0.24	0.63	2006/01 – 2009/03
Gulf News	3,091,303	66,788,604	1.77	2.08	2001 – 2009/06
Arab News	3,488,291	77,734,765	1.87	2.22	2001 – 2009/05
CU CNN	3,409,649	71,179,681	0.48	0.42	2006/01 – 2007/04
Taipei Times	4,783,482	112,738,807	0.78	0.75	2000 – 2009/06
CNN	5,659,748	118,739,780	0.49	0.41	2007/08 – 2009/06
People Daily	6,522,560	138,011,229	0.52	0.51	2000 – 2009/06
India Times	9,772,853	222,460,470	1.77	2.10	2001 – 2009/06
BBC	15,910,460	307,273,975	0.71	0.85	1999 – 2008/01
The Hindu	17,587,474	379,443,315	3.27	3.81	2000 – 2009/06
GigaWord v4 CNA	1,241,023	35,917,445	0.66	0.46	1994 – 2008
GigaWord v4 LTW	11,562,054	299,721,395	0.64	0.79	1994 – 2008
GigaWord v4 XIN	14,241,922	358,147,081	0.81	0.90	1994 – 2008
GigaWord v4 AFP	27,241,209	725,927,421	0.68	0.90	1994 – 2008
GigaWord v4 APW	52,580,272	1,292,445,558	1.19	1.35	1994 – 2008
GigaWord v4 NYT	65,996,636	1,653,075,147	0.99	1.14	1994 – 2008
LDC webdata	54,607,175	989,530,139	3.99	4.14	2005/01 – 2008/02
GoogleNews	104,262,520	2,527,448,142	0.70	0.87	2006/02 – 2009/02
Total	405,012,706	9,459,986,926	1.30	1.47	1994 – 2009

Table 5.3: Tokenised English LM training corpora, number of lines, number of tokens, epoch, and OOV rate (%) for Arabic→English and Chinese→English translation tasks.

The total number of sentences, average tokenised reference length, and average out-of-vocabulary (OOV) rate (%) for the Arabic→English and Chinese→English testsets are summarised in Tables 5.1 and 5.2. There are two newswire and two web data testsets for each language pair, and each foreign sentence has either 1 or 4 reference translations. The Chinese→English test.text.nw and test.text.web testsets correspond to the GALE P4 system combination sets SysCombTune.text.nw and SysCombTune.text.web. The large language model vocabulary sizes ensure that the testset OOV rate is low for both language pairs.

5.4.1 Language Model Training Data

Table 5.3 shows the monolingual training corpora used to estimate the parameters of the Arabic→English and Chinese→English second-pass 5-gram and 6-gram language models. There is a total of nearly 9.5 billion words of monolingual data available. The largest single corpus, containing over 4 billion tokens, is the GigaWord Fourth Edition (Parker et al., 2009). The English side of the parallel texts (230M tokens for Arabic→English and 260M tokens for Chinese→English) are added to the monolingual data, resulting in a total of approximately 10

r	1g	2g	3g	4g	5g	6g
1	11.29	42.74	55.98	64.91	70.87	74.59
2	8.60	15.89	16.42	15.98	15.21	14.50
3	4.10	7.82	7.00	5.95	5.01	4.34
4	3.30	5.03	4.19	3.33	2.65	2.19
5	2.30	3.39	2.63	1.93	1.43	1.11
6	2.01	2.58	1.93	1.38	1.00	0.77
7	1.62	1.97	1.40	0.95	0.65	0.48
8	1.45	1.60	1.11	0.73	0.49	0.35
9	1.29	1.31	0.87	0.56	0.36	0.25
10+	64.04	17.67	8.46	4.27	2.33	1.42

Table 5.4: Proportion of n-grams (%) at each order $n = 1 \ldots 6$ with the specified count-of-counts frequency r for the Arabic→English language model vocabulary.

LM	1g	2g	3g	4g	5g	6g	Total
AR→EN	0.476	112	859	2,283	3,592	4,352	11,198
ZH→EN	0.417	108	846	2,258	3,558	4,311	11,081

Table 5.5: Effective number of n-gram language model parameters ($\times 10^6$) by order for the Arabic→English and Chinese→English language model vocabularies.

billion tokens. The table shows for each corpus the OOV rate computed with respect to the Arabic→English and Chinese→English LM vocabularies. The OOV rate is generally low: less than 1% for most LM corpora. The 'LDC webdata' and 'The Hindu' corpora have the highest OOV rates. The monolingual training data covers a period of almost 15 years; the training data blackout epochs for non-LDC-released corpora are November and December 2006, June 2007, and June 2008, corresponding to the GALE P2, P3, and P4 evaluation periods.

The proportion of n-grams (%) at each order $n = 1 \ldots 6$ with counts-of-counts frequency r for counts extracted using the Arabic→English vocabulary is shown in Table 5.4. The majority of higher-order parameters in a zero-cutoff n-gram language model consist of n-gram sequences that were observed only once in the training data. Estimates of their probabilities are unlikely to be reliable. In machine translation, however, just knowing that the sequence of words occurred in (presumably) fluent target language text is useful information that can aid the selection of fluent target hypotheses from a translation lattice. The effect of singleton counts in higher-order second-pass language model rescoring is evaluated in Section 5.4.3.4.

The number of model parameters in a zero-cutoff n-gram language model is equal to the number of distinct n-gram counts. The estimation procedure described in Section 5.2 constructs individual sentence-specific language models containing only the subset of parameters required to rescore a single lattice; the majority of these parameters are thus never instantiated. The effective number of model parameters ($\times 10^6$) at each order $n = 1 \ldots 6$ is shown in Table 5.5. A total of over 11 billion n-grams are extracted from the monolingual data.

5.4.2 System Development and Lattice Generation

The Arabic→English and Chinese→English first-pass translation lattices are generated as follows. Hierarchical rules are extracted from the aligned parallel text using the constraints

described in Chiang (2007) and the rule count and pattern filters of Iglesias et al. (2009a). First-pass translation decoding with HiFST (Iglesias et al., 2009b) (Section 4.3.3) produces word lattices encoding large numbers of alternative translations. For both language pairs, minimum error rate training (Och, 2003) under the BLEU score (Papineni et al., 2002b) optimises the feature weights of the decoder with respect to the tuning set of each language pair. Feature weights are normalised with respect to a fixed LM weight of 1. The first-pass lattice topologies are optimised (Mohri et al., 2008) prior to 5-gram and 6-gram rescoring.

The English language model used during first-pass decoding is a modified Kneser-Ney (Kneser and Ney, 1995) smoothed 4-gram estimated over the English side of the parallel text and the AFP and XIN subsets of the English GigaWord Third Edition (Graff et al., 2007) (Arabic→English) or Fourth Edition (Parker et al., 2009) (Chinese→English). The first-pass LM is estimated using SRILM (Stolcke, 2002) with default cutoffs: all unigrams and bigrams are retained, but higher-order n-grams with a count of one are discarded. These cutoffs are required in order to allow for efficient estimation of first-pass LM parameters in memory.

Second-pass language model counts are extracted for each of the corpora in Table 5.3 using SRILM, and then merged to form a single large counts file for each language pair. The compressed counts file requires about 25GB of disk space and includes more than 11 billion n-gram counts. The stream-based counts extraction and filtering procedures described in Sections 5.2.1 and 5.2.2 identify the subset of counts relevant to each sentence in the testset. Sentence-specific, zero-cutoff, stupid-backoff 5-gram and 6-gram language models are then estimated from these counts in accordance with Equation (5.1).

The exponential scaling factors α_1 and α_2, and word penalty β in the rescoring decision rule of Equation (5.3) are optimised with respect to the tune.text.nw and tune.text.web tuning sets of each language pair. A lattice rescoring decoder implemented using OpenFst (Allauzen et al., 2007) applies the decision rule and performs a grid-based search over a specified range of parameter values. The optimised parameters are then applied to rescore the test.text.nw and test.text.web testset lattices.

5.4.2.1 Lattice Hypothesis Space Size

This section compares the size of the lattice hypothesis space for each language pair and genre. Since it is difficult to measure the number of hypotheses in very large lattices, the number of unique n-grams is compared. Figure 5.3 plots the total number of lattice n-grams (orders $n = 1 \ldots 6$) as a function of expected sentence length for the Arabic→English (top) and Chinese→English (bottom) newswire and web data tuning sets.

Comparing genres shows that for Arabic→English lattices, the web data testset contains many longer sentences than the newswire testset; these longer sentences have more n-grams, as expected. The difference between genres is less pronounced for Chinese→English translation. Comparing languages shows that Chinese→English lattices have nearly an order of magnitude more n-grams than Arabic→English lattices (see the vertical axis scales). The difference in the number of lattice n-grams is a result of the different grammars used during first-pass translation decoding. Arabic→English translation is performed with a Shallow-1 grammar (de Gispert et al., 2010) that allows only a single level of non-terminal rule nesting. By contrast, Chinese→English translation is performed with a fully hierarchical grammar which results in a far larger and more varied space of translation hypotheses. Unfortunately, as will be demonstrated by the n-gram coverage experiments in Section 5.4.3.1, a large proportion of the hypotheses in the Chinese→English lattices are of fairly low quality.

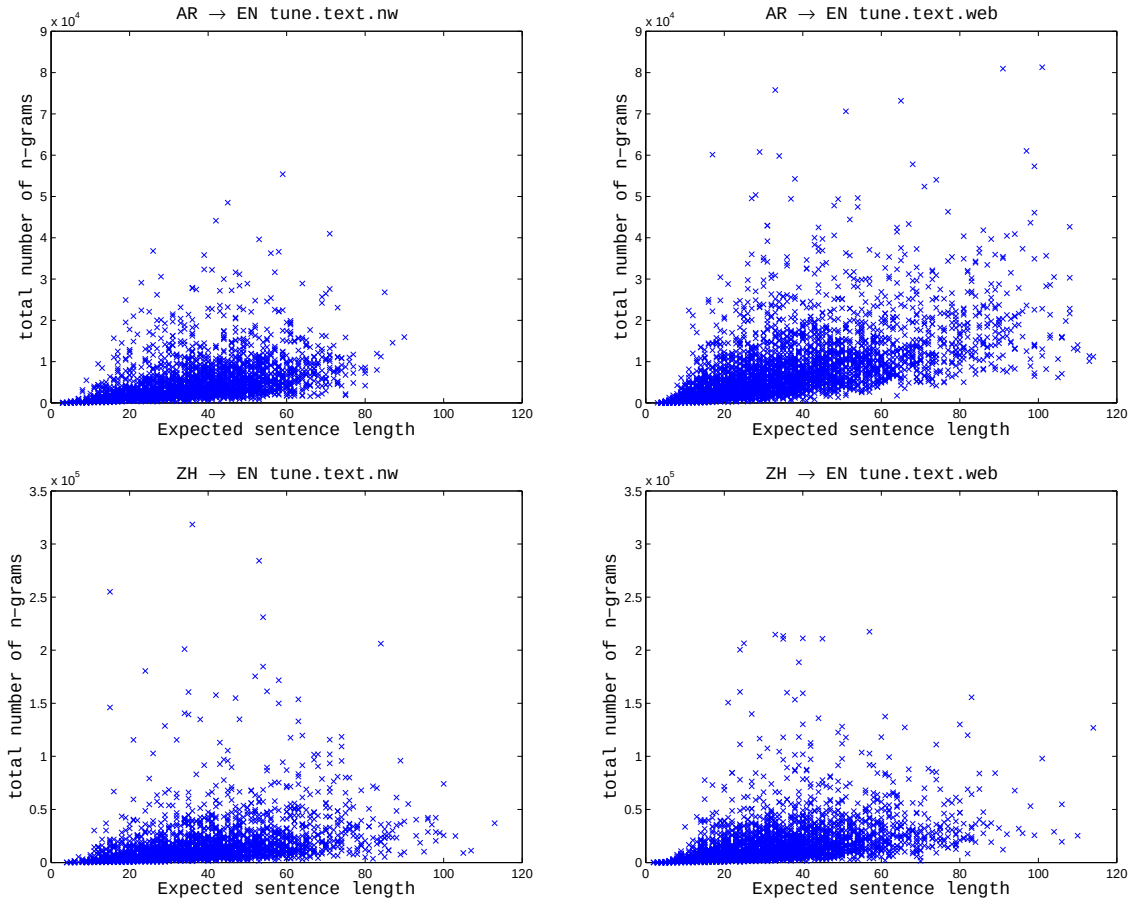


Figure 5.3: Number of lattice n-grams (orders $n = 1 \ldots 6$) as a function of expected sentence length for Arabic→English and Chinese→English newswire and web data tuning sets ($p = 7$).

5.4.3 Language Model Rescoring Results and Analysis

Rescoring results for Arabic→English and Chinese→English translation lattices are shown in Tables 5.6 and 5.7. The tables show the BLEU score and brevity penalty for first-pass maximum likelihood (ML) newswire and web data translations, and the scores obtained after rescoring the first-pass lattices with 5-gram and 6-gram second-pass language models. The first-pass lattices were generated using a likelihood pruning threshold of $p = 7$ (Chapter 4, Section 4.2.2). In these experiments, a single context-independent back off weight γ is used for all orders. The first-pass LM scale α_1 is fixed at 0.5 and the second-pass LM scale α_2 and word penalty β are optimised on the tune.text.nw and tune.text.web testsets. The optimised 5-gram and 6-gram rescoring parameters were $\alpha_2 = 0.5$ and $\beta = 0.0$.

Arabic→English lattice rescoring with the 5-gram second-pass LM gives good gains. The BLEU score is improved by +1.6 on the test.text.nw testset and +1.3 on test.text.web. The gains on the tuning sets tune.text.nw and tune.text.web are smaller: +1.1 for newswire data and +0.8 for web data. The gains on the tuning sets are smaller because the first-pass decoder has been optimised for these sets during minimum error rate training.

AR→EN	Newswire Data				Web Data			
	tune.nw		test.nw		tune.web		test.web	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
HiFST	45.9	1.000	45.0	0.995	25.2	0.998	33.1	1.000
+5-gram	47.0	1.000	46.6	0.992	26.0	0.998	34.4	1.000
+6-gram	46.7	0.994	46.5	0.987	25.9	0.986	34.7	1.000

Table 5.6: Arabic→English 5-gram and 6-gram rescoring results using order-independent backoff weight $\gamma = 0.4$, scaling factors $\alpha_1 = \alpha_2 = 0.5$, and word penalty $\beta = 0.0$.

ZH→EN	Newswire Data				Web Data			
	tune.nw		test.nw		tune.web		test.web	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
HiFST	27.7	0.999	28.1	0.999	15.8	0.994	15.2	0.993
+5-gram	28.2	0.993	28.8	0.998	16.2	0.992	15.9	0.989
+6-gram	28.3	0.998	29.0	1.000	16.1	0.992	15.9	0.989

Table 5.7: Chinese→English 5-gram and 6-gram rescoring results using order-independent backoff weight $\gamma = 0.4$, scaling factors $\alpha_1 = \alpha_2 = 0.5$, and word penalty $\beta = 0.0$.

Configuration		newstest2008	newstest2009	newstest2010
FR→EN	HiFST	24.7	28.4	28.5
	+5-gram	24.9	28.6	29.0
ES→EN	HiFST	24.6	26.0	29.1
	+5-gram	25.2	26.8	30.1

Table 5.8: French→English and Spanish→English 5-gram second-pass language model lattice rescoring BLEU scores for the WMT 2010 constrained data track evaluation.

Chinese→English rescoring gains are smaller than the gains for Arabic→English: +0.7 for both test.text.nw and test.text.web. The ML baseline BLEU scores show that the quality of hypotheses in the Chinese→English lattices is much lower than those in the Arabic→English lattices. This makes it harder for the LM to separate the good hypotheses from the bad.

Comparing 5-gram and 6-gram rescoring results shows that with these parameter settings the 6-gram models do not perform as well as the 5-gram models for Arabic→English rescoring, except for test.text.web where there is a small gain of +0.3 BLEU. The 6-gram models perform slightly better in Chinese→English rescoring. Section 5.4.3.2 will show that the reason for this mixed performance is that the backoff weight γ is set inappropriately.

Table 5.8 summarises French→English and Spanish→English 5-gram rescoring results from the CUED submission to the WMT 2010 constrained data track evaluation. Gains from French→English 5-gram rescoring are smaller than for rescoring of Arabic→English and Chinese→English lattices. One reason for this is that the WMT testsets are scored with respect to only a single reference. The amount of training data used to train the second-pass LM is also around half as much as was used to train the Arabic→English and Chinese→English second-pass LMs. Gains are a little larger for Spanish→English lattice rescoring.

5.4.3.1 Lattice and Reference Coverage Statistics

Stupid backoff language model scores do not define a normalised probability distribution so it is not possible to compute the language model perplexity. An alternative metric that is sometimes used as a broad indicator of LM quality and relevance (Brants et al., 2007) is the coverage of testset n-grams in the training corpus. Intuitively, a language model that has previously seen many of the higher-order testset n-grams is likely to assign better scores to hypotheses than a model with low testset coverage.

Reference N-gram Coverage Tables 5.9 and 5.10 show training data coverage of reference n-grams at orders $n = 1 \ldots 6$ for the Arabic→English and Chinese→English testsets. Coverage is computed at the testset level; the reference n-grams for each testset consist of all n-grams in the union of the reference translations. Good coverage of unigrams, bigrams and trigrams is observed, but higher-order n-gram coverage falls off rapidly. 6-gram coverage is particularly low, especially for the web data testsets where coverage falls to around 12% in both language pairs. The low coverage suggests a mismatch between training data and testset data: the web data testsets consist mainly of newsgroup postings and blogs written in an informal conversational style, while much of the language model training data is from the newswire genre. These coverage statistics partly explain the relatively poor performance of 5-gram and 6-gram rescoring of web data translation lattices in Tables 5.6 and 5.7.

Lattice N-gram Coverage Tables 5.11 and 5.12 show coverage statistics for lattice n-grams. Lattice n-gram coverage is computed with respect to the set of all n-grams in the translation lattices of each testset. High levels of coverage are again observed for unigrams and bigrams. However, higher-order coverage rates are extremely low. More than 98% of all 6-grams in the web data lattices are not found in the training corpus; the situation is not much better for newswire lattices. Given that the unigram coverage is so high, the low levels of higher-order n-gram coverage indicate that many lattice n-grams consist of word sequences with unusual target language word order. Most of the translation hypotheses in the lattice, therefore, will have a poor level of fluency since they fail to respect the target language grammar. These low 6-gram coverage rates explain why the 5-gram and 6-gram language models have such similar performance: the 6-gram language model must constantly back off to lower-order 5-gram probabilities when assigning scores to translation hypotheses.

5.4.3.2 Tuning the Backoff Weight

The backoff weight $\gamma(n)$ smoothes the language model scores of Equation (5.1). Table 5.13 shows the effect on the BLEU score of tuning a single order-independent backoff weight $\gamma(n)$ in Arabic→English 5-gram and 6-gram lattice rescoring. The BLEU score is observed to be fairly insensitive to changes in the backoff weight, with good performance obtained over a wide range of values. The optimised order-independent backoff weight depends on the language model order: 0.4 for the 5-gram LM and 0.7 for the 6-gram LM.

The low coverage of higher-order lattice n-grams shown in Tables 5.11 and 5.12 implies that in the 6-gram rescoring experiments of Section 5.4.3, almost all 6-grams were applied as backed-off 5-gram probabilities scaled by an associated backoff penalty $\gamma(5)$. Constant application of the same fixed penalty may degrade the quality of the language model. If $\gamma(5) = 1.0$, then no penalty is associated with backing off from a 6-gram to a 5-gram. The

Order	AR→EN reference n-grams coverage (%)			
	tune.text.nw	test.text.nw	tune.text.web	test.text.web
1g	100.00	100.00	100.00	100.00
2g	97.57	97.90	97.28	97.06
3g	87.87	88.31	85.30	83.97
4g	66.13	66.43	59.16	56.78
5g	40.89	40.39	31.31	28.67
6g	21.76	20.74	13.24	11.50

Table 5.9: Coverage (%) of Arabic→English reference n-grams by order.

Order	ZH→EN reference n-grams coverage (%)			
	tune.text.nw	test.text.nw	tune.text.web	test.text.web
1g	100.00	100.00	100.00	100.00
2g	97.52	97.53	97.21	97.08
3g	85.48	85.49	84.04	83.06
4g	60.70	60.79	57.20	55.96
5g	35.10	35.30	29.70	28.76
6g	17.84	17.88	12.49	12.00

Table 5.10: Coverage (%) of Chinese→English reference n-grams by order.

Order	AR→EN lattice n-grams coverage (%)			
	tune.text.nw	test.text.nw	tune.text.web	test.text.web
1g	99.99	100.00	99.57	99.71
2g	91.61	91.86	87.44	88.30
3g	67.42	67.35	58.52	59.41
4g	37.32	36.68	26.86	26.99
5g	15.25	14.57	8.32	8.27
6g	4.76	4.38	1.79	1.74

Table 5.11: Coverage (%) of Arabic→English lattice n-grams by order ($p = 7$).

Order	ZH→EN lattice n-grams coverage (%)			
	tune.text.nw	test.text.nw	tune.text.web	test.text.web
1g	99.97	99.97	99.76	99.69
2g	88.02	88.49	87.39	86.38
3g	58.55	58.98	56.13	54.75
4g	27.64	28.00	25.09	23.96
5g	8.95	9.07	7.58	7.02
6g	2.08	2.09	1.58	1.41

Table 5.12: Coverage (%) of Chinese→English lattice n-grams by order ($p = 7$).

$\gamma(n)$	5-gram		6-gram	
	tune.text.nw	tune.text.web	tune.text.nw	tune.text.web
0.10	46.8	46.3	46.1	45.5
0.20	46.9	46.5	46.6	46.1
0.30	47.0	46.6	46.7	46.3
0.40	47.0	46.6	46.7	46.5
0.50	47.0	46.6	46.8	46.6
0.60	46.9	46.6	46.8	46.6
0.70	46.9	46.5	46.9	46.6
0.80	46.8	46.6	46.9	46.6
0.90	46.8	46.5	46.8	46.6
1.00	46.7	46.4	46.8	46.6

Table 5.13: Second-pass LM backoff weight tuning for 5-gram (left) and 6-gram (right) rescoring. The tuned parameters were $\alpha_1 = \alpha_2 = 0.5$ and the word penalty was $\beta = 0.0$.

AR→EN	Newswire Data				Web Data			
	tune.nw		test.nw		tune.web		test.web	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
HiFST	45.9	1.000	45.0	0.995	25.2	0.998	33.1	1.000
+5-gram	47.0	1.000	46.6	0.992	26.0	0.998	34.4	1.000
+6-gram	47.0	1.000	46.7	0.995	26.0	0.994	34.4	1.000

Table 5.14: Arabic→English rescoring performance with $\gamma(5) = 1.0$. For newswire testsets $\alpha_1 = 0.5$, $\alpha_2 = 0.4$, $\beta = 0.2$. For web data testsets $\alpha_1 = 0.5$, $\alpha_2 = 0.5$, $\beta = 0.1$.

ZH→EN	Newswire Data				Web Data			
	tune.nw		test.nw		tune.web		test.web	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
HiFST	27.7	0.999	28.1	0.999	15.8	0.994	15.2	0.993
+5-gram	28.2	0.993	28.8	0.998	16.2	0.992	15.9	0.989
+6-gram	28.3	1.000	28.9	1.000	16.3	1.000	16.0	0.997

Table 5.15: Chinese→English rescoring performance with $\gamma(5) = 1.0$. For newswire testsets $\alpha_1 = 0.5$, $\alpha_2 = 0.5$, $\beta = 0.0$. For web data testsets $\alpha_1 = 0.5$, $\alpha_2 = 0.5$, $\beta = 0.0$.

following experiment evaluates 6-gram rescoring performance when the 5-gram backoff weight $\gamma(5) = 1.0$ and the other backoff weights are fixed at $\gamma(n) = 0.4$ for $n = 1 \ldots 4$.

Tables 5.14 and 5.15 show Arabic→English and Chinese→English BLEU scores obtained after 6-gram lattice rescoring with these backoff weights. The BLEU scores from 6-gram rescoring are now equal to or better than the BLEU scores obtained from 5-gram rescoring (Tables 5.6 and 5.7), although the differences are not statistically significant. This experiment shows that a single order-independent backoff weight may not be appropriate for higher-order second-pass stupid-backoff smoothed language models. However, it is the very low level of lattice 6-gram coverage that explains why the 6-gram LM performs no better than the 5-gram in rescoring. As the quality of translation hypotheses improves, higher-order coverage may increase to the point at which 6-gram LMs prove to be more effective than 5-gram LMs.

α_1	α_2	tune.text.nw		test.text.nw	
		BLEU	BP	BLEU	BP
0.0	0.8	46.4	0.999	45.8	1.000
0.1	0.7	46.6	0.998	46.2	0.994
0.2	0.7	46.8	1.000	46.4	0.993
0.3	0.5	47.0	0.998	46.4	0.991
0.4	0.5	47.0	1.000	46.5	0.993
0.5	0.5	47.0	0.999	46.6	0.992
0.6	0.4	46.9	0.999	46.5	0.992
0.7	0.3	46.8	1.000	46.3	0.993
0.8	0.3	46.8	0.999	46.0	0.993
0.9	0.2	46.6	1.000	45.7	0.993
1.0	0.2	46.2	0.999	45.4	0.993

Table 5.16: Arabic→English tune.text.nw and test.text.nw BLEU scores and brevity penalty (BP) obtained by tuning the LM exponential scale factors α_1 and α_2 in 5-gram rescoring.

5.4.3.3 Language Model Scale Factors

Table 5.16 shows Arabic→English tune.text.nw and test.text.nw 5-gram language model rescoring performance as the exponential scale factors α_1 and α_2 in the decision rule of Equation (5.3) are tuned. For each first-pass LM scale factor α_1, the second-pass LM scale factor α_2 and word penalty β are tuned to optimise the BLEU score with respect to the development set tune.text.nw. The table shows the optimised second-pass LM scale factor α_2, BLEU score, and brevity penalty (BP) obtained at each first-pass LM scale factor α_1.

The first row of the table shows rescoring performance at $\alpha_1 = 0$ so that only the second-pass language model influences the decision rule. Compared with the HiFST first-pass tune.text.nw and test.text.nw scores of 45.9 and 45.0 (Table 5.6), rescoring with $\alpha_1 = 0$ gives gains of +0.5 and +0.8 BLEU for tune.text.nw and test.text.nw, respectively. Improved performance is obtained when both language models are allowed to influence the decision rule. Performance is maximised when $\alpha_1 = \alpha_2 = 0.5$, resulting in absolute gains over the HiFST first-pass lattices of +1.1 BLEU for tune.text.nw and +1.6 BLEU for test.text.nw.

These results are interesting because the second-pass 5-gram LM training data is a superset of the training data used to estimate the parameters of the first-pass LM. Although the second-pass 5-gram has significantly wider coverage than the first-pass 4-gram, 5-gram scores are non-normalised probabilities. Furthermore, the problem of data sparsity means that many of the second-pass 5-gram probabilities may be unreliable.

The best way of incorporating higher-order language models in SMT is to integrate them directly in the decoder and optimise their feature weights using minimum error rate training (Och, 2003) (Chapter 4, Section 4.5). Incorporating multiple LM features is shown to significantly improve the quality of SMT in Brants et al. (2007). However, the large number of parameters in higher-order n-gram language models makes direct decoder integration impractical without some form of distributed client-server architecture. This is beyond the scope of the present study.

AR→EN 5-gram					Newswire Data				Web Data			
Cutoffs					tune.nw		test.nw		tune.web		test.web	
c_1	c_2	c_3	c_4	c_5	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
0	0	1	1	1	46.8	0.998	46.4	0.992	25.8	0.995	34.3	1.000
0	0	0	1	1	46.8	0.998	46.3	0.992	25.8	0.994	34.4	1.000
0	0	0	0	1	46.9	0.998	46.5	0.991	25.9	0.994	34.3	1.000
0	0	0	0	0	47.0	1.000	46.6	0.992	26.0	0.998	34.4	1.000

Table 5.17: BLEU scores and brevity penalties (BP) for Arabic→English lattice rescoring with zero-cutoff and default cutoff 5-gram language models.

ZH→EN 5-gram					Newswire Data				Web Data			
Cutoffs					tune.nw		test.nw		tune.web		test.web	
c_1	c_2	c_3	c_4	c_5	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
0	0	1	1	1	28.0	0.991	28.6	0.995	16.1	1.000	15.8	1.000
0	0	0	1	1	28.1	0.991	28.7	0.995	16.2	1.000	15.8	1.000
0	0	0	0	1	28.1	0.989	28.7	0.993	16.2	1.000	15.9	1.000
0	0	0	0	0	28.2	0.993	28.8	0.998	16.2	0.992	15.9	0.989

Table 5.18: BLEU scores and brevity penalties (BP) for Chinese→English lattice rescoring with zero-cutoff and default cutoff 5-gram language models.

5.4.3.4 Count Frequency Cutoffs

The following experiment investigates whether or not higher-order singleton counts are useful in second-pass LM rescoring. The count-of-counts statistics in Table 5.4 showed that a large proportion of higher-order n-grams occur only once, even in a large collection of over 10 billion words of training text. Since n-gram probability estimates of singletons might be unreliable, count cutoffs (Stolcke, 2002) are often used when building language models.

Let c_n denote the count frequency cutoff for n-grams of order n. The second-pass language model is modified to include n-grams of order n and frequency r only if $r > c_n$. A zero-cutoff language model has cutoffs $c_n = 0$ for all orders. Tables 5.17 and 5.18 show the effect on the BLEU score of Arabic→English and Chinese→English second-pass 5-gram language model rescoring using various higher-order n-gram count frequency cutoffs (first three rows), and a zero-cutoff language model where $c_{1,2,3,4,5} = 0$ (last row).

The zero-cutoff language model has equivalent or better BLEU score than language models with cutoffs for all testsets and language pairs. However, the relative differences in performance are quite small. Using cutoffs $c_{1,2} = 0$ and $c_{3,4,5} = 1$ degrades performance by around -0.2 BLEU with respect to the zero-cutoff model. The model with cutoffs $c_{1,2,3,4} = 0$ and $c_5 = 1$ performs nearly as well as the zero-cutoff language model.

Excluding 5-gram singleton counts removes approximately 2.7 billion parameters (37%) from the 5-gram language model. That is a lot of additional parameters to include for a gain of only +0.1 BLEU on some testsets. These results show that although higher-order singleton counts do not harm LM performance, they are of only limited utility in second-pass lattice rescoring.

5.5 Summary and Conclusions

This chapter presented a detailed empirical study of the use of higher-order n-gram language models for rescoring statistical machine translation lattices. Arabic→English and Chinese→English lattices were rescored using zero-cutoff language models estimated over approximately ten billion tokens of monolingual training data. The simple dependency structure of stupid backoff smoothing allows an efficient low-memory streaming algorithm to be used to filter n-gram counts for relevancy. Probabilities estimated from the filtered counts are encoded in finite-state, sentence-specific language model acceptors containing only the subset of parameters required to rescore each lattice. This allows large language models to be applied in offline second-pass rescoring without a distributed client-server architecture. Large gains were achieved by rescoring Arabic→English and Chinese→English lattices; smaller gains were observed for French→English and Spanish→English lattices.

Optimal performance was obtained by rescoring hypotheses according to a combination of first-pass and second-pass language model scores. Data sparsity and poor coverage of higher-order n-grams in the first-pass lattices mean that 6-gram rescoring performance is currently no better than 5-gram rescoring performance. 6-gram language models may begin to outperform 5-gram language models when much larger corpora are available, or when the quality of hypotheses in the first-pass translation lattices improves.

The language model rescoring framework described in this chapter is a simple but effective way of exploiting multi-billion token monolingual corpora. Second-pass rescoring has consistently delivered gains in submissions to the NIST, WMT and GALE evaluations of statistical machine translation quality. The 1-best translation hypotheses in the 5-gram rescored lattices serves as the baseline for more sophisticated lattice rescoring methods in the following chapters. 5-gram rescored lattices are taken as the input for rescoring with phrasal segmentation models in Chapter 6, efficient lattice minimum Bayes-risk decoding in Chapter 7, multiple lattice combination in Chapter 8, and for lattice segmentation and rescoring with fluency-motivated hypothesis space constraints in Chapter 9.

Chapter 6

Phrasal Segmentation Models for Statistical Machine Translation

Phrasal segmentation models (Blackwood et al., 2008b) define a mapping from the words of a sentence to sequences of translatable phrases, where the space of possible segmentations is determined by the phrases extracted from the word-aligned parallel data. This chapter proposes the estimation of phrasal segmentation models from large quantities of monolingual training text, and describes their realisation as weighted finite state transducers for incorporation into phrase-based statistical machine translation systems.

One of the main advantages of phrasal segmentation models is that they offer another way in which abundantly available monolingual training data, data that is normally used only for building word language models, can be exploited to improve the quality of phrase-based statistical machine translation. In this chapter, phrasal segmentation models are applied to the task of rescoring large Arabic→English word lattices produced by a phrase-based SMT decoder; significant complementary gains in BLEU score with respect to 5-gram and 6-gram word language models are demonstrated. The use of phrasal segmentation models for rescoring Arabic→English and Chinese→English lattices produced by a hierarchical phrase-based statistical machine translation decoder is also investigated.

6.1 Introduction and Motivation

In phrase-based statistical machine translation, phrases extracted from word-aligned parallel data are the fundamental unit of translation (Koehn et al., 2003; Koehn, 2010) (Chapter 4, Section 4.1.3). Each phrase is a sequence of contiguous translatable words and there is no explicit model of syntax or structure.

The first step in translating a foreign sentence is to segment the foreign sentence into a sequence of translatable phrases. Segmentations ideally capture two aspects of natural language. Firstly, segmentations should reflect the underlying grammatical sentence structure. Secondly, common word sequences should be grouped together as phrases in order to preserve context and respect collocations. Although these aspects of translation are not normally explicitly evaluated, phrases have been found very useful in translation. They have the advantage that, within extracted phrases, words appear as they were found in fluent text.

One potential disadvantage with using phrases as translation units is that reordering in current phrase-based translation models can be a major source of disfluencies. Phrasal segmentation models address such disfluencies by defining a probability distribution over the space of possible source language segmentations.[1] A strength of this approach is that it exploits abundantly available monolingual training corpora that are usually only used for building word n-gram language models.

Most prior work on phrase-based statistical language models concerns the problem of identifying useful phrasal units. In Ries et al. (1996) an iterative algorithm selectively merges pairs of words as phrases with the goal of minimising perplexity. Several criteria including word pair frequencies, unigram and bigram log likelihoods, and a correlation coefficient related to mutual information are compared in Kuo and Reichl (1999). The main difference between those approaches and the approach described in this chapter is that for phrasal segmentation models there is already a definition of the phrases of interest (that is, the phrases extracted from the word-aligned parallel text). Here, the focus is on estimating a distribution over the space of possible alternative segmentations of the sentence.

6.2 Phrasal Segmentation Models

Under the extension of the generative model of statistical machine translation (Brown et al., 1990) to phrase-based statistical machine translation (Och, 2002; Kumar and Byrne, 2003; Kumar et al., 2006), a source language sentence $s_1^I = s_1, \ldots, s_I$ generates sequences $u_1^K = u_1, \ldots, u_K$ of source language phrases that are to be considered in translation. Sentences cannot be segmented into phrases arbitrarily: the space of possible segmentations is constrained by the source language side of the phrase inventory. These are the translatable phrases found in the aligned parallel text using the phrase extraction procedures described in Chapter 4, Section 4.1.3. The distribution over phrasal segmentations is assumed to have the form

$$P(u_1^K, K|s_1^I) = P(u_1^K|K, s_1^I)P(K|I), \tag{6.1}$$

where the number of phrases K depends only on the number of source words I, and the phrase sequence u_1^K is conditioned on the number of phrases K and words of the source sentence s_1^I.

[1] In this chapter, following the source-channel model of SMT, the 'source language' refers to the output of the translation process and phrasal segmentation models are applied in lattice rescoring.

6.2.1 Uniform Phrasal Segmentation Model

The simplest phrasal segmentation model uses a uniform segmentation distribution. The distribution over the number of phrases K is chosen to be uniform so that $P(K|I) = 1/I$ for $K \in \{1, 2, \ldots, I\}$ and all segmentations are considered equally likely. Let $\{U_K\}$ denote the space of all possible length K segmentations of the source sentence s_1^I. The probability of a particular segmentation is then

$$P(u_1^K|K, s_1^I) = \begin{cases} \frac{1}{C(K,s_1^I)} & \text{if } u_1^K = s_1^I \\ 0 & \text{otherwise} \end{cases}, \tag{6.2}$$

where $C(K, s_1^I) = |\{U_K\}|$ ensures the distribution is correctly normalised and each phrase in the sequence $u_1, \ldots, u_K$ is found in the phrase inventory. This simple model of segmentation has been found useful in practice (Kumar and Byrne, 2005; Kumar et al., 2006).

6.2.2 Context-Dependent Phrasal Segmentation Model

The uniform phrasal segmentation model of Equation (6.2) can be improved by estimating phrase probabilities from naturally occurring sequences of phrases in a large monolingual training corpus (Blackwood et al., 2008b). An order-n phrasal segmentation model assigns a probability to a phrase sequence u_1^K according to

$$\begin{aligned} P(u_1^K|K, s_1^I) &= \frac{1}{Z(K, s_1^I)} \prod_{k=1}^{K} P(u_k|u_1^{k-1}, K, s_1^I) \\ &\approx \begin{cases} \frac{1}{Z(K,s_1^I)} \prod_{k=1}^{K} P(u_k|u_{k-n+1}^{k-1}) & \text{if } u_1^K = s_1^I \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{6.3}$$

where the approximation is due to the Markov assumption that only the $n-1$ most recent phrases are relevant when predicting the next phrase. Again, each u_k must be a phrase with a known translation. For a fixed sentence s_1^I, the normalisation term $Z(K, s_1^I)$ can be calculated by summing over all possible length K segmentations as follows:

$$Z(K, s_1^I) = \sum_{u_1^K \in \{U_K\}} \prod_{k=1}^{K} P(u_k|u_{k-n+1}^{k-1}) \tag{6.4}$$

In translation decoding, however, calculating this quantity becomes harder since the words of the source sentence s_1^I are not fixed. The normalisation term is therefore ignored and the unnormalised likelihoods used as scores.

6.2.3 First-Order Segmentation Model Parameter Estimation

This section describes an effective parameter estimation process for first-order phrasal segmentation models. Let $c(u_{k-1}, u_k)$ denote the count of occurrence of a contiguous string of words w_i^j in a very large training corpus that can be split at position x such that $i < x \leq j$ and the substrings w_i^{x-1} and w_x^j match precisely the words of two phrases u_{k-1} and u_k in the

phrase inventory. The first-order phrasal segmentation model parameters are computed from the relative frequency of phrase occurrences such that

$$P(u_k|u_{k-1}) = \begin{cases} \delta(u_{k-1},u_k)\frac{c(u_{k-1},u_k)}{\sum_{u_i} c(u_{k-1},u_i)} & \text{if } c(u_{k-1},u_k) > 0 \\ \gamma(u_{k-1})P(u_k) & \text{otherwise} \end{cases} \quad (6.5)$$

$$P(u_k) = \begin{cases} \frac{c(u_k)}{\sum_{u_i} c(u_i)} & \text{if } c(u_k) > 0 \\ p_0 & \text{otherwise} \end{cases} \quad (6.6)$$

where $\delta(u_{k-1},u_k)$ is a discount coefficient that reserves probability mass for unseen phrase bigrams and the context-specific backoff weights $\gamma(u_{k-1})$ ensure the distribution is correctly normalised (Katz, 1987) (Chapter 3, Section 3.2.2). Unigram phrases that were not observed in the training data are assigned a small default probability p_0. This ensures that segmentations containing single-word out-of-vocabulary phrases are assigned non-zero probabilities.

6.2.4 Phrasal Segmentation Transducers

Phrase-based TTM translation (Kumar et al., 2006) (Chapter 4, Section 4.2.3) under the uniform segmentation distribution of Equation (6.2) considers all phrasal segmentations of the source language sentence as equally likely. The uniform segmentation model can be implemented by an unweighted transducer $\mathcal{W}$ that maps word sequences to phrase sequences in accordance with the phrases of the phrase inventory. For example, if an acceptor for the source language sentence "*exhibition of students returning from abroad*" (Figure 6.1) is composed with the unweighted segmentation transducer $\mathcal{W}$ shown in Figure 6.2, the result (after optimisation) is the phrase lattice shown in Figure 6.3. This phrase lattice encodes twelve possible segmentations of the source language sentence. The shortest segmentation is the three phrase sequence "*(exhibition of) (students) (returning from abroad)*".

TTM first-pass translation using the WFST composition chain of Equation (4.11) generates word lattices $\mathcal{L}$ under the uniform segmentation distribution. In the following experiments, first-order phrasal segmentation models are applied via lattice rescoring. The word lattice $\mathcal{L}$ is first composed with the unweighted segmentation transducer $\mathcal{W}$ to obtain a lattice of source language phrases $\mathcal{L} \circ \mathcal{W}$. After ϵ-removal, determinization and minimisation operations, this lattice contains phrase sequences and translation scores consistent with the initial translation. The vocabulary of phrases relevant to each translation is also extracted.

The first-order phrasal segmentation distribution of Equation (6.3) is applied to the phrase lattice $\mathcal{L} \circ \mathcal{W}$. The conditional probabilities and backoff structure defined in Equations (6.5) and (6.6) are encoded as a weighted finite state acceptor (Allauzen et al., 2003). In this acceptor, $\mathcal{P}$, states encode histories and arcs specify the bigram and backed-off unigram phrase probabilities, as described in Chapter 3, Section 3.3.2. The raw phrase n-gram counts required by Equations (6.5) and (6.6) are collected prior to translation and first-order probabilities computed only for phrases found in the lattice. The phrasal segmentation model is composed with the phrase lattice and projected on the input to obtain the rescored word lattice:

$$\mathcal{L}' = \Pi_1(\,(\mathcal{L} \circ \mathcal{W}) \circ \mathcal{P}\,) \quad (6.7)$$

The most likely translation after applying the phrasal segmentation model $\mathcal{P}$ is found as the path in $\mathcal{L}'$ with least cost using the tropical semiring shortest distance algorithm (Mohri, 2002). Apart from likelihood pruning when generating the first-pass translation word lattice $\mathcal{L}$, the model scores are included correctly in decoder search.

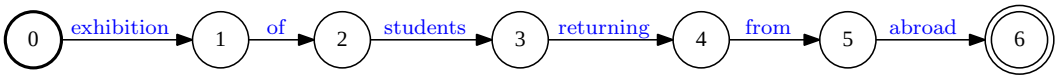


Figure 6.1: Source language sentence acceptor for the word sequence "*exhibition of students returning from abroad*". Sentence start and end tokens are omitted for clarity.

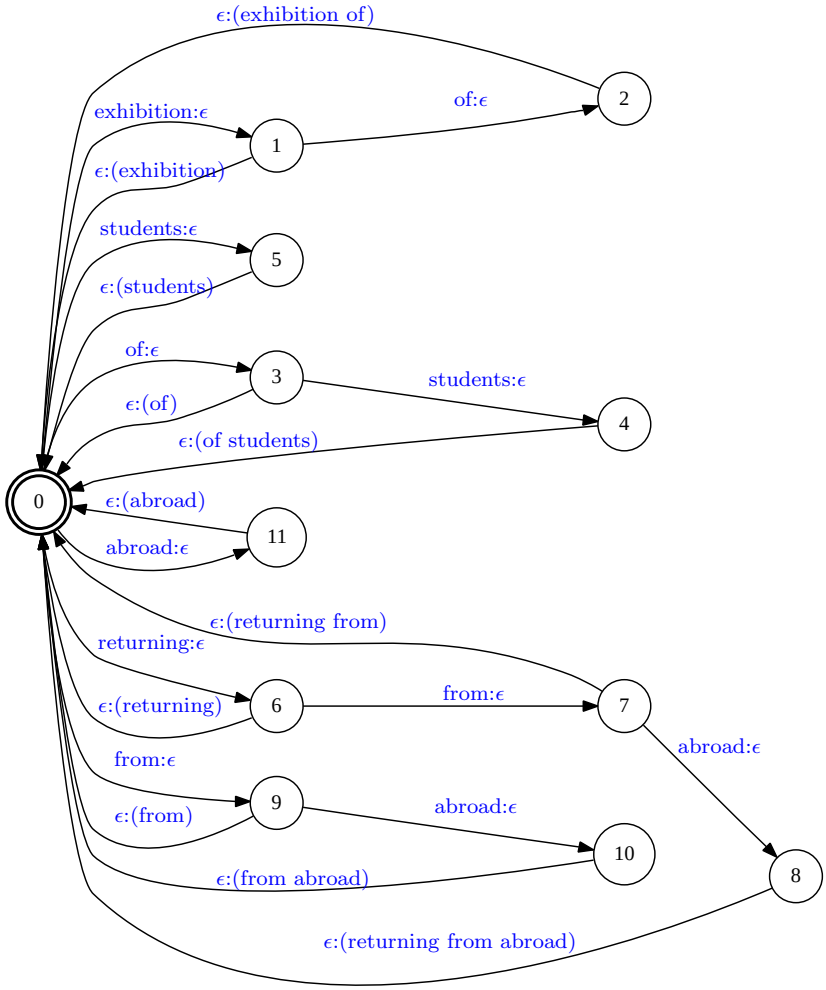


Figure 6.2: Phrasal segmentation transducer $\mathcal{W}$ for the source language string "*exhibition of students returning from abroad*" using only the phrases of the phrase inventory.

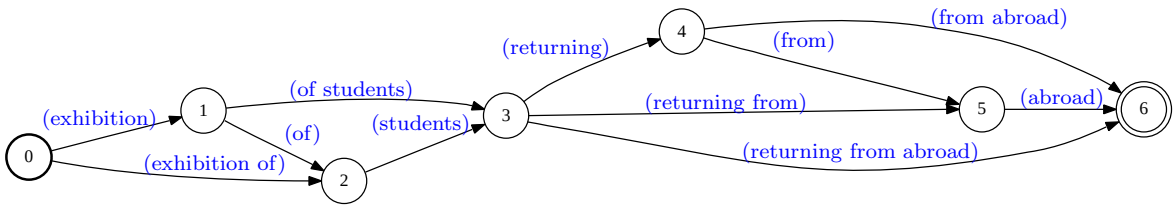


Figure 6.3: Phrase lattice encoding all possible segmentations of the source language string "*exhibition of students returning from abroad*" consistent with the phrase inventory.

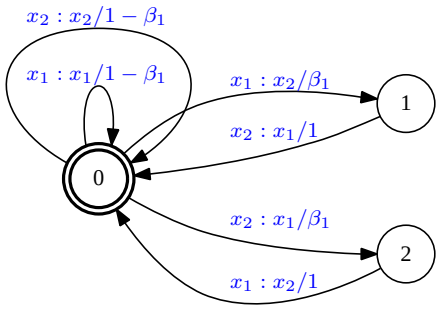


Figure 6.4: MJ1-Flat reordering transducer for any sequence of phrases $\{x_1, x_2\}^*$ where the probability of reordering two phrases is determined by a fixed jump probability of β_1.

6.2.5 Phrase Reordering Transducers

Phrasal segmentation models assign probability to phrase sequences using parameters estimated from monolingual data. Phrase sequence probabilities can also be estimated from parallel data and encoded directly in the phrase-based SMT reordering model. In the experiments reported in Section 6.3.2, the performance of phrasal segmentation model rescoring of lattices generated with first-pass phrase reordering probabilities is investigated.

In the TTM (Chapter 4, Section 4.2.3), reordering is implemented by means of a phrase jump transducer, typically combined through composition with a single-state phrase translation WFST. In qualitative terms, this simple reordering model defines a jump sequence associated with each admissible permutation of the phrases (Kumar and Byrne, 2005). In practice, it takes input source phrase sequences and outputs their translations in both monotonic and non-monotonic order.

In the simplest reordering model, known as MJ1-Flat, two adjacent phrases are allowed to swap positions with a fixed jump probability β_1 that is determined empirically. Figure 6.4 shows the MJ1-Flat WFST reordering transducer for any sequence of phrases from $\{x_1, x_2\}^*$. The form of this transducer is such that reordered phrases are always immediately followed by the phrase that was 'jumped' by the reordering phrase swap.

The MJ1-Flat reordering model is effective since it significantly broadens the search space and, as source phrases can be arbitrarily long, individual words may move quite far in translation. However, it makes no distinction as to which phrases are more likely to be reordered in translation. This problem can be addressed by defining a separate jump probability $\beta_1(v_k, u_k)$ for each phrase pair consisting of target phrase v_k and source phrase u_k (Kumar and Byrne, 2005). These probabilities can be estimated from the word alignments by examining adjacent phrase pairs and their orientation with respect to (v_k, u_k) and computing relative frequency estimates, in a similar fashion to Tillmann (2004). The β_1 probabilities then define a distribution over phrase pair sequences based on parameters estimated from the parallel data. The actual WFST implementation is analogous to MJ1-Flat, but a new state is required for each phrase bigram, since the jump probability differs in each case.

Phrase Length	Phrase Frequency	
	mt0205tune	mt0205test
1	14,570	14,347
2	85,511	84,014
3	113,541	112,601
4	77,730	78,419
5	38,516	40,014
6	16,104	17,087
7	6,193	6,561
8	2,316	2,376
9	800	841
10+	468	487
Total	355,749	356,747
w/p	3.37	3.40

Table 6.1: Source language phrase inventory statistics for Arabic→English TTM translation.

6.3 Phrase-Based Statistical Machine Translation Lattice Rescoring Experiments

In this section, phrasal segmentation models are used to rescore lattices generated by a phrase-based statistical machine translation decoder. Section 6.4 applies phrasal segmentation models to the task of rescoring lattices generated by a hierarchical phrase-based decoder.

6.3.1 TTM System Development and Lattice Generation

Phrasal segmentation model lattice rescoring is evaluated in the context of the constrained data NIST Arabic→English machine translation task. The development set mt0205tune is formed from the odd numbered sentences of the NIST MT02–MT05 testsets; the even numbered sentences form the validation set mt0205test. Test performance is reported for the NIST MT06 and MT08 testsets: mt06nw and mt08nw for newswire data; mt06ng and mt08ng for newsgroup data. NIST BLEU scores are reported for lower-case translations.

The uniformly segmented TTM baseline system is trained using all of the available Arabic↔English parallel data for the NIST MT08 evaluation.[1] Table 6.1 shows the total number of phrases, average phrase length (w/p), and lengths distribution for the phrases that could be used in translating the mt0205tune and mt0205test testsets. The average length is 3.4 words; the longest phrase is 27 words for mt0205tune and 33 words for mt0205test.

First-pass translation decoding is performed with an interpolated Kneser-Ney smoothed 4-gram language model (Kneser and Ney, 1995) estimated over the parallel text and a 965 million word subset of monolingual data from the English GigaWord Third Edition (Graff et al., 2007). Minimum error rate training (Och, 2003) under BLEU optimises the decoder feature weights with respect to the development set mt0205tune. Two first-pass translation decoders are trained: the first uses the simple MJ1-Flat reordering model; the second includes the $\beta_1(v_k, u_k)$ phrase reordering probabilities trained from the parallel data (Section 6.2.5) and the binary phrase-pair count features of Bender et al. (2007), indicating for each phrase pair

[1] http://www.nist.gov/speech/tests/mt/2008/

Corpus	# Lines	# Tokens
ptext.aren	1,442,619	46,189,306
ptext.zhen	4,904,003	122,011,558
fbis	1,027,905	29,278,009
news	12,148,324	267,821,482
giga.xin	12,071,879	299,926,282
giga.afp	21,389,287	564,645,033
Total	52,984,017	1,329,871,670

Table 6.2: Tokenised English language training corpora used to estimate parameters for phrasal segmentation model rescoring of NIST MT08 Arabic→English TTM lattices.

whether it occurred once, twice, or more than twice in the parallel data. This second decoder constitutes a much stronger baseline and is equivalent to the official CUED submission to the NIST MT08 evaluation, where it was ranked among the top systems.

In second-pass translation, 5-gram and 6-gram zero-cutoff stupid-backoff language models (Brants et al., 2007) estimated over 4.7 billion words of English newswire text are used to rescore lattices prior to applying the phrasal segmentation model. The phrasal segmentation model parameters are estimated from a 1.3 billion word subset of the same monolingual training data used to build the second-pass word language model; the phrasal segmentation model training data is summarised in Table 6.2. A phrasal segmentation model scale factor α and phrase penalty φ are tuned by grid-based search to optimise the BLEU score of the development set mt0205tune.

6.3.2 TTM Lattice Rescoring Results and Analysis

The following experiment demonstrates that phrasal segmentation models improve the quality of phrase-based SMT, even when applied to lattices that have already been rescored with powerful, high-order word language models. Table 6.3 shows phrasal segmentation model rescoring of the NIST Arabic→English mt0205tune and mt0205test testset lattices (Blackwood et al., 2008b). The optimised PSM rescoring parameters were model weight $\alpha = 0.20$ and phrase penalty $\varphi = -0.6$ for 5-gram lattice rescoring. For 6-gram lattice rescoring, the parameters were $\alpha = 0.20$ and $\varphi = -0.7$. The first row TTM+MERT shows that translations generated under the uniform segmentation model baseline obtain BLEU scores of 48.9 for mt0205tune and 48.6 for mt0205test. Large gains of +2.6 BLEU for mt0205tune and +2.9 BLUE for mt0205test are obtained though 5-gram lattice rescoring. Applying phrasal segmentation models to the 5-gram rescored lattices improves the BLEU score by an additional +1.1 BLEU for both mt0205tune and mt0205test.

For a limited quantity of monolingual training data it is not always possible to improve the quality of translation simply by increasing the order of the language model. Comparing the performance of PSMs applied to 5-gram and 6-gram rescored lattices in Table 6.3 shows that the gains in moving from a 5-gram to a 6-gram LM are very small; these results agree with the empirical study of 5-gram and 6-gram second-pass LM rescoring presented in Chapter 5. Even setting aside the practical difficulty of estimating and applying such higher-order word language models, it is doubtful that further gains could be achieved simply by increasing the order beyond $n = 6$. That phrasal segmentation model rescoring improves over the 6-gram

	mt0205tune		mt0205test	
	BLEU	BP	BLEU	BP
TTM+MERT	48.9	1.000	48.6	1.000
+5g	51.5	1.000	51.5	1.000
+PSM	52.6	1.000	52.6	1.000
TTM+MERT	48.9	1.000	48.6	1.000
+6g	51.7	1.000	51.6	0.999
+PSM	52.7	0.999	52.8	1.000

Table 6.3: NIST BLEU score and brevity penalty (BP) for phrasal segmentation model rescoring of Arabic→English NIST mt0205tune and mt0205test development set lattices.

	mt0205tune		mt0205test	
	BLEU	BP	BLEU	BP
TTM+MERT	50.9	1.000	50.3	1.000
+5g	53.5	1.000	52.4	0.987
+PSM	53.9	1.000	53.3	0.994

Table 6.4: NIST BLEU score and brevity penalty (BP) for phrasal segmentation model rescoring of Arabic→English NIST mt0205tune and mt0205test development set lattices. These experiments include the β_1 reordering probabilities and phrase pair count features in MERT.

	Newswire Data				Web Data			
	mt06nw		mt08nw		mt06ng		mt08ng	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
TTM+MERT	48.1	1.000	48.4	1.000	37.5	0.943	33.7	0.940
+5g	50.2	1.000	50.0	0.995	39.2	0.994	36.1	0.989
+PSM	51.0	1.000	50.7	0.992	39.2	0.994	36.5	0.981

Table 6.5: NIST BLEU score and brevity penalty (BP) for phrasal segmentation model rescoring of Arabic→English NIST tuning and evaluation set lattices. These experiments include the β_1 reordering probabilities and phrase pair count features in MERT.

rescored lattices suggests they capture more than just a longer n-gram context, and that gains in translation quality are complementary to the gains from second-pass word LM rescoring.

6.3.2.1 Reordering Probabilities and Phrase-Pair Count Features

The following experiment investigates phrasal segmentation model rescoring performance when the baseline first-pass translation system includes the $\beta_1(v_k, u_k)$ reordering probabilities (Kumar and Byrne, 2005) (Section 6.2.5), and the binary phrase pair count (PPC) features of Bender et al. (2007) in MERT. This system is the same as the Cambridge University Engineering Department submission to the NIST MT08 constrained data translation task (Blackwood et al., 2008a) and represents a much more challenging experimental baseline than was studied in the previous section.

Table 6.4 shows Arabic→English phrasal segmentation model rescoring results for the development sets mt0205tune and mt0205test ($\alpha = 0.2$, $\varphi = -0.7$). Table 6.5 shows rescoring results for the tuning sets mt06nw and mt06ng, and for the NIST MT08 evaluation sets

mt08nw and mt08ng. The PSM scale factor and phrase penalty parameters are optimised using mt06nw and mt06ng: for the newswire data testsets mt06nw and mt08nw, $\alpha = 0.20$ and $\varphi = -0.8$; for the web data testsets mt06ng and mt08ng, $\alpha = 0.10$ and $\varphi = -0.3$.

Comparing BLEU scores for mt0205tune and mt0205test in Tables 6.3 and 6.4 shows that the addition of phrase reordering probabilities and phrase pair count features improves the quality of the first-pass translation baseline and 5-gram rescored lattices significantly. Even with this much stronger baseline, phrasal segmentation model rescoring leads to good gains on the newswire testsets. On mt06nw and mt08nw the gains are +0.8 BLEU and +0.7 BLEU, respectively. Performance on the newsgroup data is quite a lot worse: there is no gain on mt06ng and only +0.4 BLEU on the mt08ng testset.

The overall gains from PSM rescoring are smaller than the gains observed when rescoring lattices generated without reordering probabilities and phrase pair count features. These results suggest that there is an overlap between the aspects of phrase sequence order captured by the β_1 probabilities and PPC features, and the information captured by phrasal segmentation models. However, while the β_1 probabilities and PPC features are estimated from parallel data, the phrasal segmentation model parameters are estimated from monolingual data. Since there is so much more monolingual data, it may be easier to improve PSM performance than to improve the contribution to translation quality from the β_1 probabilities and phrase pair count features.

Comparing the in-domain newswire (mt08nw and mt06nw) and out-of-domain newsgroup (mt06ng and mt08ng) testset performance shows the importance of choosing appropriate data for estimating the parameters of the phrasal segmentation model. When in-domain data is of limited availability, count mixing (Bacchiani et al., 2004) or other language model adaptation strategies (Bellegarda, 2004) may lead to improved performance.

6.3.2.2 Phrase Penalty Tuning

The role of the phrase penalty φ is to encourage longer phrases in translation. Table 6.6 shows the effect of tuning this parameter. The upper part of the table shows the NIST BLEU score, brevity penalty and individual n-gram precisions. The lower part of the table shows the total number of words in the output, the number of words translated as a phrase of the specified length, and the average number of words per phrase.

When the phrase penalty is too low, single word phrases dominate the output and the benefits of longer context and phrase-internal fluency are lost. As the phrase penalty increases, there are large gains in precision at each order and many longer phrases appear in the output. At the optimal phrase penalty, the average phrase length is 1.58 words and over 60% of the translation output is generated from multi-word phrases.

6.4 Hierarchical Phrase-Based Translation Lattice Rescoring

Phrasal segmentation models were originally developed to model the segmentation process in phrase-based statistical machine translation. Hierarchical phrase-based translation (Chiang, 2007; Iglesias et al., 2009b) (Chapter 4, Section 4.3) is driven by synchronous context-free grammar rules extracted from word-aligned parallel data. Although there is no explicit model

	Phrase penalty φ				
	-4.0	-2.0	0.0	2.0	4.0
BLEU	48.6	50.1	51.1	49.9	48.7
BP	0.000	0.000	0.000	-0.034	-0.072
1g	82.0	83.7	84.9	85.7	86.2
2g	57.3	58.9	59.9	60.5	61.1
3g	40.8	42.2	43.1	43.6	44.2
4g	29.1	30.3	31.1	31.5	32.0
words	70550	66964	63505	60847	58676
1	58840	46936	25040	15439	11744
2	7606	12388	18890	19978	18886
3	2691	4890	11532	13920	14295
4	860	1820	5016	6940	8008
5	240	450	1820	2860	3500
6+	313	480	1207	1710	2243
w/p	1.10	1.21	1.58	1.86	2.02

Table 6.6: Effect of phrase penalty φ on NIST BLEU score, brevity penalty (BP), individual n-gram precisions at each order, phrase lengths distribution, and average number of words per phrase (w/p) for the Arabic→English mt0205tune testset.

of the segmentation process, phrasal segmentation models can still be applied to rescore lattices generated by a hierarchical phrase-based decoder if an appropriate collection of phrases can be identified. Rules in the grammar have the form

$$X \rightarrow \langle \gamma, \alpha, \sim \rangle, \tag{6.8}$$

where X is a non-terminal, γ is a sequence of non-terminals and terminals in the source language, and α is a sequence of non-terminals and terminals in the target language.[1] The relation $\sim$ defines the one-to-one alignment of source and target non-terminals.

Target language phrases can be extracted from the rules of the grammar by identifying contiguous sequences of terminals in α. These sequences are added to the phrase inventory that determines the space of possible segmentations. Figure 6.5 shows one possible source language derivation obtained in translating an GALE P4 Chinese→English reference sentence. The source and target language phrases associated with this tree are shown by the boxes at the bottom of the figure. For this example, seven distinct target language phrases are extracted with lengths varying from one to four words.

Phrase unigram and bigram counts are collected for each of the phrases extracted from the testset rules; phrasal segmentation model parameters are then estimated using Equations 6.5 and 6.6. The lattice rescoring procedure is the same as for phrase-based lattices.

6.4.1 HiFST System Development and Lattice Generation

This section describes phrasal segmentation model rescoring of large Arabic→English and Chinese→English lattices generated by a state-of-the-art hierarchical phrase-based transla-

[1] In this section, following the use of the log-linear direct translation model, 'source language' denotes the input language and 'target language' denotes the output language; the goal, then, is to rescore translated lattices using a 'target language' segmentation model.

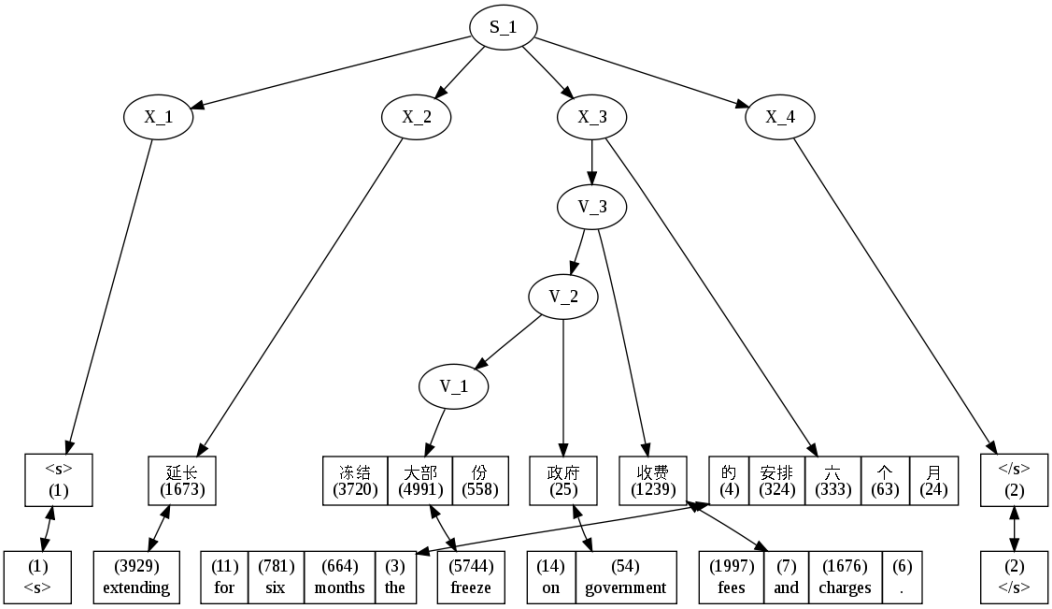


Figure 6.5: Derivation example for an GALE P4 Chinese→English reference sentence showing how a list of target language phrases can be extracted from the rules of grammar.

Corpus	# Lines	# Tokens
ptext.aren	9,359,668	231,963,900
ptext.zhen	11,019,719	263,633,514
giga.xin	14,242,285	358,147,081
giga.afp	27,241,407	725,927,421
Total	61,863,079	1,579,671,916

Table 6.7: Tokenised English language training corpora used for phrasal segmentation model rescoring of GALE P4 Arabic→English and Chinese→English HiFST lattices.

tion system. The testsets, first-pass translation decoder, and lattice generation procedures constitute the same GALE P4 evaluation framework used for the empirical study of language model rescoring in Chapter 5, Section 5.4. Those 5-gram rescored lattices serve as the baseline in the following phrasal segmentation model rescoring experiments.

The target language training corpora used to estimate the phrasal segmentation model parameters of Equations 6.5 and 6.6 are summarised in Table 6.7. These corpora constitute a total of around 1.6 billion words and include the target language side of the Arabic→English and Chinese→English parallel texts, and the monolingual Xinhua and AFP subsets of the English GigaWord Fourth Edition (Parker et al., 2009).

The total number of phrases, average phrase length in words (w/p), and phrase lengths distribution for the phrases extracted from the grammar rules of each GALE P4 testset are summarised in Table 6.8. Comparing these phrase inventories with the phrase inventories used to rescore phrase-based SMT lattices (Table 6.1) shows that considerably more phrases are extracted. For Chinese→English translation, more than one million phrases are extracted from the rules of the grammar. Apart from the larger testset size, one reason for the very large number of phrases is the increased size of the parallel data used in the GALE P4 experiments:

Phrase	Arabic→English		Chinese→English	
Length	tune.text.nw	tune.text.web	tune.text.nw	tune.text.web
1	22,785	26,726	23,061	23,351
2	179,387	182,054	245,453	246,175
3	259,342	238,512	358,978	357,962
4	185,538	148,391	246,338	240,048
5	99,131	66,844	129,754	121,881
6	45,087	25,410	57,792	52,053
7	18,764	9,323	24,208	20,613
8	7,322	3,460	10,172	8,131
9	2,814	1,393	4,612	3,435
10+	2,116	1,100	4,929	3,009
Total	822,286	703,213	1,105,297	1,076,658
w/p	3.54	3.28	3.55	3.48

Table 6.8: Source language phrase inventory statistics for GALE P4 Arabic→English and Chinese→English hierarchical phrase-based translation using HiFST.

∼230M words vs. ∼46M words for the Arabic→English NIST MT08 experiments. A second reason is that extracting contiguous strings of terminals from rules results in many more phrases than are obtained from the word alignments using the conservative phrase-extraction algorithm described in Chapter 4, Section 4.1.3.

6.4.2 HiFST Lattice Rescoring Results and Analysis

Tables 6.9 and 6.10 show BLEU scores and brevity penalties (BP) for PSM rescoring of GALE P4 Arabic→English and Chinese→English 5-gram rescored lattices generated using the hierarchical phrase-based decoder HiFST. The first-pass lattices were generated at a likelihood pruning threshold of $p = 9$. Rescoring with the second-pass 5-gram LM results in large gains of +1.6 BLEU on test.text.nw and +1.3 BLEU on test.text.web over the Arabic→English MERT optimised baseline lattices; for Chinese→English, the gains are a little smaller.

Phrasal segmentation model rescoring of both Arabic→English and Chinese→English lattices is observed to provide only relatively small gains with respect to the 5-gram rescored lattices. The gains on the testsets are +0.2 BLEU for newswire data and +0.1 BLEU for web data. Compared to the large gains obtained through phrasal segmentation model rescoring of lattices produced by the TTM phrase-based SMT decoder (Tables 6.4 and 6.5), these gains are somewhat disappointing. The optimised model weight and phrase penalty for each language pair and genre are summarised in Table 6.11.

The translation baseline in these GALE P4 experiments represents the strongest system yet developed at CUED, incorporating word alignments over a very large parallel text and a powerful hierarchical decoder supporting direct generation of target language lattices. As described in Chapter 5, the first-pass lattices are rescored with zero-cutoff stupid-backoff n-gram language models estimated over approximately 10 billion words of monolingual data prior to phrasal segmentation model rescoring. This is a very difficult baseline to improve. An alternative method of parameter estimation and efficient training algorithm that enables the use of much larger corpora may be required in order to effectively apply phrasal segmentation models to hierarchical phrase-based statistical machine translation lattices.

AR→EN	Newswire Data				Web Data			
	tune.text.nw		test.text.nw		tune.text.web		test.text.web	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
HiFST	45.9	1.000	45.0	0.995	25.2	0.998	33.1	1.000
+5g	47.0	1.000	46.6	0.992	26.0	0.998	34.4	1.000
+PSM	47.1	1.000	46.8	0.995	26.2	0.994	34.6	1.000

Table 6.9: IBM BLEU score and brevity penalty (BP) for PSM rescoring of GALE P4 Arabic→English 5-gram rescored tuning and evaluation set lattices.

ZH→EN	Newswire Data				Web Data			
	tune.text.nw		test.text.nw		tune.text.web		test.text.web	
	BLEU	BP	BLEU	BP	BLEU	BP	BLEU	BP
HiFST	27.7	0.999	28.1	0.999	15.8	0.994	15.2	0.993
+5g	28.2	0.993	28.8	0.998	16.2	0.992	15.9	0.989
+PSM	28.3	0.996	28.9	0.998	16.2	0.999	16.0	0.997

Table 6.10: IBM BLEU score and brevity penalty (BP) for PSM rescoring of GALE P4 Chinese→English 5-gram rescored tuning and evaluation set lattices.

		AR→EN	ZH→EN
Newswire	α	0.075	0.100
	φ	-0.2	-0.3
Web	α	0.200	0.050
	φ	-0.2	-0.2

Table 6.11: Phrasal segmentation model rescoring model weight (α) and phrase penalty (φ) for GALE P4 Arabic→English and Chinese→English newswire and web data testsets.

6.5 Summary and Conclusions

This chapter defined a simple but effective stochastic model of the phrasal segmentation process appropriate for phrase-based SMT (Blackwood et al., 2008b). The model parameters are estimated from naturally occurring phrase sequence examples in a large monolingual corpus. First-order phrasal segmentation models applied to the NIST Arabic→English MT08 task demonstrated complementary improved translation quality with respect to large, zero-cutoff 5-gram and 6-gram word language models (Blackwood et al., 2009).

Phrasal segmentation models represent a novel way of exploiting the same abundantly available monolingual data normally used only for building word language models. Chapter 9 proposes another way in which the same monolingual data can be used to improve the quality of statistical machine translation: monolingual coverage constraints.

One possible extension of the phrasal segmentation model described in this chapter is to use the n-multigram model of Deligne and Bimbot (1995). The n-multigram model defines a joint distribution over word sequences and their segmentation as a sequence of multi-word units; these multi-word units are phrases in phrase-based SMT. Starting from an initial estimate of the parameters, the expectation-maximisation algorithm (Dempster et al., 1977) can be used to iteratively re-estimate the parameters from segmentations of the training data.

Source	# Lines	# Words	W / L	Segmentable
ptext.aren	1,163,405	35,953,362	30.9	77.8%
ptext.zhen	3,924,866	93,050,876	23.7	76.3%
giga.xin	8,643,103	193,856,353	22.4	64.6%
giga.afp	13,829,942	334,064,863	24.2	59.2%
Total	27,561,316	656,925,454	23.8	63.6%

Table 6.12: Corpus segmentability using the phrases of the NIST MT08 phrase inventory.

Conditional expected counts of phrase n-grams under the current parameterisation determine the parameters of the next iteration. This approach differs from the phrasal segmentation model described in this chapter since it requires segmenting the monolingual training data instead of just the sentences of the testset. Segmenting the monolingual data requires a rich inventory of phrases. Table 6.12 shows that over 75% of the Arabic→English and Chinese→English parallel text and around 60% of the monolingual training data can be segmented using the phrases of the NIST MT08 translation task, a total of over 650M words. The average length of segmentable sentences is close to the average length of 26 words per sentence in the full training data. This implies that the segmentable subset is representative of the full corpus. The segmentable subset may also be more relevant since it is the subset of sentences that can be segmented using only phrases extracted from the alignments. Given the large quantities of data involved, the multigram phrasal segmentation model will require an efficient training algorithm of the form described in Deligne and Bimbot (1995).

Phrasal segmentation models were originally developed to address disfluencies introduced by the reordering process in phrase-based SMT. Extracting phrase n-gram statistics from large monolingual corpora allows a probability distribution to be defined over the space of possible segmentations; hypotheses can then be ranked according to the likelihood of their segmentation. Hierarchical phrase-based translation supports much more flexible reordering and longer distance movement of words and phrases. Although the segmentation process is not explicitly modelled, this chapter showed how an inventory of phrases can be extracted from the rules of the target language grammar so that phrasal segmentation models can be applied in lattice rescoring. However, this approach resulted in only small gains when applied to GALE P4 Arabic→English and Chinese→English lattices.

For hierarchical phrase-based translation, a different set of statistics may be more appropriate. Instead of computing regular phrase n-gram counts as described in Section 6.2.3, it might be useful to compute counts that incorporate the hierarchical relationships between the rules of the grammar. For example, counts of phrase bigrams could be collected by summing over all target language non-terminal phrasal substitutions; the aim of this form of parameter estimation would be to more closely match the reordering patterns actually used in hierarchical phrase-based translation.

Chapter 7

Lattice Minimum Bayes-Risk Decoding with Weighted Finite-State Transducers

Minimum Bayes-risk (MBR) decoding has been found useful in many areas of natural language processing (Duda et al., 2000; Goel and Byrne, 2000; Goel et al., 2004; Kumar and Byrne, 2004). This chapter describes the use of MBR decoding to improve the quality of large-scale statistical machine translation systems. The general form of the MBR decoder is first defined and described. A linear approximation to the loss function based on n-gram posterior probabilities (Tromble et al., 2008) allows MBR decoding to be applied to the full space of hypotheses encoded in large translation lattices.

This chapter starts by reviewing the Tromble et al. (2008) linearised form of lattice MBR decoder. Then, an original and improved exact formulation of linearised lattice MBR based on efficient path counting transducers is introduced (Blackwood and Byrne, 2010). Comprehensive experiments with multiple language pairs provide a contrastive study of the performance and efficiency of k-best MBR and lattice MBR. The following chapter applies lattice MBR techniques to multi-input and multi-source translation in a system combination framework.

7.1 Minimum Bayes-Risk Decoding for Machine Translation

This section describes minimum Bayes-risk (MBR) decoding for statistical machine translation. An implementation based on weighted finite-state acceptors allows MBR decoding to be applied to the full space of hypotheses in large machine translation lattices. Faster lattice MBR decoding based on efficient path counting transducers is introduced in Section 7.2, followed by an empirical study of lattice MBR applied to large-scale Arabic→English and Chinese→English translation tasks in Section 7.3.

7.1.1 Background and Related Work

Decoding under the standard Maximum A Posteriori (MAP) decision rule chooses the output with the highest posterior probability (Duda et al., 2000). Minimum Bayes-risk decoding differs from MAP decoding by choosing the output that minimises the expected loss due to errors, according to a loss function that measures task performance. The decoding decision rule is thus optimised directly for specific loss functions based on metrics of interest.

The MAP decision rule can be derived as a special case of the MBR decision rule by using a zero–one loss function in which all misclassifications are considered equally poor. The loss function of a MAP decoder is thus too harsh – it applies the same fixed penalty to outputs regardless of their quality and fails to distinguish between different types of error.

The exact choice of loss function depends on the application and metric of interest. Typically, in machine translation, lexical loss functions such as the BLEU score (Papineni et al., 2002b), translation edit rate (TER) (Snover et al., 2006), or position-independent word error rate (PER) are used. Since these functions depend only on the string of words in the candidate hypothesis, the loss can be computed relatively efficiently. More complex loss functions incorporating richer sources of linguistic information such as word-to-word alignments or syntactic parse trees have also been shown to be useful for SMT (Kumar and Byrne, 2004).

MBR decoding under the sentence-level BLEU score has been successfully applied to the task of re-ranking machine translation hypotheses in a k-best list (Kumar and Byrne, 2004). For efficiency reasons, these lists typically range in depth from 100 to 10,000 hypotheses. MBR decoding can also be applied to translation lattices, directed acyclic graphs that efficiently encode very large numbers of alternative translations (Ueffing et al., 2002). The large number of additional translations available to lattice MBR has been shown to significantly improve translation quality over k-best MBR (Tromble et al., 2008).

7.1.2 Minimum Bayes-Risk Decoding for Machine Translation

Minimum Bayes-risk decoding for statistical machine translation (Kumar and Byrne, 2004; Ehling et al., 2007) selects the translation hypothesis with the lowest expected risk given the underlying probabilistic model. The result is a sentence-level consensus choice of the best hypothesis. For arbitrary loss function $L(E, E')$ between translation hypothesis E' and reference translation E, and given the underlying probabilistic model $P(E|F)$, MBR decoding has the general form

$$\hat{E} = \underset{E' \in \mathcal{E}}{\operatorname{argmin}} R(E') = \underset{E' \in \mathcal{E}}{\operatorname{argmin}} \sum_{E \in \mathcal{E}} L(E, E') P(E|F), \tag{7.1}$$

where $R(E')$ is the Bayes-risk of hypothesis E' (i.e. the conditional expected loss under loss function L) and $\mathcal{E}$ represents the space of available translations, e.g. a k-best list or lattice produced by a machine translation decoder. If L_{max} bounds the maximum loss between any two hypotheses, then the decoder can be rewritten in terms of a gain function $G(E, E') = L_{max} - L(E, E')$. The MBR decision rule is then

$$\hat{E} = \underset{E' \in \mathcal{E}}{\operatorname{argmax}} \sum_{E \in \mathcal{E}} G(E, E') P(E|F). \tag{7.2}$$

For MBR decoding of translation lattices, an appropriate gain function is the sentence-level BLEU score (Papineni et al., 2002b) (Section 4.4.1). Sentence-level BLEU is simply the geometric mean of n-gram precisions and ignores the brevity penalty required for corpus-level BLEU. It varies between 0 and 1 with higher values indicating a greater degree of similarity between hypothesis and reference. MBR decoding is able to use different spaces for hypothesis search and risk computation. The general form of the decoder is therefore

$$\hat{E} = \underset{E' \in \mathcal{E}_h}{\operatorname{argmax}} \sum_{E \in \mathcal{E}_e} G(E, E') P(E|F), \tag{7.3}$$

where $\mathcal{E}_h$ is the hypothesis space from which the minimum risk hypothesis is selected and $\mathcal{E}_e$ is the evidence space used to compute the Bayes-risk. The relative importance of these two spaces is analysed in Tromble et al. (2008), where it is shown that accurate computation of the expected risk in a large evidence space is more important than a large hypothesis space of translations to search amongst during decoding.

MBR decoding on k-best lists has computational complexity $\mathcal{O}(n^2)$. Ehling et al. (2007) show that the summation over $E \in \mathcal{E}$ in Equation (7.1) can be terminated as soon as the accumulation of expected risk exceeds the current lowest-risk hypothesis. Even with this optimisation, however, the computational complexity still limits k-best MBR to a relatively short list of hypotheses.

7.1.3 Lattice Minimum Bayes-Risk Decoding

Machine translation lattices are a compact representation of large numbers of translation alternatives with scores (see Section 4.2.2 in Chapter 4). Each arc corresponds to a single word and the weight obtained by aggregating arc costs along a complete path through the lattice gives the likelihood of the hypothesised word sequence. The posterior probability of translation hypothesis E given foreign source sentence F is

$$P(E|F) = \frac{\exp(\alpha H(E, F))}{\sum_{E' \in \mathcal{E}} \exp(\alpha H(E', F))}, \tag{7.4}$$

where $H(E, F)$ gives the score of candidate translation E according to the model, e.g. the product of feature and weight vectors in a log-linear model. The exponential scale factor α smoothes the posterior distribution, flattening when $\alpha < 1$ and sharpening when $\alpha > 1$.

Since the number of hypotheses encoded in a lattice can be exponential in the number of states, it is not always possible to explicitly compute the gain for each individual hypothesis. This is the reason why k-best MBR is typically applied to relatively shallow lists. However, by decomposing the gain function $G(E, E')$ of the MBR decision rule in Equation (7.2) as

a sum of independent local gain functions g_u the decoder can be reformulated in terms of n-gram matches between E and E' and computed efficiently (Tromble et al., 2008).

Let $\mathcal{N} = \{u_1, \ldots, u_{|\mathcal{N}|}\}$ denote the set of all n-grams in lattice $\mathcal{E}$ and define the n-gram local gain function between two hypotheses $g_u : \mathcal{E} \times \mathcal{E} \rightarrow \mathbb{R}$ for each $u : \in \mathcal{N}$ as

$$g_u(E, E') = \theta_u \#_u(E')\delta_u(E), \tag{7.5}$$

where θ_u is an n-gram specific constant, $\#_u(E')$ is the number of times u occurs in E', and $\delta_u(E)$ is 1 if u occurs in E and zero otherwise. The gain g_u is thus θ_u times the number of occurrences of u in E', or zero if u does not occur in E. Using a first order Taylor-series approximation to the gain in log corpus BLEU (Tromble et al., 2008), the overall gain function $G(E, E')$ can be approximated as a linear sum of these local gain functions and a constant θ_0 times the length of the hypothesis E':

$$G(E, E') = \theta_0|E'| + \sum_{u \in \mathcal{N}} g_u(E, E') \tag{7.6}$$

Substituting this linear decomposition of the gain function into Equation (7.2) results in an MBR decoder with the form

$$\hat{E} = \underset{E' \in \mathcal{E}}{\operatorname{argmax}} \left\{ \theta_0|E'| + \sum_{u \in \mathcal{N}} \theta_u \#_u(E')p(u|\mathcal{E}) \right\}, \tag{7.7}$$

where $p(u|\mathcal{E})$ is the path posterior probability of n-gram u which can be computed from the lattice. The important point is that the linear decomposition of the gain function replaces the sum over an exponentially large set of hypotheses in the lattice $E \in \mathcal{E}$ with a sum over n-grams $u \in \mathcal{N}$ which can be computed exactly even for large lattices. The n-gram path posterior probability is the sum of the posterior probabilities of all paths containing the n-gram:

$$p(u|\mathcal{E}) = \sum_{E \in \mathcal{E}_u} P(E|F), \tag{7.8}$$

where $\mathcal{E}_u = \{E \in \mathcal{E} \colon \#_u(E) > 0\}$ is the subset of lattice paths containing the n-gram u at least once. The next section describes how these path posterior probabilities can be computed efficiently using general purpose WFST operations.

7.1.4 Decoding with Weighted Finite-State Acceptors

This section describes an implementation of lattice minimum Bayes-risk decoding based on weighted finite-state acceptors (Mohri, 1997) and the OpenFst toolkit (Allauzen et al., 2007). Each lattice $\mathcal{E}$ is a weighted directed acyclic graph (DAG) (Cormen et al., 2001) encoding a large space of hypothesised translations output by the baseline system. Denote by $\mathcal{E}_h$ the hypothesis space (e.g. the top 1000-best hypotheses in an k-best list generated from the lattice) and by $\mathcal{E}_e$ the evidence space.

The lattice MBR decoder of Equation (7.7) is implemented by the algorithm shown in Figure 7.1. The input parameters are the posterior distribution smoothing factor α, evidence space $\mathcal{E}_e$, hypothesis space $\mathcal{E}_h$, and n-gram factors θ_n for $n = 0, \ldots, 4$. The return value is the translation hypothesis that maximises the conditional expected gain. The algorithm corresponds to the following sequence of operations:

LMBR-DECODE($\alpha, \mathcal{E}_e, \mathcal{E}_h, \theta_{0\ldots4}$)

1 $\mathcal{E}_e \leftarrow$ FST-NORMALIZE($\alpha \times \mathcal{E}_e$)
2 $\mathcal{N} \leftarrow$ EXTRACT-NGRAMS($\mathcal{E}_h$)
3 **for each** $u \in \mathcal{N}$
4 **do** $\Psi_u \leftarrow$ MAKE-COUNT-FSA(u)
5 $\mathcal{E}_u \leftarrow \mathcal{E}_e \circ \Psi_u$
6 $p(u|\mathcal{E}_e) \leftarrow \sum_{E \in \mathcal{E}_u} P(E|F)$
7 $\mathcal{E}_h \leftarrow$ APPLY-WORD-FACTOR($\mathcal{E}_h, \theta_0$)
8 **for each** $u \in \mathcal{N}$
9 **do** $\Omega_u \leftarrow$ MAKE-GAIN-FSA($u, \theta_{|u|} \times p(u|\mathcal{E}_e)$)
10 $\mathcal{E}_h \leftarrow \mathcal{E}_h \circ \Omega_u$
11 **return** FIND-BEST-PATH($\mathcal{E}_h$)

Figure 7.1: Lattice minimum Bayes-risk decoding algorithm.

(i) After applying the exponential scale factor α of Equation (7.4), the hypothesis likelihoods are converted to normalised posterior translation probabilities $P(E|F)$ by mapping to the log semiring, pushing weights to the final state, and removing the final state costs (line 1) (these operations are described in Chapter 2, Sections 2.4.1 and 2.4.3). After this operation, $\sum_E P(E|F) = 1$. The n-gram path posterior probabilities are simply the log semiring $\oplus$-sum of the weights of paths containing the n-gram.

(ii) The set of n-grams $\mathcal{N} = \{u_1, u_2, \ldots, u_{|\mathcal{N}|}\}$ is extracted from the hypothesis space $\mathcal{E}_h$ (line 2). Tromble et al. (2008) describes an arc traversal algorithm for generating n-gram sequences from a topologically sorted acyclic lattice. For large lattices with high average branching factor, this algorithm is slow. The n-grams can be more efficiently extracted by composing $\mathcal{E}_h$ with an n-gram counting transducer (Allauzen et al., 2003).

(iii) The sum over $\mathcal{N}$ in the MBR decoder of Equation (7.7) requires the path posterior probability $p(u|\mathcal{E})$ of each n-gram in the hypothesis space. These probabilities are computed from the evidence space $\mathcal{E}_e$ by creating an unweighted acceptor $\Psi_u = \Sigma^* \, u \, \Sigma^*$ that is composed with $\mathcal{E}_e$ to form the subspace $\mathcal{E}_u = \mathcal{E}_e \circ \Psi_u$ of paths with at least one occurrence of the n-gram u. The composition $\mathcal{E}_e \circ \Psi_u$ counts paths in $\mathcal{E}_e$ containing u, where each count is weighted by the posterior probability of the path on which it occurs. The n-gram path posterior probability $p(u|\mathcal{E})$ is the sum of the probabilities of all paths in $\mathcal{E}_u$, and is computed by pushing weights in the log semiring. This process is repeated for each n-gram in the hypothesis space (lines 3–6).

(iv) The contribution to the gain function $g_u(E, E')$ for each n-gram u in Equation (7.6) is applied by creating an automaton Ω_u that accepts u with weight $\theta_{|u|} \times p(u|\mathcal{E})$ (line 9). The composition $\Omega_u \circ \mathcal{E}_h$ applies n-gram factor $\theta_{|u|}$ and posterior probability $p(u|\mathcal{E})$ once for each occurrence of u, thus incorporating the count $\#_u(E')$ in $G(E, E')$.

(v) Decoding starts from an initially unweighted copy of the hypothesis space $\mathcal{E}_h$ and composes in sequence the partial gain applicator Ω_u for each $u \in \mathcal{N}$ to accrue the conditional expected gain (lines 8–10). The word factor is applied by setting all costs of all arcs

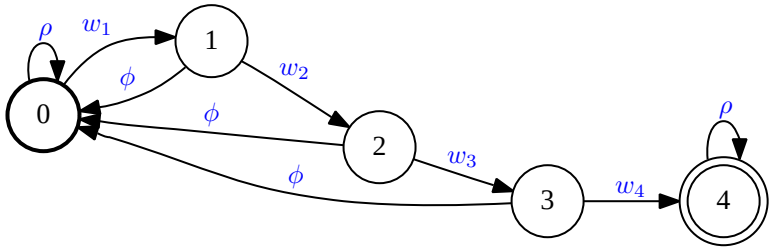


Figure 7.2: Unweighted finite-state acceptor Ψ_u for counting paths in the lattice subset $\mathcal{E}_u$ containing at least one occurrence of the n-gram $u = w_1 w_2 w_3 w_4$.

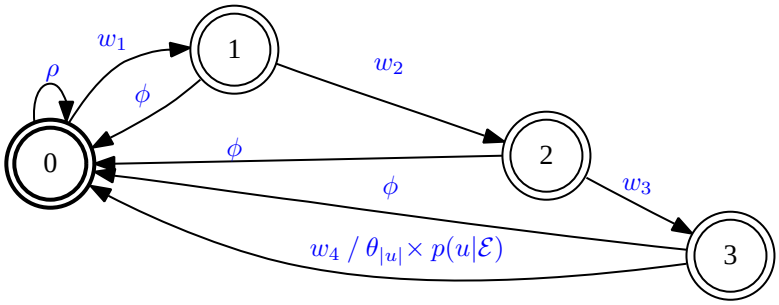


Figure 7.3: Weighted finite-state acceptor Ω_u to apply gain $\theta_{|u|} \times p(u|\mathcal{E})$ to each occurrence of the n-gram $u = w_1 w_2 w_3 w_4$ in a translation lattice.

in $\mathcal{E}_h$ to θ_0. After the gain for all n-grams has been applied, a path corresponding to word sequence E' has weight $\theta_0|E'| + \sum_{u \in \mathcal{N}} \theta_{|u|} \#_u(E') p(u|\mathcal{E})$. The hypothesis $\hat{E}$ that maximises the expected gain is the best path in the $(max, +)$ semiring (line 11).

Figure 7.2 shows the path counting acceptor Ψ_u for the n-gram $u = w_1 w_2 w_3 w_4$. This acceptor can be used to count paths in the lattice with at least one occurrence of the word sequence $w_1 w_2 w_3 w_4$. Composing Ψ_u with the lattice discards all paths that do not contain u since the final state is only reached by reading the complete n-gram u. The use of ϕ-transitions (see Chapter 2, Section 2.4.2) avoids duplication of paths during composition. If instead regular ϵ-transitions are used, then duplicate paths must be discarded by tropical $(min, +)$ semiring determinization before summing path probabilities. Figure 7.3 shows the gain applicator Ω_u for the same n-gram $u = w_1 w_2 w_3 w_4$. Composing Ω_u with the hypothesis space accrues the partial gain associated with n-gram u. Again, the use of non-consuming ϕ-transitions ensures that duplicate paths are not introduced during decoding.

The implementation of lattice MBR using the algorithm of Figure 7.1 follows Tromble et al. (2008), with local refinements for efficient extraction of n-grams from the lattice, and faster matching through the use of special labels when computing path posterior probabilities and decoding. The next section presents a faster novel implementation of lattice MBR decoding based on the use of efficient path counting transducers.

7.2 Efficient Path Counting Transducers for Lattice Minimum Bayes-Risk Decoding

This section presents a novel implementation of the Tromble et al. (2008) linearised form of lattice minimum Bayes-risk decoding based on general purpose weighted finite-state transducer operations (Blackwood and Byrne, 2010). The use of transducers instead of acceptors allows the posterior probabilities of all n-grams of a given order to be computed simultaneously with a single composition. This yields an implementation that is fast and exact even for very large lattices.

7.2.1 Path Posterior Probabilities and Expected Counts

The quantity $p(u|\mathcal{E})$ in the lattice MBR decoder of Equation (7.7) is the path posterior probability of n-gram u given the lattice $\mathcal{E}$. This particular posterior is defined as

$$p(u|\mathcal{E}) = \sum_{E \in \mathcal{E}_u} P(E|F), \tag{7.9}$$

where $\mathcal{E}_u = \{E \in \mathcal{E} : \#_u(E) > 0\}$ is the subset of paths in the lattice containing u at least once, and $P(E|F)$ is the posterior probability of translation hypothesis E given the source language input sentence F. Even though a lattice may contain many n-grams, it is possible to extract and enumerate them exactly whereas this is often impossible for individual paths. Therefore, while the Tromble et al. (2008) linearisation of the gain function is an approximation, the decision rule of Equation (7.7) can be computed exactly even over very large lattices. The challenge is to do so as efficiently as possible.

If the quantity $p(u|\mathcal{E})$ had the form of a conditional expected count, then it could be computed efficiently using the regular form of WFST counting transducer (Allauzen et al., 2003). The conditional expected count $c(u|\mathcal{E})$ of n-gram u is computed as

$$c(u|\mathcal{E}) = \sum_{E \in \mathcal{E}} \#_u(E) P(E|F), \tag{7.10}$$

so that the statistic $c(u|\mathcal{E})$ counts the number of times an n-gram occurs on each path, accumulating the weighted count over all paths. By contrast, what is needed by the approximation in Equation (7.7) is to identify all paths containing an n-gram and accumulate their probabilities. The accumulation of probabilities at the path level, rather than the n-gram level, is what makes the exact computation of $p(u|\mathcal{E})$ difficult.

The implementation of lattice MBR in Tromble et al. (2008) (Section 7.1.4) computes each n-gram path posterior probability by composing the lattice with a finite-state acceptor for a single n-gram. Their approach is referred to here as the *sequential method*, since $p(u|\mathcal{E})$ is calculated separately for each u in sequence. Computing the posterior probabilities for the full set of lattice n-grams requires $|\mathcal{N}|$ separate compositions and log semiring weight pushing operations. This can be slow when the lattice contains a large number of n-grams.

Allauzen et al. (2010) introduce a transducer for simultaneous calculation of $p(u|\mathcal{E})$ for all unigrams $u \in \mathcal{N}_1$ in a lattice. This transducer is effective for finding path posterior probabilities of unigrams because there are relatively few unique unigrams in the lattice. As will be shown, however, it is less efficient for higher-order n-grams.

The lattice MBR decoder of Allauzen et al. (2010) uses the exact n-gram path posterior probabilities of Equation (7.9) to compute the unigram contribution to the expected gain, but uses the conditional expected counts of Equation (7.10) for higher-order n-grams:

$$\hat{E} = \operatorname*{argmax}_{E' \in \mathcal{E}} \left\{ \theta_0 |E'| + \sum_{u \in \mathcal{N}_1} \theta_u \#_u(E') p(u|\mathcal{E}) + \sum_{k=2}^{4} \sum_{u \in \mathcal{N}_k} \theta_u \#_u(E') c(u|\mathcal{E}) \right\} \tag{7.11}$$

Equation (7.11) is thus an approximation to the approximation. In many cases it will be perfectly fine, depending on how closely $p(u|\mathcal{E})$ and $c(u|\mathcal{E})$ agree for higher-order n-grams. Experimentally, Allauzen et al. (2010) show this approximation to work well in lattice minimum Bayes-risk decoding of statistical machine translation lattices. However, there may be scenarios in which $p(u|\mathcal{E})$ and $c(u|\mathcal{E})$ differ so that Equation (7.11) is no longer useful in place of the original Tromble et al. (2008) approximation. Section 7.3.2.3 will show that the exact n-gram path posterior probabilities must be used for orders $n = 1$ and $n = 2$ to obtain optimal Arabic→English lattice MBR decoding performance.

The following sections describe a path counting transducer that enables efficient simultaneous computation of $p(u|\mathcal{E})$ for all n-grams of a fixed order, and an acceptor for fast decoding with a similar form to the WFST implementation of an n-gram language model (Allauzen et al., 2003). Fast MBR decoding is applied to large statistical machine translation lattices in Section 7.3.3, where it is shown to offer significant improvements in efficiency over the sequential method of Tromble et al. (2008).

7.2.2 N-gram Mapping Transducer

A useful transformation can be applied to the evidence space in order to simplify the counting of paths containing higher-order n-grams. Transducer Φ_n is constructed to map word sequences to n-gram sequences of order n. Φ_n has a similar form to the WFST implementation of a backoff n-gram language model (Allauzen et al., 2003). In addition to arcs from the start state mapping the words of each distinct n-gram history to ϵ, Φ_n includes for each n-gram $u = w_1^n$ of order n arcs of the form shown in Figure 7.4.

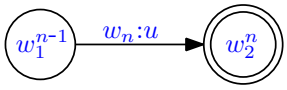
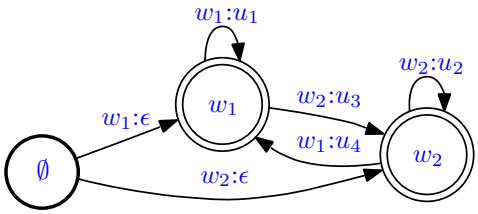


Figure 7.4: Mapping transducer arc example for the n-gram $u = w_1^n$.

The n-gram lattice of order n is called $\mathcal{E}_n$ and is found by composing $\mathcal{E} \circ \Phi_n$, projecting on the output, removing ϵ-arcs, determinizing, and minimising. The construction of $\mathcal{E}_n$ is fast even for large lattices and is memory efficient. $\mathcal{E}_n$ itself may have more states than $\mathcal{E}$ due to the association of distinct n-gram histories with states. However, the counting transducer for unigrams is much simpler than the corresponding counting transducer for higher-order n-grams. As a result, counting unigrams in $\mathcal{E}_n$ is easier than counting n-grams in $\mathcal{E}$.

Figure 7.5 shows a bigram mapping transducer example Φ_2. This transducer can be used to transform a word lattice $\mathcal{E}$ to a lattice of bigrams $\mathcal{E}_2$. In composition, word sequences $\{w_{1,2}\}^*$ in $\mathcal{E}$ are transformed to bigram sequences $\{u_{1,2,3,4}\}^*$ in $\mathcal{E}_2$, according to the table of bigrams on the right of the figure.



N-gram	Words
u_1	w_1w_1
u_2	w_2w_2
u_3	w_1w_2
u_4	w_2w_1

Figure 7.5: Mapping transducer Φ_2 for all possible bigrams $\Sigma_2 = \{u_1, u_2, u_3, u_4\}$ formed from lattice alphabet $\Sigma_1 = \{w_1, w_2\}$. States and arcs need only be added for bigrams $u \in \mathcal{N}_2$.

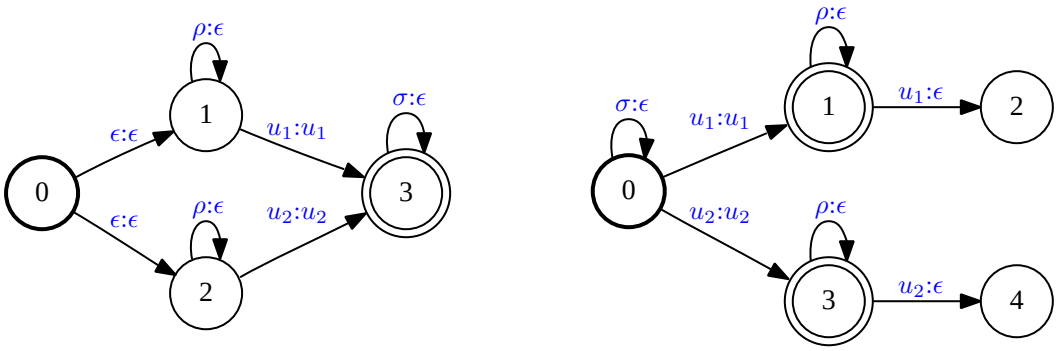


Figure 7.6: Path counting transducers Ψ_n^L (left) and Ψ_n^R (right) to match each $u \in \mathcal{N}_n$.

7.2.3 Efficient Path Counting

Associated with each $\mathcal{E}_n$ is a transducer Ψ_n that can be used to calculate the path posterior probabilities $p(u|\mathcal{E})$ for all $u \in \mathcal{N}_n$. Figure 7.6 shows two possible forms of Ψ_n that can be used to compute path posterior probabilities over n-grams $u_{1,2} \in \mathcal{N}_n$ for some n. Examples showing the sequence of operations required to count paths in $\mathcal{E}_n$ using Ψ_n^L and Ψ_n^R are given in Section 7.2.3.2. The special symbols ρ and σ are described in Chapter 2, Section 2.4.2.

Transducer Ψ_n^L is used by Allauzen et al. (2010) to compute the exact unigram contribution to the gain in Equation (7.11). For example, in counting paths that contain u_1, Ψ_n^L retains the *first* occurrence of u_1 and maps every other symbol to ϵ. This ensures that in any path containing a given u, only the first u is counted, avoiding multiple counting of paths.

A more efficient path counting transducer Ψ_n^R is now introduced. Transducer Ψ_n^R effectively deletes all symbols except the *last* occurrence of u on any path by ensuring that any paths in composition which match earlier instances do not end in a final state. Multiple counting is avoided by counting only the last occurrence of each symbol u on a path.

The reason why Ψ_n^L is inefficient for large $\mathcal{N}_n$ is that the initial $\epsilon{:}\epsilon$ arcs in Ψ_n^L effectively create $|\mathcal{N}_n|$ copies of $\mathcal{E}_n$ in composition while searching for the first occurrence of each u. Composing with Ψ_n^R creates only a single copy of $\mathcal{E}_n$ while searching for the last occurrence of u; this is found to be much more efficient for large $\mathcal{N}_n$.

Path posterior probabilities are calculated over each $\mathcal{E}_n$ by composing with Ψ_n in the log semiring, projecting on the output, removing ϵ-arcs, determinizing, minimising, and pushing weights to the initial state (Allauzen et al., 2010). Using either Ψ_n^L or Ψ_n^R, the resulting counts

acceptor is $\mathcal{X}_n$. It has a compact form with arcs from the start state for each $u_i \in \mathcal{N}_n$. Each arc has the form shown in Figure 7.7 with weight $-\log p(u_i|\mathcal{E})$.

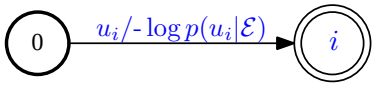


Figure 7.7: Optimised counts acceptor arc example for the n-gram u.

7.2.3.1 Efficient Path Posterior Computation

Although $\mathcal{X}_n$ has a convenient and elegant form, it can be difficult to build for large $\mathcal{N}_n$ because the composition $\mathcal{E}_n \circ \Psi_n$ results in millions of states and arcs. The log semiring ϵ-removal and determinization required to sum the probabilities of paths labelled with each u can be slow. This section describes an optimisation based on the forward procedure that enables more efficient calculation of n-gram path posterior probabilities.

If the transducer Ψ_n^R is used instead of Ψ_n^L, then each path in $\mathcal{E}_n \circ \Psi_n^R$ has only one non-ϵ output label u and all paths leading to a given final state share the same u. A modified forward procedure can be used to calculate $p(u|\mathcal{E})$ without costly ϵ-removal and determinization.

The modification to the forward procedure simply requires keeping track of which symbol u is encountered along each path to a final state. Let the forward variable $\alpha[q]$ denote the negative log of the sum of the probabilities of all partial paths to state q, and let $u[q]$ denote the output label shared by all paths passing through state q. The forward procedure using these variables is shown in Figure 7.8. For each $q \in Q$, the variables $\alpha[q]$ and $u[q]$ are initialised to $\bar{0}$ and ϵ, apart from the start state which is initialised to $\bar{1}$ (lines 1–3). Then, the forward variable $\alpha[n[e]]$ at the target state of each edge is incremented by the $\otimes$-product of the value of the forward variable at the source state $\alpha[q]$ and the arc weight $w[e]$ (line 6). The symbol at the target state is the output label $o[e]$ if $o[e] \neq \epsilon$, or propagated from the source state otherwise (line 7). When the forward procedure is completed, the n-gram path posterior probabilities are computed from the values of the forward variable $\alpha[q]$ and final state weight $\rho[q]$ at each final state $q \in F$. More than one final state may gather probabilities for the same u; to compute $p(u|\mathcal{E})$ these probabilities are added:

$$-\log p(u_i|\mathcal{E}) = \bigoplus_{q \in F : u[q] = i} \{\alpha[q] \otimes \rho[q]\} \tag{7.12}$$

The forward procedure requires that the counts transducer $\mathcal{C}_n = \mathcal{E}_n \circ \Psi_n^R$ be topologically sorted; although sorting can be slow, the forward procedure is $\mathcal{O}(V + E)$ which is normally more efficient than ϵ-removal and determinization of a large composition result.

Unlike the composition $\mathcal{E}_n \circ \Psi_n^R$, the composition $\mathcal{E}_n \circ \Psi_n^L$ does not segregate paths by u such that there is a direct association between final states and symbols. The forward procedure cannot be applied, but an arc weight vector indexed by symbols could be used to correctly aggregate probabilities (Riley et al., 2009). For large $\mathcal{N}_n$ this would be memory intensive. The association between final states and symbols could also be found by label pushing (Mohri et al., 2008), but this can be very slow for large $\mathcal{E}_n \circ \Psi_n$.

FORWARD-PROCEDURE($\mathcal{C}_n$)

```
1  for each state q ∈ Q[C_n]
2        do α[q] ← 0̄; u[q] ← ϵ
3  α[0] ← 1̄
4  for each state q ∈ Q[C_n]
5        do for each arc e ∈ E[q]
6              do α[n[e]] ← α[n[e]] ⊕ (α[q] ⊗ w[e])
7                 if o[e] ≠ ϵ then u[n[e]] ← o[e] else u[n[e]] ← u[q]
```

Figure 7.8: Modified forward procedure for computing path posterior probabilities.

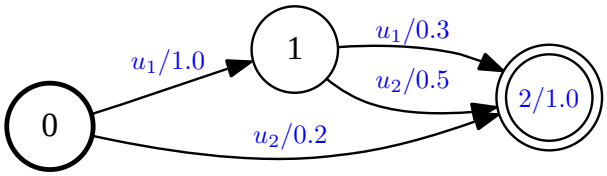


Figure 7.9: Toy lattice $\mathcal{E}_n$ encoding three distinct n-gram hypothesis sequences.

7.2.3.2 Path Counting Transducer Examples

This section shows how the path counting transducers Ψ_n^L and Ψ_n^R can be used to count weighted paths. Consider the lattice $\mathcal{E}_n$ shown in Figure 7.9. All arc weights in this section are shown in the real $(+, \times)$ semiring for clarity. This lattice encodes three distinct n-gram sequences with the following posterior probabilities:

Sequence	$p(E\|F)$
$E_1 = u_1 u_2$	0.5
$E_2 = u_1 u_1$	0.3
$E_3 = u_2$	0.2

The n-gram path posterior probabilities $p(u|\mathcal{E})$ of Equation (7.9) and expected counts $c(u|\mathcal{E})$ of Equation (7.10) computed from the lattice are shown below:

	$p(u\|\mathcal{E})$	$c(u\|\mathcal{E})$
u_1	0.8	1.1
u_2	0.7	0.7

The values of $p(u_2|\mathcal{E})$ and $c(u_2|\mathcal{E})$ agree because there are no paths in $\mathcal{E}_n$ with multiple occurrences of u_2. The conditional expected count of u_1 is $c(u_1|\mathcal{E}) = 1 \times 0.5 + 2 \times 0.3 = 1.1$ which differs from the n-gram path posterior probability $p(u|\mathcal{E}) = 0.5 + 0.3 = 0.8$ because u_1 occurs twice on the path $E_2 = u_1 u_1$.

The sequence of operations used to compute expected counts and n-gram path posterior probabilities is shown in Figure 7.10 on page 82. Let Ψ_n^N denote the regular n-gram counting transducer for computing expected counts (Allauzen et al., 2003). The top row shows – from

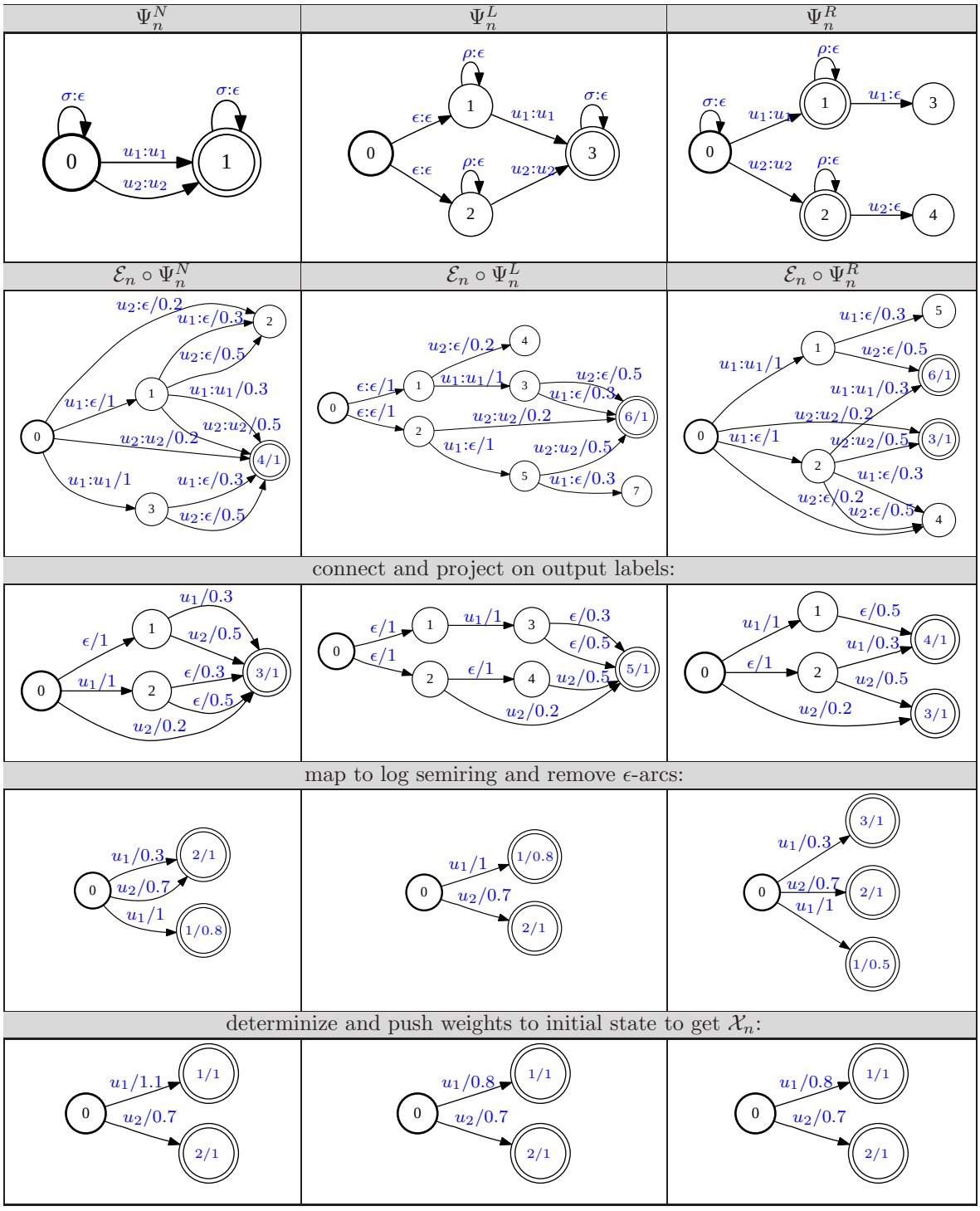


Figure 7.10: Weighted counting operations for n-gram counting transducer Ψ_n^N, and left-most Ψ_n^L and right-most Ψ_n^R path-counting transducers. Weights of arcs and final states in all weighted automata are shown for the real $(+, \times)$ semiring.

left to right – the expected counts transducer Ψ_n^N, the left-most matching path counting transducer Ψ_n^L, and the right-most matching path counting transducer Ψ_n^R.

The second row shows the result of composing each of these transducers with the weighted lattice $\mathcal{E}_n$. The composition $\mathcal{E}_n \circ \Psi_n^N$ shows how u_1 is counted twice on the path $E_2 = u_1 u_1$ in $\mathcal{E}_n$. The composition state sequence $(0,0) \rightarrow (0,1) \rightarrow (1,4)$ corresponds to matching the first u_1 with the σ:ϵ transition in Ψ_n^N and the second u_1 with the transition u_1:u_1; the state sequence $(0,0) \rightarrow (1,3) \rightarrow (1,4)$ corresponds to matching the first u_1 with the u_1:u_1 transition in Ψ_n^N and the second u_1 with the σ:ϵ transition in the final state of Ψ_n^N. Since both matching paths write u_1 on the output label, u_1 is counted twice, each time with weight 0.3.

In both $\mathcal{E} \circ \Psi_n^L$ and $\mathcal{E}_n \circ \Psi_n^R$, however, the symbol u_1 is counted only once per path. Ψ_n^R counts only the last occurrence of u_1 on the path $E_2 = u_1 u_1$. The state sequence $(0,0) \rightarrow (1,1) \rightarrow (3,5)$ counts the first u_1 by taking the u_1:u_1 transition in Ψ_n^R but the second u_1 is also matched, leading to a non-final state; the path in the composition result that matches the first occurrence of u_1 therefore contributes nothing to the count. The state sequence $(0,0) \rightarrow (0,2) \rightarrow (1,6)$ maps the first u_1 to ϵ by taking the u_1:ϵ transition from the initial state of Ψ_n^R, and counts the second u_1 by then taking the u_1:u_1 transition to the final state of Ψ_n^R. This avoids multiple counting of the same symbol on each path.

The third row shows the results of connecting and projecting on the output labels. Note that for $\mathcal{E}_n \circ \Psi_n^L$, a mixture of u_1 and u_2 labelled paths lead to the final state q_5. For $\mathcal{E}_n \circ \Psi_n^R$, only paths labelled u_1 lead to final state q_4 and only paths labelled u_2 lead to final state q_3. It is this segregation of symbols and final states that allows the use of the modified forward procedure described in Section 7.2.3.1.

The remaining rows show the optimisation operations that are used to sum the matched counts to obtain $\mathcal{X}_n$. If Ψ_n^R is used then these optimisation operations can be omitted and the modified forward procedure applied directly to the composition result $\mathcal{E}_n \circ \Psi_n^R$.

7.2.4 Efficient Decoder Implementation

This section describes an efficient implementation of the linearised lattice MBR decoder decision rule. In contrast to Equation (7.11), the exact values of $p(u|\mathcal{E})$ for all $u \in \mathcal{N}_n$ at orders $n = 1 \ldots 4$ are used to compute

$$\hat{E} = \underset{E' \in \mathcal{E}}{\operatorname{argmin}} \left\{ \theta_0 |E'| + \sum_{n=1}^{4} g_n(E, E') \right\}, \tag{7.13}$$

where $g_n(E, E') = \sum_{u \in \mathcal{N}_n} \theta_u \#_u(E') p(u|\mathcal{E})$ is the contribution to the conditional expected gain from n-grams of order n. An acceptor Ω_n is constructed so that $\mathcal{E} \circ \Omega_n$ assigns order n partial gain $g_n(E, E')$ to all paths $E \in \mathcal{E}$. Ω_n is derived from the mapping transducer Φ_n by assigning arc weight $\theta_u \times p(u|\mathcal{E})$ to arcs with output label u and then projecting on the input labels. The algorithm in Figure 7.13 performs this procedure. For each n-gram $u = w_1^n$ in $\mathcal{N}_n$ arcs of Ω_n have the form shown in Figure 7.11.

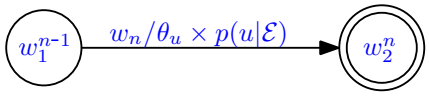
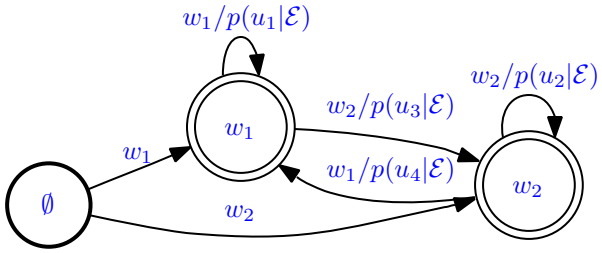


Figure 7.11: Decoder arc example to apply partial gain associated with n-gram $u = w_1^n$.



N-gram	Words
u_1	w_1w_1
u_2	w_2w_2
u_3	w_1w_2
u_4	w_2w_1

Figure 7.12: Decoding acceptor Ω_2 for all possible bigrams $\Sigma_u = \{u_1, u_2, u_3, u_4\}$ formed from lattice alphabet $\Sigma_w = \{w_1, w_2\}$. Ω_2 is derived directly from the mapping transducer Φ_u.

MAKE-DECODER-FST(Φ_n)
1 $\Omega_n = \Phi_n$
2 **for each** state $q \in Q[\Omega_n]$
3 **do for each** arc $e \in E[q] : o[e] \neq \epsilon$
4 **do** $u \leftarrow o[e]$; $w[e] \leftarrow \theta_u \times p(u|\mathcal{E})$
5 **return** PROJECT(Ω_n, input)

Figure 7.13: Algorithm to build decoding automaton Ω_n. The input is the unweighted mapping transducer Φ_n for mapping a word lattice to a lattice of order-n sequences.

Figure 7.12 shows a decoding acceptor example Ω_2 derived from the n-gram mapping transducer Φ_2 in Figure 7.5.[1] Decoding with the acceptors Ω_n, $n = 1 \ldots 4$ requires only four compositions; this is much more efficient than the sequential method of Section 7.1.4 which requires a separate composition with the acceptor Ω_u for each $u \in \mathcal{N}$.

Decoding proceeds as follows. To apply θ_0 a copy is made of $\mathcal{E}$, called $\mathcal{E}_0$, with fixed weight θ_0 on all arcs. The decoder is formed as the composition chain

$$\mathcal{E}_0 \circ \Omega_1 \circ \Omega_2 \circ \Omega_3 \circ \Omega_4, \tag{7.14}$$

and the translation hypothesis $\hat{E}$ that maximises the conditional expected gain is extracted as the maximum cost string. The maximum cost string is easily extracted by multiplying all arc weights by -1 and using the shortest path algorithm in the tropical semiring.

7.3 Lattice MBR Decoding Experiments

This section describes large lattice minimum Bayes-risk decoding performance and efficiency experiments. For Arabic→English translation, single-system lattice-based minimum Bayes-risk decoding is evaluated within the framework of the NIST MT08 machine translation task.[2] The development set mt0205tune is formed from the odd numbered sentences of the MT02–MT05 evaluation sets; the even numbered sentences form the validation set mt0205test. Test performance is measured on the MT08 sets: mt08nw for newswire data and mt08ng for

[1] The n-gram factors θ_u are omitted for clarity.
[2] http://www.itl.nist.gov/iad/mig/tests/mt/2008/

newsgroup data. For Chinese→English translation, the testsets are those of the GALE P3 evaluation and include separate development and evaluation sets for newswire and web data. The Chinese→English tuning sets exclude the sentences from NIST MT08; these are reserved for the evaluation sets mt08.text.nw (newswire data) and mt08.text.web (web data). All BLEU scores and TER are reported for uncased translations. Tables 7.1 and 7.2 summarise the number of sentences and genre of these testsets.

AR→EN Testset	Genre	Sentences
mt0205tune	news	2075
mt0205test	news	2040
mt08nw	news	813
mt08ng	web	547

Table 7.1: Development and testsets for NIST MT08 Arabic→English translation.

ZH→EN Testset	Genre	Sentences
tune.text.nw	news	1755
mt08.text.nw	news	691
tune.text.web	web	2495
mt08.text.web	web	666

Table 7.2: Development and testsets for GALE P3 Chinese→English translation.

7.3.1 System Development

For Arabic→English translation, word alignments are generated using MTTK (Deng and Byrne, 2008) over approximately 150M words of parallel text specified for the constrained NIST MT08 Arabic→English track. Prior to generating the alignments, the Arabic side of the parallel text is pre-processed with the MADA morphological toolkit (Habash and Rambow, 2005). The word alignments for Chinese→English translation are trained from nearly 250M words of parallel text distributed for the GALE P3 evaluation by BBN Technologies. The source side of the parallel data is pre-processed with a Chinese word segmentation algorithm prior to generating the alignments.

For both language pairs, hierarchical rules are extracted from the aligned text using the constraints described in Chiang (2007) with the count and pattern filters of Iglesias et al. (2009a). First-pass translation decoding with HiFST (Iglesias et al., 2009b) (Section 4.3.3) generates word lattices encoding large numbers of alternative hypotheses. A *shallow-1* grammar (de Gispert et al., 2010) is used for Arabic→English decoding. With this grammar, only a single level of rule nesting is allowed and no pruning is required in search. Chinese→English decoding supports arbitrary nesting with a fully hierarchical grammar. In practice, the degree of nesting is indirectly constrained by setting the maximum number of words that may be covered by each non-terminal. The larger space of translations encoded by the Chinese→English grammar requires pruning during search.

For both systems, minimum error rate training (Och, 2003) under the BLEU score (Papineni et al., 2002b) optimises the following list of features with respect to the development set: target language model, source→target and target→source rule translation models, word and

Configuration	Newswire Data						Web Data	
	mt0205tune		mt0205test		mt08nw		mt08ng	
	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
HiFST+5g	54.23	40.49	53.78	41.02	51.35	43.47	36.31	54.92
+MBR	54.58	40.29	54.31	40.73	51.82	43.34	36.45	54.97
+LMBR	54.99	39.91	54.55	40.50	52.25	43.10	36.79	54.64

Table 7.3: BLEU scores and TER for k-best and lattice MBR of NIST MT08 Arabic→English evaluation sets. The lists used for k-best MBR contain 1000 hypotheses.

Configuration	Newswire Data				Web Data			
	tune.text.nw		mt08.text.nw		tune.text.web		mt08.text.web	
	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
HiFST+5g	34.46	59.25	34.55	57.71	17.06	70.46	22.25	61.40
+MBR	34.93	59.15	34.93	57.51	17.31	70.11	23.01	61.12
+LMBR	35.00	59.23	35.13	57.44	17.38	70.56	23.83	60.71

Table 7.4: BLEU scores and TER for k-best and lattice MBR of GALE P3 Chinese→English evaluation sets. The lists used for k-best MBR contain 1000 hypotheses.

rule penalties, number of usages of the glue rule, source→target and target→source lexical translation probabilities, and three count-based features that track the observed frequency of rules in the parallel data (Bender et al., 2007). The English language model used during decoding is a modified Kneser-Ney (Kneser and Ney, 1995) smoothed 4-gram estimated over the English side of the parallel text and a 465M word subset of the English GigaWord Third Edition (Graff et al., 2007).

The first-pass HiFST lattices are rescored with large 5-gram sentence-specific zero-cutoff stupid-backoff language models (Brants et al., 2007) estimated over a collection of more than six billion words of English language training text (Blackwood et al., 2009). For k-best list MBR, the top 1000 hypotheses are extracted from each of the first-pass translation lattices.

The posterior distribution scaling parameter α and per-word factor θ_0 in the lattice MBR decoder of Equation (7.7) are optimised with respect to the development set: mt0205tune for Arabic→English translation, and tune.text.nw or tune.text.web for Chinese→English translation.

7.3.2 Lattice MBR Results and Analysis

Tables 7.3 and 7.4 show single-system LMBR decoding baselines for Arabic→English and Chinese→English translation. The first row of each table shows the 5-gram rescored first-pass translation ML 1-best BLEU score and TER (Section 4.4.4). Row MBR shows the gains from regular k-best MBR and row LMBR shows the gains from lattice-based MBR.

The results show that for Arabic→English translation, both MBR and LMBR provide good gains over the ML 1-best. LMBR gives absolute gains of between +0.2 and +0.4 BLEU compared to k-best MBR on each of the development and testsets. Overall gains from LMBR compared to the ML 1-best are about +0.8 BLEU for newswire data and +0.5 BLEU for web data. These results validate the linearised BLEU approximation of Equation (7.7), and show that decoding can be applied to lattices containing even very large numbers of translation

		mt0205tune			mt0205test		
		BLEU	TER	BP	BLEU	TER	BP
HiFST+5g		54.23	40.49	0.995	53.78	41.02	0.994
+LMBR	$p = 1$	54.36	40.46	0.996	54.02	40.91	0.995
	$p = 2$	54.46	40.51	0.998	54.23	40.94	0.997
	$p = 3$	54.67	40.20	0.991	54.33	40.63	0.994
	$p = 4$	54.70	40.07	0.992	54.42	40.48	0.992
	$p = 5$	54.82	39.92	0.990	54.42	40.41	0.989
	$p = 6$	54.97	39.74	0.987	54.48	40.35	0.987
	$p = 7$	54.96	39.67	0.985	54.50	40.26	0.984

Table 7.5: BLEU score, TER and brevity penalty (BP) for LMBR decoding of NIST MT08 Arabic→English testsets at a range of lattice likelihood pruning thresholds p.

hypotheses. For Chinese→English MBR decoding, lattice MBR improves only a little over k-best MBR, with larger gains in some places.

7.3.2.1 Likelihood Pruning and MBR Decoding Performance

The effect of likelihood pruning on lattice MBR for Arabic→English translation of mt0205tune and mt0205test is shown in Table 7.5. The parameter p specifies a negative log probability pruning threshold that is used to prune hypotheses relative to the best translation in the lattice. Likelihood pruning is applied using the WFST prune operation (Allauzen et al., 2007) (Chapter 4, Section 4.2.2). Larger values of p indicate larger thresholds and thus pruning of fewer hypotheses; the heaviest pruning occurs at $p = 1$. It is clear from the results in the table that the BLEU score is maximised by pruning as little as possible. Comparing these results with the k-best MBR gains in Table 7.3 shows that lattice MBR achieves as large a gain as k-best MBR even when the lattices are pruned to $p = 3$. These results show that the k-best lists contain only a relatively small subset of the hypotheses encoded in the lattice and that these additional hypotheses are useful for MBR decoding.

7.3.2.2 Evidence Space Size and MBR Decoding Performance

One of the main reasons why lattice MBR decoding performs so much better than k-best list MBR decoding is that it is able to exploit a much larger evidence space of translations (Tromble et al., 2008). The purpose of the following experiment is to show that k-best list MBR decoding is limited because the k-best lists often represent a surprisingly small fraction of the total probability mass in the lattice.

Let $\mathcal{E}_e$ denote the full evidence space of the lattice and $\mathcal{E}_k$ the k-best list of hypotheses obtained from it. If $\phi(\mathcal{E}) = \sum_{E \in \mathcal{E}} P(E|F)$ sums the posterior translation probabilities of all hypotheses in $\mathcal{E}$ (computed according to Equation (7.4)), then the proportion of lattice probability mass contained in the k-best list is the ratio $\phi(\mathcal{E}_k)/\phi(\mathcal{E}_e)$. It follows that the proportion of lattice probability mass missing from the list is $1-\phi(\mathcal{E}_k)/\phi(\mathcal{E}_e)$. These statistics can be computed exactly by converting the k-best list to a lattice and pushing weights to the final state in the log semiring (Chapter 2, Section 2.4.3).

Figure 7.14 plots the proportion of lattice probability mass missing from k-best lists of size $k = 1000$ hypotheses (top) and $k = 20000$ hypotheses (bottom) as a function of the number of

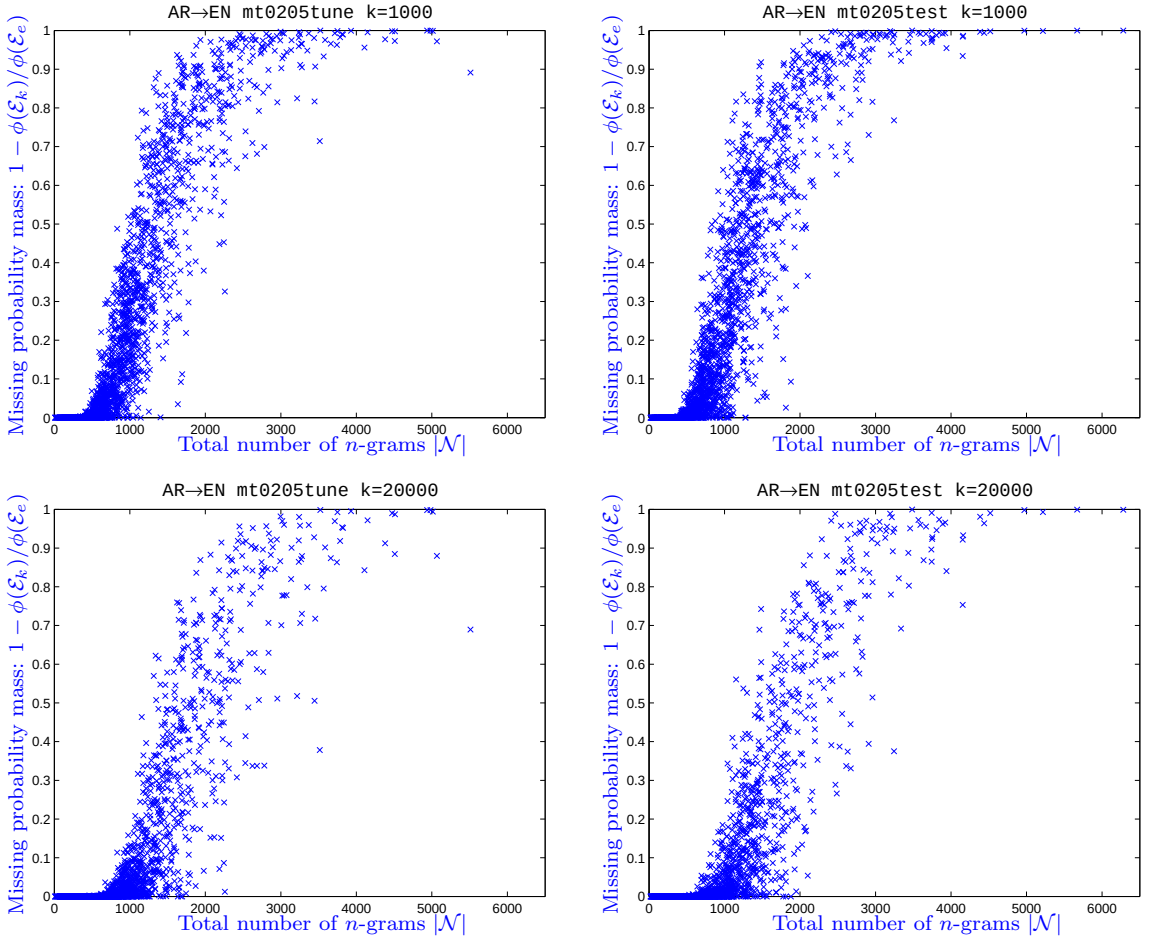


Figure 7.14: Proportion of lattice probability mass $1 - \phi(\mathcal{E}_k)/\phi(\mathcal{E}_e)$ missing from k-best lists of size $k = 1000$ (top) and $k = 20000$ (bottom) as a function of the number of lattice n-grams $|\mathcal{N}|$ for the NIST MT08 Arabic→English mt0205tune and mt0205test translation testsets.

lattice n-grams $|\mathcal{N}|$ for the Arabic→English mt0205tune and mt0205test testsets. The lattices for this experiment were generated at a likelihood pruning threshold of $p = 7$. For $k = 1000$, about half of the sentences in each testset have 1000 or fewer hypotheses and therefore use the same space for lattice and k-best list MBR decoding (for these sentences $\mathcal{E}_n = \mathcal{E}_e$). However, the plots show that there are many sentences for which the top 1000 hypotheses accounts for only a relatively small proportion of the total lattice probability mass. For example, 111 sentences of mt0205tune and 122 sentences of mt0205test (approximately 5% of each testset) have 1000-best lists that account for less than 10% of the lattice probability mass; this means that more than 90% of the probability mass distributed amongst the hypotheses in the lattice evidence space is missing from the 1000-best lists and therefore ignored during 1000-best list MBR decoding. Comparing $k = 1000$ and $k = 20000$ shows that longer k-best lists account for a larger proportion of the lattice probability mass. However, there are still a fair number of sentences, particularly the longer sentences, for which $k = 20000$ lists account for less than 50% of the total lattice probability mass. Table 7.6 shows the average proportion of missing

k	mt0205tune	mt0205test
1000	24.41	24.91
10000	13.96	14.27
20000	11.73	12.00
50000	9.30	9.52
100000	7.78	7.98

Table 7.6: Average proportion (%) of missing probability mass by k-best list size for the NIST MT08 Arabic→English mt0205tune and mt0205test translation testsets.

probability mass in k-best lists of various sizes for mt0205tune and mt0205test.

Although hypotheses in the lattice that are missing from the k-best lists may have low probability with respect to the ML 1-best translation, there are so many of them that their aggregate probability is significant and can be usefully exploited to improve translation quality through lattice minimum Bayes-risk decoding. The investigation of evidence space size in this section shows why k-best list minimum Bayes-risk decoding does not normally perform so well. The full space of the lattice is required for good performance.

7.3.2.3 Hybrid Decision Rule Accuracy

The hybrid decision rule for linearised lattice minimum Bayes-risk decoding (Allauzen et al., 2010) can be written as

$$\hat{E} = \underset{E' \in \mathcal{E}}{\operatorname{argmax}} \left\{ \theta_0 |E'| + \sum_{u \in \mathcal{N}: 1 \leq |u| \leq k} \theta_{|u|} \#_u(E') p(u|\mathcal{E}) + \sum_{u \in \mathcal{N}: k < |u| \leq N} \theta_{|u|} \#_u(E') c(u|\mathcal{E}) \right\}, \quad (7.15)$$

where k determines the range of n-gram orders at which the path posterior probabilities $p(u|\mathcal{E})$ (Equation (7.9)) and conditional expected counts $c(u|\mathcal{E})$ (Equation (7.10)) are used to compute the conditional expected gain. Note that when $k = 0$ the conditional expected counts are used for all orders, and when $k = 4$ the path posterior probabilities are used for all orders.

The following experiment investigates the accuracy of the hybrid decision rule approximation in lattice MBR decoding. Tables 7.7 and 7.8 show BLEU scores for Arabic→English and Chinese→English first-pass ML 1-best translations (row ML), regular linearised lattice MBR (row LMBR) (Equation (7.7)), and scores obtained using the hybrid decision rule of Equation (7.15) for $0 \leq k \leq 4$. The optimised LMBR parameters for Arabic→English translation are $\alpha = 0.4$ and $\theta_0 = -0.02$ for newswire data, and $\alpha = 0.6$ and $\theta_0 = 0.01$ for newsgroup data. For Chinese→English translation, $\alpha = 0.4$ for both newswire and web data, with $\theta_0 = -0.02$ for newswire data and $\theta_0 = +0.10$ for web data.

For both Arabic→English and Chinese→English translation, the hybrid decision rule performs poorly when $k = 0$ so that the conditional expected counts are used for all orders. The $k = 0$ hybrid decoding scores are considerably lower than even the ML 1-best scores. This poor performance is because there are many unigrams u for which $c(u|\mathcal{E})$ is much greater than $p(u|\mathcal{E})$. The selection of the consensus translation maximising the conditional expected gain is then dominated by unigram matches: this has a big impact on LMBR decoding performance. If the posterior probabilities are used for unigrams and conditional expected counts are used for all other orders (i.e. $k = 1$), then Chinese→English hybrid MBR decoding performs as

		mt0205tune	mt0205test	mt08nw	mt08ng
ML		54.2	53.8	51.4	36.3
k	0	52.6	52.3	49.8	34.5
	1	54.8	54.4	52.2	36.6
	2	54.9	54.5	52.4	36.8
	3	54.9	54.5	52.4	36.8
	4	55.0	54.6	52.4	36.8
LMBR		55.0	54.6	52.4	36.8

Table 7.7: BLEU scores for Arabic→English maximum likelihood translation, linearised lattice MBR, and MBR decoding using the hybrid decision rule at values of $k = 0 \ldots 4$.

		tune.text.nw	mt08.text.nw	tune.text.web	mt08.text.web
ML		34.5	34.6	17.1	22.3
k	0	33.5	34.0	17.0	23.7
	1	34.9	35.1	17.3	23.9
	2	35.0	35.1	17.4	23.9
	3	35.0	35.1	17.4	23.8
	4	35.0	35.1	17.4	23.8
LMBR		35.0	35.1	17.4	23.8

Table 7.8: BLEU scores for Chinese→English maximum likelihood translation, linearised lattice MBR, and MBR decoding using the hybrid decision rule at values of $k = 0 \ldots 4$.

well as regular linearised lattice MBR on the mt08.text.nw and mt08.text.web testsets. For Arabic→English translation, the hybrid decision rule is an accurate approximation only when $k \geq 2$. The exact contribution to the gain function must be computed using the path posterior probabilities for orders $n = 1$ and $n = 2$.

Figure 7.15 compares the n-gram path posterior probabilities and conditional expected counts with ratio $c(u|\mathcal{E})/p(u|\mathcal{E}) > 1.05$ for n-grams in a single sentence of the mt0205tune testset. The large differences between $p(u|\mathcal{E})$ and $c(u|\mathcal{E})$ for many unigrams shows why the hybrid decision rule is a poor approximation when $k = 0$. Some large bigram differences are also observed; this explains the slight degradation in Arabic→English hybrid LMBR decoding at $k = 1$. For higher-order n-grams, the conditional expected counts are an acceptable approximation since there are relatively few higher-order n-grams with significantly differing values of $p(u|\mathcal{E})$ and $c(u|\mathcal{E})$.

These experiments have shown that the hybrid decoder of Equation (7.15) is an acceptable approximation for Chinese→English LMBR decoding when $k = 1$, and for Arabic→English LMBR decoding when $k = 2$. These results differ from Allauzen et al. (2010) where the $k = 1$ hybrid decoder was reported to perform well for both language pairs. Since the suitability of the hybrid decoder depends on how closely $c(u|\mathcal{E})$ approximates $p(u|\mathcal{E})$, the fast path counting transducers proposed in Section 7.2 should be used to extract the exact statistics required to compute the n-gram path posterior probabilities at all orders.

n-gram order	ratio $\frac{c(u\|\mathcal{E})}{p(u\|\mathcal{E})}$	$p(u\|\mathcal{E})$	$c(u\|\mathcal{E})$	n-gram u
1g	10.54	1.00	10.54	the
	6.08	1.00	6.08	of
	5.24	1.00	5.24	,
	3.92	1.00	3.92	”
	2.72	1.00	2.71	which
	2.27	1.00	2.27	in
	2.18	1.00	2.18	and
	2.00	1.00	2.00	line
	2.00	1.00	2.00	one
	1.98	1.00	1.98	commitment
	1.98	1.00	1.98	hariri
	1.77	0.99	1.75	president
	1.73	0.95	1.64	to
	1.72	1.00	1.72	national
	1.70	0.97	1.66	-
	1.61	0.99	1.60	@-@
	1.59	0.95	1.51	is
	1.56	0.99	1.55	al
	1.43	0.84	1.20	last
	1.35	0.74	1.00	on
	1.29	1.00	1.29	.
	1.27	0.78	0.99	it
	1.18	1.00	1.18	with
	1.14	0.87	0.99	was
	1.11	1.00	1.11	him
	1.10	0.31	0.34	a
	1.08	0.21	0.23	that
	1.07	0.30	0.32	had
	1.05	0.32	0.33	has
2g	3.45	0.99	3.43	of the
	1.61	0.86	1.38	, which
	1.56	0.99	1.55	al @-@
	1.55	0.92	1.43	, and
	1.52	0.99	1.50	the ”
	1.42	0.99	1.41	@-@ hariri
	1.31	0.75	0.99	the last
	1.29	0.70	0.91	” which
	1.20	0.56	0.67	” ,
	1.15	0.92	1.05	to the
	1.10	0.63	0.69	which was
	1.09	0.73	0.80	the president
	1.09	0.29	0.32	line ,
	1.08	0.27	0.29	, ”
	1.06	0.99	1.06	the commitment
	1.06	0.98	1.03	commitment of
3g	1.42	0.99	1.41	al @-@ hariri
	1.20	0.56	0.67	” , which
	1.08	0.26	0.28	, ” which
	1.07	0.46	0.49	of the ”
	1.06	0.97	1.02	the commitment of
	1.06	0.93	0.98	commitment of the
4g	1.06	0.92	0.97	the commitment of the

Figure 7.15: Path posterior probabilities $p(u|\mathcal{E})$ and conditional expected counts $c(u|\mathcal{E})$ of n-grams with ratio $c(u|\mathcal{E})/p(u|\mathcal{E}) > 1.05$ for an Arabic→English mt0205tune testset sentence.

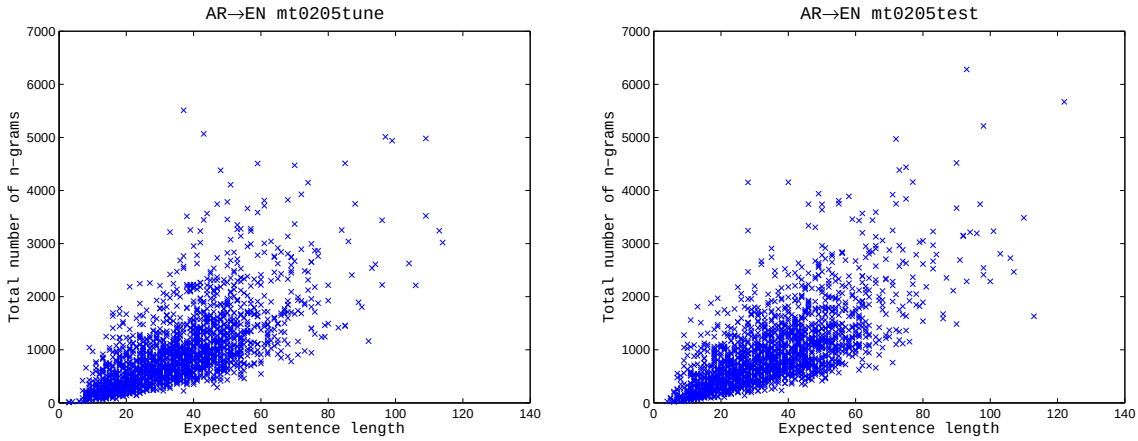


Figure 7.16: Expected target sentence length $\mathbb{E}(l)$ and total number of lattice n-grams ($p = 7$) for the NIST MT08 Arabic→English mt0205tune and mt0205test evaluation sets.

7.3.3 Lattice Minimum Bayes-Risk Decoding Efficiency

This section compares the lattice minimum Bayes-risk decoding efficiency of the sequential implementation (Tromble et al., 2008) described in Section 7.1.3, and the implementation based on path counting transducers proposed in Section 7.2.3. It should be noted that the sequential method and both simultaneous implementations Ψ_n^L and Ψ_n^R yield the same hypotheses (allowing for numerical accuracy); they differ only in speed and memory usage.

The time required for linearised LMBR decoding using the decision rule of Equation (7.7) is a function of the number of n-grams in the lattice. It is useful to examine how the number of n-grams varies as a function of the target sentence length. Figure 7.16 plots the total number of lattice n-grams against expected sentence length for each lattice in the NIST MT08 Arabic→English mt0205tune and mt0205test testsets. Most sentences have an expected length of less than 80 words and contain less than 3000 n-grams. LMBR decoding is fast for these sentences. Some sentences contain many more n-grams than expected. These are sentences with a high lattice branching factor resulting from the existence of many alternative translations and applications of hierarchical rules in first-pass decoding. Although quicker than k-best list MBR, computing many thousands of n-gram posterior probabilities from a large lattice one-by-one in sequence is inefficient.

The time in seconds required to compute the n-gram path posterior probabilities, time required to execute the MBR decision rule, and overall time, summed over each sentence of the Arabic→English and Chinese→English testsets, is shown in Tables 7.9 and 7.10.

7.3.3.1 Posteriors Efficiency

In calculating path posterior n-gram probabilities $p(u|\mathcal{E})$, the use of the left-most matching path counting transducer Ψ_n^L is found to be around twice as slow as the sequential method for both Arabic→English and Chinese→English lattice decoding. This is due to the difficulty of counting higher-order n-grams in large lattices. Ψ_n^L is clearly not an appropriate form of counting transducer for efficient lattice MBR. Using the right-most matching path counting transducer Ψ_n^R is nearly twice as fast as the sequential method for Arabic→English lattices,

		mt0205tune	mt0205test	mt08nw	mt08ng
Posteriors	sequential	3160	3306	2090	3791
	Ψ_n^L	6880	7387	4201	8796
	Ψ_n^R	1746	1789	1182	2787
Decoding	sequential	4340	4530	2225	4104
	Ψ_n	284	319	118	197
Overall	sequential	7711	8065	4437	8085
	Ψ_n^L	7458	8075	4495	9199
	Ψ_n^R	2321	2348	1468	3149

Table 7.9: Time in seconds for n-gram path posterior probability computation and decoding using sequential and left-most (Ψ_n^L) or right-most (Ψ_n^R) counting transducer implementations for Arabic→English translation testsets.

		tune.text.nw	mt08.text.nw	tune.text.web	mt08.text.web
Posteriors	sequential	7779	3974	11796	2581
	Ψ_n^L	12321	6208	19301	4341
	Ψ_n^R	2954	1525	4223	855
Decoding	sequential	10161	4725	16072	3215
	Ψ_n	245	91	352	77
Overall	sequential	18356	8899	28506	5922
	Ψ_n^L	13013	6503	20341	4554
	Ψ_n^R	3576	1795	5157	1047

Table 7.10: Time in seconds for n-gram path posterior probability computation and decoding using sequential and left-most (Ψ_n^L) or right-most (Ψ_n^R) counting transducer implementations for Chinese→English translation testsets.

and nearly three times faster for the much larger Chinese→English lattices, which contain many more n-grams. This difference in speed is due to the simultaneous computation of all n-grams of a fixed order in a single composition. The transducer Ψ_n^R is also designed so as to allow the use of an efficient forward algorithm. For higher-order n, the composition $\mathcal{E}_n \circ \Psi_n^R$ requires less memory and produces a smaller machine than $\mathcal{E}_n \circ \Psi_n^L$. This shows that it is easier to count weighted paths by the final occurrence of a symbol than by the first. Since much of the time in calculation is spent dealing with ϵ-arcs that are ultimately removed, an optimised composition algorithm that skips over such redundant structure may lead to further improvements in time efficiency.

7.3.3.2 Decoding Efficiency

Decoding times are significantly faster using Ω_n than the sequential method; average decoding time is just 0.1 seconds per sentence. Decoding is faster since only four compositions are required to assign the exact expected partial gain to all hypotheses in the lattice. The absence of ϵ-arcs and deterministic topology of Ω_n also allows for very fast composition.

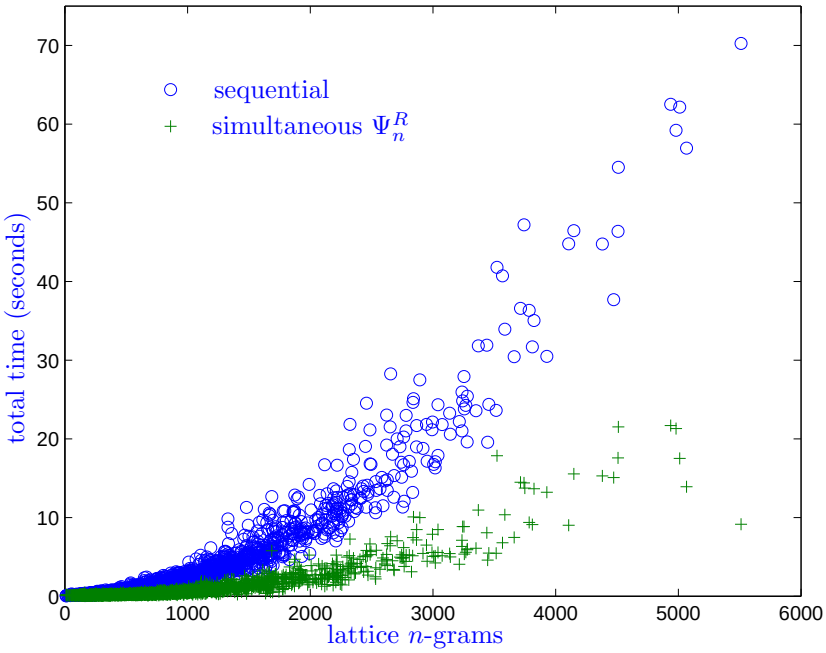


Figure 7.17: Total decoding time in seconds versus number of lattice n-grams for Arabic→English mt0205tune translation testset.

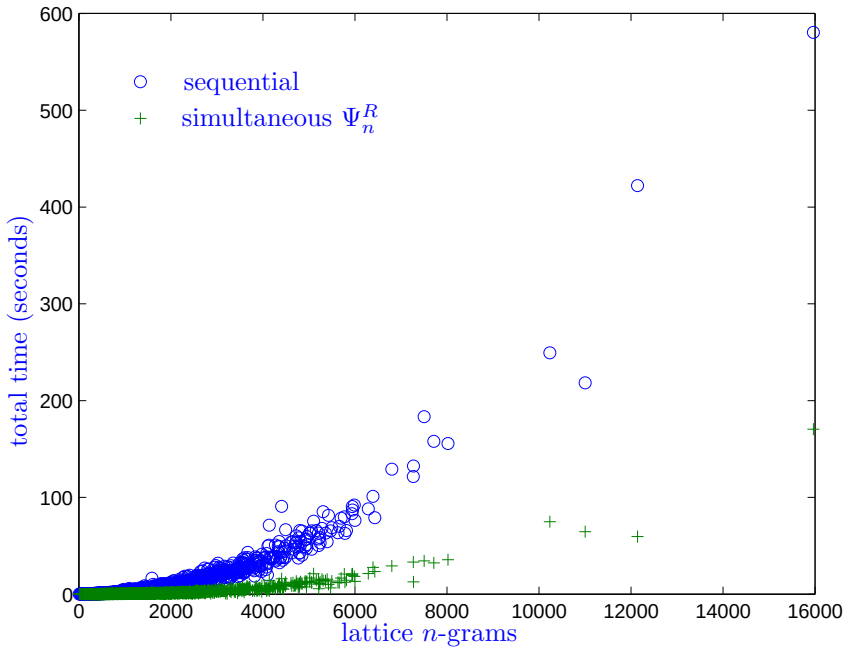


Figure 7.18: Total decoding time in seconds versus number of lattice n-grams for Chinese→English tune.text.nw translation testset.

7.3.3.3 Overall Efficiency

The overall MBR time is dominated by the calculation of the path posterior n-gram probabilities. This is a function of the number of n-grams in the lattice $|\mathcal{N}|$. For each sentence in the Arabic→English mt0205tune and Chinese→English tune.text.nw testsets, Figures 7.17 and 7.18 plot the total LMBR time for the sequential method (marked 'o') and for the implementation using the right-most matching efficient path counting transducer Ψ_n^R (marked '+'). This compares the two techniques on a sentence-by-sentence basis. As $|\mathcal{N}|$ grows, the simultaneous path counting transducer is found to be much more efficient. Although the path counting transducer Ψ_n^R requires the additional step of mapping to a lattice of n-gram sequences, a large proportion of the testset sentences can be processed more quickly. Using the sequential method, MBR decoding can be performed in one second or less for 768 out of 2075 Arabic→ mt0205tune sentences; using Ψ_n^R, 1608 sentences can be processed in one second or less. These results suggest the path counting transducer Ψ_n^R is a more appropriate implementation for real-time MBR translation decoding.

7.3.4 Summary and Conclusions

This chapter proposed an efficient and exact implementation of linearised lattice minimum Bayes-risk decoding using general purpose weighted finite-state transducer operations (Blackwood and Byrne, 2010). A mapping transducer was described for transforming sequences of words to sequences of n-grams, simplifying the extraction of higher-order statistics. A weighted path counting transducer Ψ_n^R was introduced that can be used to extract the required statistics for all n-grams of order n in a single composition. The topology of Ψ_n^R is designed to allow the path posterior probabilities to be efficiently accumulated using a modified version of the forward procedure.

The efficiency of the path counting transducer was evaluated on large Arabic→English and Chinese→English machine translation lattices, where it was shown to be nearly twice as fast as the sequential method of Tromble et al. (2008). The importance of a large lattice evidence space was demonstrated by examining the effect of pruning on MBR performance, and through a comparison of lattice decoding with regular k-best list decoding. Analysing the efficiency of the two forms of weighted path counting transducer shows that it is more efficient to count paths by the last occurrence of a symbol than by the first.

Even approximate search criteria should be implemented exactly where possible, so that it is clear exactly what the system is doing. For SMT lattices, conflating $p(u|\mathcal{E})$ and $c(u|\mathcal{E})$ may not be a serious problem, but for other scenarios – especially where symbol sequences are repeated multiple times on the same path – it may be a poor approximation. The efficient weighted path counting operations described in this chapter are general techniques that may prove useful in applications other than machine translation, whenever it is necessary to accumulate statistics at the path level rather than the symbol or symbol sequence level.

In the following chapter, lattice minimum Bayes-risk decoding is applied to the task of combining multiple statistical machine translation lattices generated from alternative analyses of the foreign source sentence.

Chapter 8

Lattice Minimum Bayes-Risk Decoding for System Combination

Different machine translation paradigms have different strengths and weaknesses. Although example-based systems are capable of generating highly accurate translations when the input sentence matches a previously observed example, SMT systems typically provide better generalisation and robustness. The goal of hypothesis combination is to combine the outputs from multiple translations in a way that is able to exploit differences in the nature of errors made by the individual systems.

This chapter extends the lattice minimum Bayes-risk decoder described in Chapter 7 to the task of combining multiple statistical machine translation lattices. This allows the decoder to operate on a much richer and more diverse space of translations, resulting in significant improvements in translation quality for several language pairs.

A comparative overview of recent approaches to machine translation combination is presented in Section 8.1. An efficient lattice MBR system combination decoder based on weighted finite-state transducers is introduced in Section 8.2. Arabic→English, Chinese→English and Finnish→English multi-input translation experiments exploiting alternative analyses of the source language input sentence are described in Section 8.3. In Section 8.4, the lattice MBR combination decoder is used for multi-source translation by combining French→English and Spanish→English translation lattices.

8.1 Background and Related Work

This section presents an overview and comparison of the main approaches to improving machine translation quality through the combination of multiple system outputs.

8.1.1 Consensus Network Decoding for Machine Translation

Probably the most widely used method for system combination is *consensus network decoding*. Consensus network decoding has been successfully applied to combine multiple system outputs in automatic speech recognition using recogniser output voting error reduction (ROVER) based on simple confidence measures (Fiscus, 1997). More recently, consensus decoding techniques have been demonstrated to improve the quality of machine translation (Matusov et al., 2006; Rosti et al., 2007a,b; Sim et al., 2007). The importance of ensuring sufficient diversity amongst individual system outputs is shown to have a significant impact on consensus decoding performance in an empirical study by Macherey and Och (2007). Consensus networks are constructed from k-best lists by aligning each hypothesis against a single alignment reference. For machine translation, an appropriate choice of alignment reference is the minimum Bayes-risk hypothesis (Kumar and Byrne, 2004). Alignments are computed with respect to alignment metrics such as Word Edit Rate (WER) in Bangalore et al. (2001) or Translation Edit Rate (TER) in Sim et al. (2007). The use of TER is motivated by the greater flexibility in word reordering allowed by shifts and the relative simplicity of the alignment model.

The consensus network created by aligning each hypothesis to the alignment reference consists of a sequence of word alternatives with scores. Figure 8.1 shows an example word confusion network from Sim et al. (2007). The scores on each arc indicate the number of hypotheses for which the labelled word was aligned to the chosen reference. The consensus output is easily found during decoding by selecting the word sequence with the maximum score. Computing alignments for each pair of hypotheses is computationally expensive so this form of system combination is usually limited to relatively short k-best lists.

Matusov et al. (2006) use IBM Model 1 and HMM alignments to explicitly model word reordering in pairs of hypotheses produced by different machine translation systems. These alignments allow each translation alternative to be reordered with respect to a chosen reference. The confusion network is formed as the union of the monotone one-to-one word alignments and the consensus translation is then extracted using voting based on global system probabilities. The reordering and alignment of words in the confusion network allows consensus translations that differ from all of the original system translations. One limitation of this approach is that it is expensive to compute the alignment so only the single best translation hypothesis from each system is considered during combination.

Sentence-level, phrase-level and word-level system combination approaches based on confidence measures have been applied to the task of combining k-best lists generated by six different machine translation systems (Rosti et al., 2007a). At the sentence level, a linear

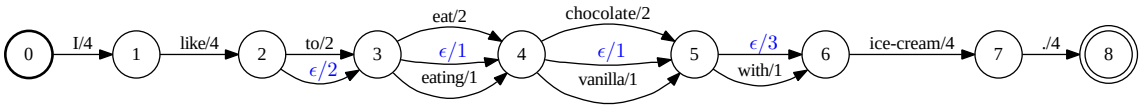


Figure 8.1: Word confusion network formed from four alternative translation hypotheses.

interpolation of the sum, max and average system-specific confidence scores, sentence posterior probabilities and a 5-gram language model are combined in a generalised linear model to re-rank merged k-best lists using feature weights optimised on the tuning set. For phrase-level combination, sentence-specific phrase tables are estimated using posterior probabilities and target-to-source phrase alignments. The best hypothesis is obtained by re-decoding with the new phrase table and a language model. System combination was shown to be most effective at the word level; decoding selects the hypothesis with minimum Bayes risk under WER, based on a TER alignment to the sentence that best agrees with the other hypotheses following the work of Sim et al. (2007).

8.1.2 Multi-Source Machine Translation

System combination can be applied to the task of *multi-source translation* whenever multiple translations of the source language input sentence are available. Multinational corporations and multi-lingual international organisations such as the European Parliament and the United Nations often need to provide translations in multiple languages. It is interesting to consider how such translations can be exploited to improve overall quality. The main motivation for multi-source translation is that some of the ambiguity that must be resolved in translating between one pair of languages may not be present in a different pair.

For example, a French document might first be translated into Spanish. If the document must also be translated into English, then it is useful to be able to exploit both existing translations by combining French→English and Spanish→English translation knowledge. Combining translations from multiple sources is useful since different language pairs might better handle the translation of particular syntactic or semantic ambiguities, and the inclusion of additional source inputs with similar word order to the target language may reduce the prevalence of errors due to limitations in word reordering associated with particular language pairs.

Och and Ney (2001) show small improvements in WER through multi-source translation of up to six European languages by replacing the single-system translation probability in the standard Bayes decision rule with either the product or max of the individual system translation probabilities. The advantage of this simple approach is that no changes are required to the decoder or search algorithms. However, only the 1-best output from each bilingual system is considered and there is no hypothesis combination, only likelihood-based hypothesis re-ranking. More recently, Schwartz (2008) has shown the product and max methods to perform less well with modern multi-parallel corpora such as Europarl (Koehn, 2005) when evaluated using translation metrics such as BLEU and TER.

Consensus decoding based on word alignments is applied to multi-source translation in order to select hypotheses from a combination of Chinese→English and Japanese→English translation outputs in Matusov et al. (2006). The errors common in one language pair may be correctly translated in the other language pair and consensus decoding is an effective way of exploiting the individual strengths of each system. Again, only the 1-best translation from each source language is considered during combination. Multi-source translation using two Chinese→English and Japanese→English systems results in good gains in BLEU score with respect to the best of the individual translations.

Schroeder et al. (2009) compares three methods for multi-source translation: (i) selection of a single hypothesis from one of a set of regular bilingual translation outputs using the max method of Och and Ney (2001) (this is the baseline); (ii) hypothesis combination using consensus network decoding with a lattice of confusion networks following the work of Rosti

et al. (2007b); and (iii) a novel approach to multi-source translation that combines multiple source language input sentences as a single multi-lingual lattice prior to single-pass translation decoding. System combination of individual k-best lists is found to be simpler and more effective than the multi-lingual input lattice approach.

8.1.3 Multi-Input Machine Translation

If multiple representations of the input sentence in the same source language are available, *multi-input translation* can be used to improve translation quality. One example of this is the exploitation of multiple morphological decompositions of the source language sentence.

When translating from languages with a rich morphology it is common practice to apply a morphological analyser to the training corpus and evaluation data prior to training the alignments and models used during translation. In phrase-based and hierarchical phrase-based statistical machine translation systems, different morphological analyses result in different phrasal constituents and probabilities (phrase-pairs in phrase-based SMT; hierarchical rules in hierarchical phrase-based SMT). Multiple translations can be generated by running the same system trained on each of the morphological analyses, a technique known as *hybrid translation*. MBR decoding over k-best lists has been shown to improve Arabic→English and Finnish→English translation quality by combining translations generated from multiple morphological decompositions of the foreign language sentence (de Gispert et al., 2009). The experiments presented later in this chapter extend these techniques to lattice-based MBR decoding. Figure 8.2 shows the Arabic→English multi-input hybrid translation pipeline. Lattice MBR decoding is an appropriate framework for combination since it allows for efficient combination of large lattices generated from multiple alternative morphological analyses of the input sentence.

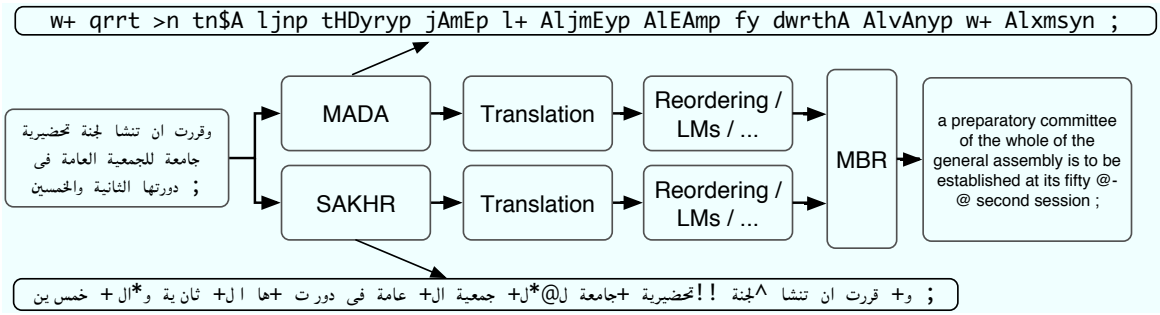


Figure 8.2: Multiple lattice hybrid translation pipeline using MBR decoding to combine lattices generated by two different morphological analysers: MADA and SAKHR.

Multi-input translation can be applied to other tasks where there is uncertainty associated with the input. Chinese text is normally preprocessed by segmenting the unbroken character stream as a series of words (Chang et al., 2008). Since there is uncertainty about the most appropriate segmentation for MT, multiple segmentations can be translated separately and then combined. Multi-input methods may also be appropriate for translating alternative ASR transcriptions in speech translation (Mathias and Byrne, 2006), or for combining translations generated from context-independent and context-dependent alignment models (Brunning et al., 2009).

8.1.4 Lattice-Based Combination Techniques

Consensus decoding enforces a monotone one-to-one alignment and order on the words of the translated sentence with null tokens used to ensure all hypotheses are the same length. While this topology is well suited to the time series nature of automatic speech recognition, it is less appropriate for machine translation which allows arbitrary alignments and a large degree of phrase reordering. Furthermore, word-level confusion networks permit hypotheses that break phrase-internal fluency and consistency, one of the main advantages of phrase-based and hierarchical phrase-based systems. These limitations motivate more recent lattice-based approaches to the combination of multiple translations.

8.2 Minimum Bayes-Risk Decoding for Lattice Combination

The much larger space of hypotheses encoded in lattices motivates the extension of lattice MBR decoding to lattice MBR system combination. First-pass decoding results in a set of M distinct translation lattices $\mathcal{E}^{(i)}$, $i = 1 \ldots M$ for each foreign input sentence. These lattices might be generated by M separate translation systems, or, alternatively, the same translation system under different training conditions or configurations. The evidence space for MBR decoding is formed as the union of the individual lattices using the WFST union operator:

$$\mathcal{E} = \bigoplus_{i=1}^{M} \mathcal{E}^{(i)} \tag{8.1}$$

Let $\mathcal{N}^{(i)} = \{u_1^{(i)}, u_2^{(i)}, \ldots, u_{|N|}^{(i)}\}$ denote the set of all n-grams in lattice $\mathcal{E}^{(i)}$. Then the set of all n-grams in the union of lattices $\mathcal{E}$ is defined as follows:

$$\mathcal{N} = \bigcup_{i=1}^{M} \mathcal{N}^{(i)} \tag{8.2}$$

With the definitions (8.1) and (8.2), the multiple lattice MBR system combination decoder has the same form as the single lattice decoder of Equation (7.7). The only difference is in the computation of the n-gram path posterior probabilities $p(u|\mathcal{E})$. The posterior probability of n-gram u in the union of lattices is computed as a linear interpolation of the posterior probabilities according to the evidence of each individual lattice so that

$$p(u|\mathcal{E}) = \sum_{i=1}^{M} \lambda_i \, p_i(u|\mathcal{E}^{(i)}), \tag{8.3}$$

where the parameters $0 \leq \lambda_i \leq 1$ such that $\sum_{i=1}^{M} \lambda_i = 1$ specify the interpolation weight associated with each system in the combination and are optimised with respect to a tuning set. The system specific posteriors required for the interpolation are computed as

$$p_i(u|\mathcal{E}^{(i)}) = \sum_{E \in \mathcal{E}_u^{(i)}} P_i(E|F), \tag{8.4}$$

LMBR-SYSTEM-COMBINATION($\mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(M)}, \alpha_1^M, \lambda_1^M, \theta_{0\ldots4}$)

for $i \leftarrow 1 \ldots M$
  do $\mathcal{E}^{(i)} \leftarrow$ NORMALIZE($\alpha_i \times \mathcal{E}^{(i)}$)
    $\mathcal{N}^{(i)} \leftarrow$ EXTRACT-NGRAMS($\mathcal{E}^{(i)}$)
    for each $u \in \mathcal{N}^{(i)}$
      do $\Psi_u \leftarrow$ MAKE-COUNT-FSA(u)
        $\mathcal{E}_u^{(i)} \leftarrow \mathcal{E}^{(i)} \circ \Psi_u$
        $p_i(u|\mathcal{E}^{(i)}) \leftarrow \sum_{E \in \mathcal{E}_u^{(i)}} P_i(E|F)$
$\mathcal{N} \leftarrow \bigcup_{i=1}^{M} \mathcal{N}^{(i)}$
for each $u \in \mathcal{N}$
  do $p(u|\mathcal{E}) \leftarrow \sum_{i=1}^{M} \lambda_i p_i(u|\mathcal{E}^{(i)})$
$\mathcal{E}_h \leftarrow 0 \times \bigoplus_{i=1}^{M} \mathcal{E}^{(i)}$
$\mathcal{E}_h \leftarrow$ FST-OPTIMIZE($\mathcal{E}_h$)
$\mathcal{E}_h \leftarrow$ APPLY-WORD-FACTOR($\mathcal{E}_h, \theta_0$)
for each $u \in \mathcal{N}$
  do $\Omega_u \leftarrow$ MAKE-GAIN-FSA($u, \theta_{|u|} \times p(u|\mathcal{E})$)
    $\mathcal{E}_h \leftarrow \mathcal{E}_h \circ \Omega_u$
return FIND-BEST-PATH($\mathcal{E}_h$)

Figure 8.3: Lattice minimum Bayes-risk system combination algorithm.

where $P_i(E|F)$ is the posterior probability of translation E given source sentence F and the sum is taken over the subset $\mathcal{E}_u^{(i)} = \{E \in \mathcal{E}^{(i)} \colon \#_u(E) > 0\}$ of the lattice containing paths with at least one occurrence of the n-gram u. These posterior probabilities can be computed efficiently by pushing weights in the log semiring, as described in Section 7.1.4. The smoothing factor α in Equation (7.4) applied to the posterior translation probabilities can be optimised independently for each set of lattices, or jointly for system combination.

8.2.1 Lattice Combination Implementation with WFSTs

The lattice minimum Bayes-risk system combination decoder over M individual evidence spaces $\mathcal{E}^{(i)}$, $i = 1 \ldots M$ is obtained by substituting the definitions (8.1), (8.2) and (8.3) into the single-system LMBR decoder of Equation (7.7). The system combination hypothesis $\hat{E}_+$ that maximises the conditional expected gain is

$$\hat{E}_+ = \operatorname*{argmax}_{E' \in \bigoplus_{i=1}^{M} \mathcal{E}^{(i)}} \left\{ \theta_0 |E'| + \sum_{u \in \bigcup_{i=1}^{M} \mathcal{N}^{(i)}} \left(\theta_u \#_u(E') \sum_{i=1}^{M} \lambda_i \, p_i(u|\mathcal{E}^{(i)}) \right) \right\}, \tag{8.5}$$

where $p_i(u|\mathcal{E}^{(i)})$ is the n-gram path posterior probability of u defined by Equation (8.4).

The lattice minimum Bayes-risk system combination decoder of Equation (8.5) is implemented by the algorithm shown in Figure 8.3. The input parameters are the individual evidence spaces $\mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(M)}$, exponential smoothing factors α_1^M, interpolation weights λ_1^M, fixed per-word factor θ_0 and order-specific n-gram factors $\theta_1 \ldots \theta_4$.

The evidence space is first normalised to form the smoothed posterior distribution (line 2). Next, the n-grams in each lattice $\mathcal{E}^{(i)}$ are extracted (line 3). Then, the n-gram path posterior probabilities $p_i(u|\mathcal{E}^{(i)})$ are computed from each evidence space $\mathcal{E}^{(i)}$ (lines 4–7). These probabilities can be efficiently computed using path counting transducers (Blackwood and Byrne, 2010) (Section 7.2). The set of n-grams in the union of the lattices is formed (line 8), and the interpolated posterior distribution $p(u|\mathcal{E})$ of Equation (8.3) is computed in lines 9 and 10. The system combination hypothesis space is formed as the unweighted union of the M individual hypothesis spaces (line 11), and optimised by ϵ-removal, determinization, and minimisation (line 12). MBR decoding in the union of the evidence spaces under the interpolated distribution $p(u|\mathcal{E})$ proceeds in exactly the same way as for single-system LMBR decoding (lines 13–17). Fast decoding (Section 7.2.4) can be used to improve time efficiency.

Instead of computing the interpolated posterior distribution from the individual distributions as in Equation (8.3), the n-gram path posterior probabilities could be found directly from the weighted union of lattice evidence spaces:

$$p(u|\mathcal{E}) = p(u| \cup_{i=1}^{M} \{\lambda_i \times \mathcal{E}^{(i)}\}) \tag{8.6}$$

However, the WFST optimisation operations (especially determinize) take a very long time when applied to the weighted union of lattices containing many similar paths with slightly different costs. If the lattice is not optimised, then computing the n-gram path posterior probabilities is very slow. It is therefore much faster to compute $p_i(u|\mathcal{E}^{(i)})$ for each individual lattice and then interpolate the posterior distribution offline. The optimisation of the hypothesis space (line 12) is fast since the weights of the unioned lattices are removed before the determinization.

8.3 Multi-Input Translation Experiments

This section describes the use of lattice minimum Bayes-risk decoding to improve the quality of Arabic→English and Chinese→English translation by combining lattices generated from multiple inputs (Section 8.1.3). Finnish→English system combination experiments are also briefly summarised. For Arabic and Finnish translation, the multiple inputs consist of different morphological analyses of the input sentence (de Gispert et al., 2009). For Chinese translation, the multiple inputs represent different Chinese word segmentations.

8.3.1 System Development and Lattice Generation

This section describes the experimental framework and lattice generation procedures used in Arabic→English, Chinese→English, and Finnish→English LMBR system combination.

Arabic→English For Arabic→English translation, multi-input system combination is evaluated within the framework of the NIST MT08 constrained track.[1] The baseline translation system and testsets are the same as for single-system LMBR (Section 7.3).

Prior to generating the alignments, the Arabic side of the parallel text is pre-processed according to one of three different morphological analyses. The MADA1 and MADA2 analyses are generated using the MADA toolkit (Habash and Rambow, 2005). The SAKHR analysis

[1] http://www.itl.nist.gov/iad/mig/tests/mt/2008/

Arabic	wzrA′ Alby}p AlErb yTAlbwn bAglAq mfAEl dymwnp AlAsrA}ylY
MADA1	wzrA′ Alby}p AlErb yTAlbwn b+ <glAq mfAEl dymwnp Al<srA}yly
MADA2	wzrA′ Al+ by}p Al+ Erb yTAlbwn b+ <glAq mfAEl dymwnp Al+ <srA}yly
SAKHR	wzrA′ Al+ by}p Al+ Erb yTAlbwn b+ AglAq mfAEl dymwnp Al+ AsrA}ylY
English	arab environment ministers call for israeli nuclear reactor at dimona to be shut down

Figure 8.4: Buckwalter transliterated Arabic source language sentence, three different morphological analyses, and one of the English references from NIST MT08 mt0205tune.

is generated using the Arabic Morphological Tagger of Sakhr Software.[1] Figure 8.4 shows the Buckwalter transliterated Arabic language input sentence, the sentences that result from three different morphological analyses, and one of the English references for the first sentence of the Arabic→English mt0205tune testset.

Separate translation systems are trained from each of these morphological analyses and used to generate three different translation lattices for each sentence to be translated. Prior to system combination, these lattices are rescored with 5-gram language models as described in Chapter 5. It is these 5-gram rescored translation lattices that form the evidence space for multi-input LMBR system combination.

Two-way and three-way k-best list and lattice-based minimum Bayes-risk decoding is used to combine the individual system hypotheses. For k-best combination, k-best lists from each system (k=500 for two-way combination; k=333 for three-way combination) are merged to create an aggregate list, with posterior distributions over the individual lists interpolated to form a new distribution over the merged list (de Gispert et al., 2009). MBR decoding under the sentence-level BLEU score (Kumar and Byrne, 2004) is used to select the minimum risk hypothesis.

For lattice-based combination, the hypothesis space is formed from the union of the full lattices generated by decoding with each morphological analysis. The n-gram posterior probabilities required by the lattice MBR decoder of Equation (8.5) are computed as a linear interpolation of posteriors according to each individual system. The interpolation weights λ_i in Equation (8.3) are optimised with respect to the tuning set mt0205tune.

Chinese→English Since written Chinese does not normally explicitly mark the spaces between words, the Chinese input sentence must be segmented into a sequence of tokens before alignments can be generated. Various word segmentation algorithms exist, with different strengths and weaknesses (Chang et al., 2008). It is possible to improve translation quality and robustness by training a hybrid SMT system from multiple segmentations and combining the outputs using the lattice MBR decoder described in Section 8.2.

Chinese→English multi-input translation experiments are presented for the GALE P4 evaluation. The newswire testsets tune.text.nw and test.text.nw contain 3085 and 2055 sentences. The web testsets tune.text.web and test.text.web contain 4221 and 3092 sentences. The baseline system is the same hierarchical decoder as was used for single-system LMBR (Section 7.3). However, two hierarchical rulesets are extracted and separate optimised translation decoders trained for each ruleset. The first ruleset is extracted using tokenized Chinese data distributed by BBN Technologies for the GALE P4 evaluation. The second ruleset is

[1] http://www.sakhr.com/default.aspx

extracted using a segmentation of the Chinese side of the parallel data produced using the Oxford Chinese word segmentor (Zhang and Clark, 2007).

Separate lattices are generated by translating with each optimised decoder; these are rescored with 5-gram language models as described in Chapter 5. The union of the lattices is then used as the evidence space for lattice MBR system combination. The per-word factor θ_0 and interpolation weights λ_i are optimised with respect to the tune.text.nw set for newswire translation, and the tune.text.web set for web data translation.

Finnish→English Finnish has a rich morphology that can cause data sparsity problems for alignments based on words. For this reason, the Finnish side of the parallel text is usually preprocessed with a morphological analyser before generating the alignments. One of the aims of Morpho Challenge 2009 (Kurimo et al., 2009) was to use unsupervised morpheme analysis to improve the quality of statistical machine translation from morphologically complex languages such as Finnish and German into English.

The effect of morpheme analysis on translation quality was evaluated by translating the proceedings of the European Parliament (Koehn, 2005) using the Moses decoder (Koehn et al., 2007).[1] For both the word-based and morphologically analysed models, k-best lists of depth k=200 were generated, converted to lattices, and then evaluated using the MBR system combination framework described in Section 8.2.

The development set eu-dev contains 2849 sentences and the test set eu-test contains 3000 sentences. Approximately 1.2M lines of Finnish→English parallel data were used to train the translation decoder. The k-best lists obtained by translating with the word-based and morpheme-based models were generated to contain only unique hypotheses. The interpolation weights λ_i are optimised with respect to the development set eu-dev.

8.3.2 Minimum Bayes-Risk Combination Results and Analysis

Arabic→English Table 8.1 shows IBM BLEU scores and TER for k-best list and lattice-based minimum Bayes-risk system combination of Arabic→English translation lattices. The rows A, B, and C show single-system MERT optimised translation scores after rescoring with the large 5-gram language models for the MADA1, MADA2, and SAKHR systems respectively. Interestingly, all three systems have quite similar translation quality as measured by the BLEU score, although the SAKHR system handles the newsgroup set (mt08ng) less successfully.

The following rows show the results of two-way and three-way system combination using k-best lists (merged k=1000) (row MBR) and lattices (row LMBR). In agreement with de Gispert et al. (2009) large gains are observed for two-way k-best system combination of lattices generated from alternative morphological decompositions. The relative gains from lattice-based MBR are about +0.6 BLEU higher than the gains from k-best MBR. The optimised interpolation weights for two-way lattice combination were $\lambda_1 = \lambda_2 = 0.5$ for all three pairs A+B, B+C, and A+C. This is not surprising given that the ML 1-best BLEU scores of all three systems are so similar.

These results show that lattice minimum Bayes-risk decoding is able to exploit the much larger space of hypotheses encoded in multiple lattices, and that these additional hypotheses

[1]The Finnish→English first-pass translation and k-best list generation was performed by Sami Virpioja and Mikko Kurimo of the Adaptive Informatics Research Centre, Helsinki University of Technology.

Configuration		mt0205tune		mt0205test		mt08nw		mt08ng	
		BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
A	HiFST+5g	54.2	40.5	53.8	41.0	51.4	43.5	36.3	54.9
B	HiFST+5g	53.8	41.2	53.6	41.4	51.4	43.8	36.2	54.1
C	HiFST+5g	54.1	40.7	53.8	40.7	51.5	43.6	35.5	55.3
A+B	+MBR	55.1	40.0	54.7	40.3	52.7	42.8	37.1	54.1
	+LMBR	55.7	39.8	55.5	39.8	53.4	42.3	37.7	53.1
B+C	+MBR	54.7	40.2	54.5	40.3	52.5	43.0	37.4	54.3
	+LMBR	55.3	40.0	55.2	40.0	53.1	42.6	38.1	53.1
A+C	+MBR	55.4	39.7	54.9	39.9	53.0	42.5	37.7	54.3
	+LMBR	55.9	39.3	56.0	39.4	53.7	42.3	38.1	53.0
A+B+C	+MBR	55.3	39.7	54.9	40.0	53.0	42.6	37.7	54.4
	+LMBR	56.0	39.5	55.8	39.7	53.9	42.3	38.5	52.9

Table 8.1: BLEU scores and TER for uncased k-best list and lattice MBR system combination of Arabic→English translations generated from alternative morphological analyses.

	mt0205tune		
	A	B	C
A	0.0	+1.5	**+1.7**
B		0.0	+1.1
C			0.0

	mt0205test		
	A	B	C
A	0.0	+1.7	**+2.2**
B		0.0	+1.4
C			0.0

	mt08nw		
	A	B	C
A	0.0	+1.9	**+2.2**
B		0.0	+1.6
C			0.0

	mt08ng		
	A	B	C
A	0.0	+1.4	**+1.8**
B		0.0	+1.8
C			0.0

Table 8.2: Absolute improvements in BLEU for LMBR two-way combination of Arabic→English translations generated from alternative morphological decompositions.

are useful for improving the quality of translation. Some of the additional gain from lattice MBR might be obtained by using deeper k-best lists, but the $\mathcal{O}(n^2)$ computational complexity usually limits k-best MBR decoding to fairly short lists.

Three-way system combination (A+B+C) shows only relatively small gains over the best two-way combination (A+C) on the tuning sets, although lattice MBR is again seen to perform significantly better than k-best MBR. The investigation of lattice and k-best list evidence space sizes in Section 7.3.2.2 of Chapter 7 explains why k-best list 3-way combination is so much worse than lattice-based combination: the $k = 333$ lists are much too short. Comparing two-way and three-way combination shows modest gains of +0.2 BLEU for newswire and +0.4 BLEU for newsgroup data on the MT08 testsets. It may be that two of the three systems are too similar for there to be any real benefit from including both of them in the combination. These results were obtained under the same evaluation conditions as the NIST MT08 evaluation and are highly competitive with the other submissions.[1]

[1] http://www.itl.nist.gov/iad/mig/tests/mt/2008/doc/mt08_official_results_v0.html

Configuration		Newswire Data				Web Data			
		tune.text.nw		test.text.nw		tune.text.web		test.text.web	
A	HiFST+5g	28.4	62.10	21.0	64.05	16.1	70.37	14.4	69.10
B	HiFST+5g	27.6	63.26	20.2	66.29	15.3	73.09	13.7	71.87
A+B	+LMBR	29.2	61.77	21.9	64.26	16.4	72.00	15.0	70.79

Table 8.3: BLEU scores and TER for Chinese→English lattice MBR system combination (A+B) generated from translation systems trained on different Chinese word segmentations.

Configuration		eu-dev	eu-test
A	Moses	29.37	27.64
B	Moses	29.48	27.42
A+B	+LMBR	29.75	28.61

Table 8.4: BLEU scores for Finnish→English MBR system combination of lattices (A+B) generated from word-based model (A) and morpheme-based model (B).

The absolute gains in BLEU from lattice-based two-way MBR system combination over the best of the individual systems are summarised in Table 8.2. Absolute gains of between +1.7 and +2.2 BLEU are very large indeed on top of an already highly scoring baseline, re-emphasising that lattice-based minimum Bayes-risk decoding for multi-input translation is a very effective way of improving overall translation quality. The combination A+C (i.e. MADA1 and SAKHR) gives the largest gains on all four evaluation sets.

Chinese→English The Chinese→English multi-input translation results are shown in Table 8.3. Again, the ML 1-best translations of the individual systems have very similar BLEU scores. Compared to the best of the individual system scores, the newswire data test.text.nw set gains +0.9 BLEU and the web data test.text.web set gains +0.6 BLEU. These results show that alternative Chinese word segmentations can be exploited to improve translation quality using lattice MBR decoding over multiple lattices.

Finnish→English Table 8.4 shows the results of MBR system combination of multiple Finnish→English k-best lists. The translations generated from the word-based models (row A) and morpheme-based models (row B) have very similar BLEU scores. However, it is possible to improve translation quality by MBR system combination (row A+B), especially on the eu-test set where there is an improvement of +1.2 BLEU over the best of the individual systems. More detailed experiments that use the multiple-lattice MBR decoder described in this chapter to combine translations generated from a variety of unsupervised morphological analysers are presented in Kurimo et al. (2009).

8.3.2.1 Length Tuning

Lattice MBR system combination under linear BLEU with the per-word factor θ_0 of Equation (8.5) set in the way suggested by Tromble et al. (2008) often produces output that is shorter than required. Figure 8.5 shows the effect of tuning θ_0 on the BLEU score (upper plot) and brevity penalty (lower plot). If the per-word factor is not tuned, i.e. when $\theta_0 = 0$, then the brevity penalty is approximately 0.97 and the BLEU score is penalised accordingly.

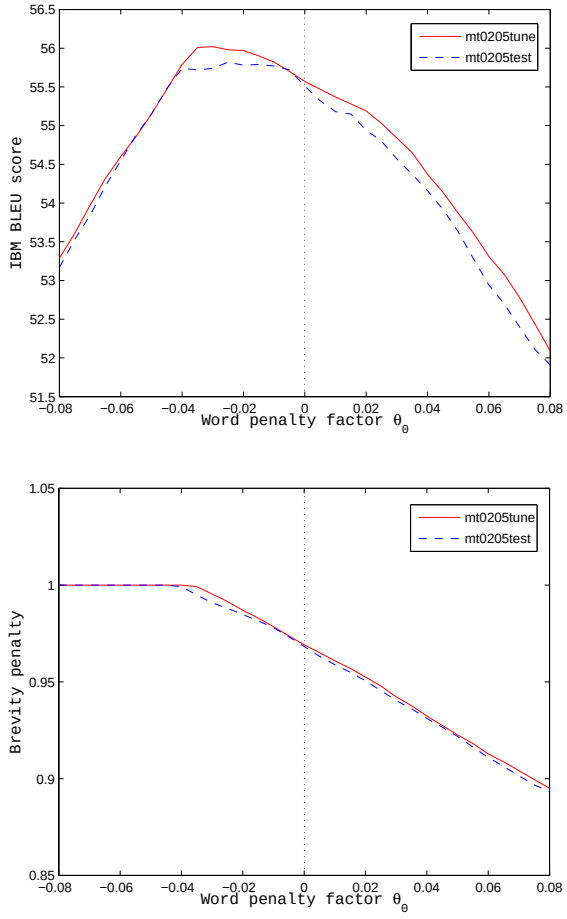


Figure 8.5: Effect of per-word factor θ_0 on IBM BLEU score (upper plot) and brevity penalty (lower plot) in three-way lattice MBR system combination of Arabic→English translations.

For the Arabic→English three-way system combination experiments reported above, the optimal word factor of $\theta_0 = -0.03$ favours longer hypotheses and results in gains of +0.5 BLEU for mt0205tune and +0.2 BLEU for mt0205test, with good testset generalisation. Comparing the two plots shows that the BLEU score is maximised with a word factor θ_0 that produces output that is as short as possible without incurring a brevity penalty.

The experiments presented throughout this chapter report scores using the IBM implementation of BLEU in which the brevity penalty is computed with respect to the closest reference length. For NIST BLEU, where brevity is computed with respect to the shortest reference, the penalty can be more severe and it is even more important to make sure the translation output is the right length.

8.3.2.2 Evidence Space Size

In order to better understand the large gains in BLEU observed for lattice minimum Bayes-risk combination, Table 8.5 contrasts BLEU scores and TER for k-best lists of various depths and for full lattice MBR. One reason for the difference in performance is that MBR and LMBR decoding use different approximations to the BLEU score: MBR uses the sentence-level BLEU

	mt0205tune			mt0205test		
	BLEU	TER	BP	BLEU	TER	BP
System A	54.23	40.49	0.995	53.78	41.02	0.994
System B	53.79	41.22	0.998	53.59	41.36	0.995
10	54.66	40.51	0.998	54.38	40.91	0.996
50	55.06	40.25	0.996	54.72	40.50	0.992
100	55.20	40.34	0.999	54.79	40.65	0.996
200	55.25	40.26	0.998	54.90	40.50	0.995
500	55.37	40.14	0.997	55.12	40.31	0.994
1000	55.41	40.10	0.996	55.18	40.28	0.995
2000	55.35	40.13	0.996	55.23	40.26	0.994
5000	55.43	40.10	0.996	55.24	40.18	0.993
10000	55.41	40.07	0.995	55.23	40.17	0.993
20000	55.52	40.02	0.995	55.21	40.18	0.992
LMBR	55.69	39.75	0.991	55.49	39.84	0.989

Table 8.5: BLEU score, TER and brevity penalty (BP) for first-pass translation and system combination of k-best lists of various sizes contrasted with full lattice-based MBR decoding.

(Kumar and Byrne, 2004), while LMBR uses a linear approximation to the sentence-level BLEU based on n-gram posterior probabilities (Tromble et al., 2008). Since computing the risk for large numbers of hypotheses can be slow, k-best MBR is typically limited to relatively short lists. In this experiment, k-best lists are first converted to lattices so that much deeper lists can be used. The conversion also allows k-best list and full lattice MBR performance to be directly compared using the same approximation to the BLEU score. The results show that increasing the k-best list depth up to 500 hypotheses gives gradual incremental gains in BLEU, but that increasing the depth further beyond 500 gives no real additional gains, even at a depth of 20000. Combination with the full lattice, however, improves upon the 20000-best lists: +0.2 BLEU for mt0205tune and +0.3 BLEU for mt0205test. Using the full evidence space of lattice hypotheses is clearly beneficial in MBR decoding.

8.3.2.3 Translation Examples

Figure 8.6 shows an example of improved Arabic→English translation obtained through combination of translation lattices generated from alternative morphological decompositions. The sentence-level BLEU score $\text{BLEU}_S(E)$ for a sentence E is just the geometric mean of n-gram precisions, ignoring the brevity penalty so that

$$\text{BLEU}_S(E) = \exp\Big(\frac{1}{N}\sum_{i=1}^{N}\log p_i\Big), \tag{8.7}$$

where the order N is 4 and p_i denotes the n-gram precision at order i computed with respect to the union of all n-grams in the set of references for sentence E.

Although the maximum likelihood 1-best translation $\hat{E}_A$ produced by system A is quite poor, several higher-order n-grams do match the references correctly. The maximum likelihood 1-best translation $\hat{E}_B$ is much more fluent and closer to the references, although information content (i.e. the number of new plants) has been omitted. The lattice minimum

Source	Tokenized translation string	BLEU_S
R_1	over the next 13 years , peking invested in the construction of 7 new plants .	-
R_2	peking invested in the construction of 7 new plants over the next 13 years .	-
R_3	beijing has invested in building 7 new plants over the following 13 years .	-
R_4	peking has invested in the construction of 7 new plants in the next 13 years .	-
$\hat{E}_A$	the beijing invested in the construction of 7 new factor in the next 13 years .	0.6865
$\hat{E}_B$	beijing has invested in building new plants in the next 13 years .	0.7882
$\hat{E}_+$	beijing has invested in the construction of 7 new plants in the next 13 years .	1.0000

Figure 8.6: Four reference translations, single-system ML 1-best translation hypotheses $\hat{E}_A$ and $\hat{E}_B$, and improved LMBR system combination hypothesis $\hat{E}_+$ (from mt0205tune).

Source	Tokenized translation string	Length	BLEU_S
R_1	but the world it was born into is no longer there .	12	-
R_2	but the world in which it was born exists no more .	12	-
R_3	but the world in which it was born no longer exists .	12	-
R_4	but the world in which it was born is no longer exists .	13	-
$\hat{E}_A$	but the world no longer existed was being born .	10	0.1809
$\hat{E}_B$	but the world in which it was born no longer exists .	12	1.0000
$\hat{E}_+$	but the world in which no longer existed .	9	0.3803

Figure 8.7: Four reference translations, single-system ML 1-best translation hypotheses $\hat{E}_A$ and $\hat{E}_B$, and degraded LMBR system combination hypothesis $\hat{E}_+$ (from mt0205test).

Bayes-risk decoding hypothesis $\hat{E}_+$ is much better than both of the individual system outputs: it is completely fluent and captures all of the information in the reference translations with perfect precision over all n-gram orders.

Figure 8.7 shows an example of degraded translation quality. The maximum likelihood hypothesis $\hat{E}_B$ is a flawless translation with perfect n-gram precisions at all orders. The LMBR system combination hypothesis $\hat{E}_+$ has a much lower sentence-level BLEU score. It is missing the important content word "born" (even though it is present in the ML 1-best of both of the individual system outputs), and has poor fluency. The per-word factor θ_0, optimised at the corpus-level, is set inappropriately for this short sentence.

For the degraded sentence shown in Figure 8.7, the expected gains for the ML 1-best hypotheses $\hat{E}_A$ and $\hat{E}_B$, and for the LMBR system combination hypothesis $\hat{E}_+$ are shown in Table 8.8. These gains are computed using the linear interpolation of n-gram path posterior probabilities defined in Equation (8.3). The table shows the partial gain at each order and the total gain before and after application of the per-word factor θ_0. The expected gain $\sum_{u \in \mathcal{N}} g_u(E, E')$ of the hypothesis $\hat{E}_B$ is higher than that of the poor quality system combination hypothesis $\hat{E}_+$. However, after applying the per-word factor θ_0, the hypothesis $\hat{E}_+$ has the highest gain. For this sentence, a smaller θ_0 may be appropriate. Ideally, the value of θ_0 should be a function of the length of the sentence and optimised on the development set.

8.3.2.4 Hypothesis Selection

Lattice MBR system combination selects the hypothesis in the union of lattices that maximises the expected gain. Ignoring the corpus-level brevity penalty, the best possible system

E'	1g	2g	3g	4g	$\sum_{u \in \mathcal{N}} g_u(E, E')$	$\theta_0 \lvert E' \rvert + \sum_{u \in \mathcal{N}} g_u(E, E')$
$\hat{E}_A$	0.2831	0.2371	0.1634	0.1184	0.8020	-0.3980
$\hat{E}_B$	0.3102	0.2830	0.2326	0.2253	1.0510	-0.3490
$\hat{E}_+$	0.2799	0.2627	0.2133	0.1849	0.9408	-0.1592

Figure 8.8: Expected gains before and after applying the per-word factor θ_0 for single-system ML 1-best hypotheses $\hat{E}_A$ and $\hat{E}_B$, and LMBR system combination hypothesis $\hat{E}_+$.

		$\hat{E}_+ < \hat{E}_A$	$\hat{E}_+ = \hat{E}_A$	$\hat{E}_+ > \hat{E}_A$
mt0205tune	$\hat{E}_+ < \hat{E}_B$	261	91	299
	$\hat{E}_+ = \hat{E}_B$	108	138	125
	$\hat{E}_+ > \hat{E}_B$	342	141	570
mt0205test	$\hat{E}_+ < \hat{E}_B$	268	88	279
	$\hat{E}_+ = \hat{E}_B$	89	142	146
	$\hat{E}_+ > \hat{E}_B$	301	135	592
mt08nw	$\hat{E}_+ < \hat{E}_B$	90	22	150
	$\hat{E}_+ = \hat{E}_B$	35	49	51
	$\hat{E}_+ > \hat{E}_B$	120	55	241
mt08ng	$\hat{E}_+ < \hat{E}_B$	80	34	83
	$\hat{E}_+ = \hat{E}_B$	16	41	27
	$\hat{E}_+ > \hat{E}_B$	79	27	160

Table 8.6: Number of Arabic→English sentences where LMBR hypothesis $\hat{E}_+$ has sentence-level BLEU score equal to, worse than, or better than the individual systems $\hat{E}_A$ and $\hat{E}_B$.

combination outcome is for LMBR decoding to choose a hypothesis that has better sentence-level BLEU score than each of the individual systems in the combination. The worst possible outcome is to choose a hypothesis that has lower sentence-level BLEU score than each of the individual systems, since then it would have been better to pick any of the individual systems rather than the combination hypothesis.

Table 8.6 shows for the Arabic→English testsets how many sentences had better, worse, or exactly equal sentence-level BLEU score when compared to each of the individual systems in a two-way combination of lattices. For mt0205tune, more than 27% of the LMBR system combination 1-best hypotheses $\hat{E}_+$ had a higher sentence-level BLEU score than both the system A hypothesis $\hat{E}_A$ and system B hypothesis $\hat{E}_B$. 13% of the system combination hypotheses had worse sentence-level BLEU score than both $\hat{E}_A$ and $\hat{E}_B$. So, although combination improves hypotheses more than twice as often as it degrades them, there are still many sentences for which it would be better not to apply combination. At the set-level, as shown by the BLEU scores in Table 8.1, LMBR system combination performs very well.

It is interesting to examine the differences in sentence-level BLEU score between the system combination hypothesis and the 1-best translations in the individual lattices. Let $\hat{E}_{i,k}$ denote the 1-best translation of the k^{th} sentence produced by the i^{th} system, $i = 1 \ldots M$, in an M-way combination of lattices. Let $\hat{E}_{+,k}$ denote the 1-best translation of the k^{th} sentence obtained by lattice MBR system combination of the M individual systems. The mean change

		$\hat{E}_+ < \hat{E}_A$	$\hat{E}_+ = \hat{E}_A$	$\hat{E}_+ > \hat{E}_A$
mt0205tune	$\hat{E}_+ < \hat{E}_B$	−6.20	−4.39	−0.06
	$\hat{E}_+ = \hat{E}_B$	−5.39	+0.00	+4.89
	$\hat{E}_+ > \hat{E}_B$	+0.12	+4.95	+7.76
mt0205test	$\hat{E}_+ < \hat{E}_B$	−5.75	−3.80	+0.09
	$\hat{E}_+ = \hat{E}_B$	−3.76	+0.00	+4.22
	$\hat{E}_+ > \hat{E}_B$	+0.82	+4.87	+7.46
mt08nw	$\hat{E}_+ < \hat{E}_B$	−5.97	−5.41	+0.51
	$\hat{E}_+ = \hat{E}_B$	−5.93	+0.00	+5.09
	$\hat{E}_+ > \hat{E}_B$	−0.53	+6.50	+7.21
mt08ng	$\hat{E}_+ < \hat{E}_B$	−4.71	−6.47	−0.18
	$\hat{E}_+ = \hat{E}_B$	−4.50	+0.00	+7.25
	$\hat{E}_+ > \hat{E}_B$	+1.65	+4.94	+5.95

Table 8.7: Mean change in sentence-level BLEU (MCB) for lattice minimum Bayes-risk two-way system combination of Arabic→English translation lattices.

in sentence-level BLEU score, MCB, in a set of S sentences is defined as

$$\text{MCB} = \frac{1}{S}\sum_{k=1}^{S}\frac{1}{M}\sum_{i=1}^{M}\Big\{\text{BLEU}_S(\hat{E}_{+,k}) - \text{BLEU}_S(\hat{E}_{i,k})\Big\}, \tag{8.8}$$

where $\text{BLEU}_S(E)$ is the sentence-level BLEU score of E given by Equation (8.7). The MCB is just the average difference in sentence-level BLEU between each individual system and the system combination hypothesis, averaged over the collection of sentences.

The mean change in sentence-level BLEU score is shown in Table 8.7. For mt0205tune, the sentences with higher scores than each of the individual systems gain on average +7.76 BLEU in system combination. These are very large gains. Unfortunately, the sentences for which MBR decoding selected a hypothesis with worse sentence-level BLEU score than each of the individual systems degrade by an average of -6.20 BLEU which is also very large. Similar patterns of gains and degradations are observed over all testsets. Part of the problem is that system tuning is optimised using corpus-level BLEU which includes the brevity penalty. In order to maximise corpus-level BLEU the length of some sentences may be increased or decreased inappropriately. What is really needed is a way to know when to take the system combination hypothesis and when to fall back to one of the individual system outputs.

8.4 Multi-Source Translation Experiments

Lattice minimum Bayes-risk decoding is readily applied to the task of multi-source translation (Section 8.1.2). The baseline for the following experiments is the Cambridge University Engineering Department (CUED) phrase-based statistical machine translation system (Blackwood et al., 2008a), as submitted to the ACL Third Workshop on Statistical Machine Translation (WMT) 2008 shared task.[1] CUED participated in two of the WMT shared task tracks: French→English and Spanish→English. The target language English output is the same for

[1] http://www.statmt.org/wmt08

both tracks so the goal is to improve translation quality by combining two separate translations generated from French and Spanish. The results presented in this section show that lattice minimum Bayes-risk decoding is a simple but highly effective framework for this form of multi-source system combination.

8.4.1 System Development

Table 8.8 summarises the parallel training data of the Europarl corpus, showing the number of sentences, number of words, and lower-cased vocabulary size for each language pair. All training and tuning was performed using only the parallel text and language model data distributed for the shared task.

	Sentences	Words	Vocabulary
FR	1.33M	39.9M	124k
EN		36.4M	106k
ES	1.30M	38.2M	140k
EN		35.7M	106k

Table 8.8: Number of sentences, number of words, and vocabulary size for French→English and Spanish→English Europarl translation. The difference in the number of English words for the two tracks is a result of limitations in the word alignment algorithm.

Word alignments were generated using GIZA++ (Och and Ney, 2003) over a stemmed version of the parallel text. Stems for each source language were obtained using the Snowball stemmer.[1] After unioning the Viterbi alignments and replacing stems with their original words, phrase-pairs of up to five foreign words in length were extracted in the usual fashion (Koehn et al., 2003).

The CUED WMT 2008 decoder follows the Transducer Translation Model (Kumar et al., 2006) (Section 4.2.3) and is implemented using the OpenFst Toolkit (Allauzen et al., 2007). Adjacent phrases can be reordered according to the MJ1 reordering model (Kumar and Byrne, 2005) with a uniform jump probability. Minimum error training (Och, 2003) under the BLEU score (Papineni et al., 2002b) optimises the following list of features with respect to the dev2006 development set: language model scale factor; word and phrase insertion penalties; reordering scale factor; insertion scale factor; source→target and target→source translation model scale factors. Three phrase-pair count features that track whether the phrase-pair occurred once, twice, or more than twice in the training data are also included (Bender et al., 2007).

The English language model is a modified Kneser-Ney (Kneser and Ney, 1995) smoothed 5-gram backoff language model estimated using SRILM (Stolcke, 2002) and converted to WFST format for use in TTM translation (Allauzen et al., 2003). TTM translation with MERT parameters results in word lattices that are then combined using the lattice minimum Bayes-risk decoder of Equation (8.5).

[1] http://snowball.tartarus.org

Configuration		dev2006		devtest2006		test2007	
		BLEU	NIST	BLEU	NIST	BLEU	NIST
FR→EN	TTM+MERT	31.9	7.65	32.5	7.72	32.9	7.81
	+LMBR	32.2	7.69	32.7	7.74	33.1	7.83
ES→EN	TTM+MERT	33.1	7.80	32.3	7.65	32.9	7.77
	+LMBR	33.2	7.85	32.6	7.70	33.4	7.84
FR→EN + ES→EN	LMBR	34.2	8.00	34.4	7.95	34.7	8.09

Table 8.9: Single-reference uncased BLEU and NIST scores for single-source and two-way minimum Bayes-risk multi-source translation of French (FR) and Spanish (ES) European Parliament proceedings into English (EN).

8.4.2 Results and Discussion

BLEU and NIST scores for single-source and multi-source translation of the dev2006, devtest2006, and test2007 Europarl evaluation sets are shown in Table 8.9. For this task, only small gains in BLEU are observed from single-system lattice minimum Bayes-risk decoding; this agrees with our previous experience of k-best list MBR when preparing our WMT 2008 submission (Blackwood et al., 2008a). The limited quantity of translation model training data and relatively simplistic reordering model result in lattices without the richness and diversity required for good gains with LMBR.

For multi-source translation much larger gains are observed. The lattices produced by each system are sufficiently different that the interpolated distribution over posteriors is able to select hypotheses with higher BLEU and NIST scores. The absolute gains in BLEU over the best of the two single systems involved in the combination are +1.1 for dev2006, +1.9 for devtest2006, and +1.8 for test2007. These are very large gains for BLEU scored with respect to a single reference translation. Optimisation of the interpolation weights of Equation (8.3) for the dev2006 set resulted in $\lambda_{FR} = \lambda_{ES} = 0.5$; this is as expected given that the baseline systems are of similar quality as measured by the BLEU score.

Even larger gains may be possible through lattice MBR multi-source translation of more than two language pairs, although careful tuning of the system-specific interpolation weights will be necessary.

8.5 Summary and Conclusions

This chapter has demonstrated the effectiveness of lattice minimum Bayes-risk decoding as a framework for the combination of multiple machine translation lattices. A multiple lattice MBR system combination decoder was introduced, and an efficient implementation based on weighted finite-state transducers was described. Large gains in BLEU score were obtained by combining Arabic→English and Chinese→English translation lattices generated from alternative analyses of the source language sentence. The quality of Finnish→English translation was improved by combining k-best lists generated from word-based and morpheme-based SMT systems. The lattice MBR decoder was also applied to multi-source translation. Large improvements in the BLEU score were obtained by combining lattices from French→English and Spanish→English SMT systems.

Lattice-based MBR system combination is very effective because it is able to operate on a much richer and more diverse space of hypotheses than traditional k-best list combination methods. The larger evidence space resulting from the combination of multiple lattices also allows for more accurate computation of the n-gram path posterior probabilities that drive the decoding process. The weighted interpolation over multiple evidence spaces provides greater robustness: aspects of translation handled poorly in one set of lattices may be compensated for by more appropriate handling in another set of lattices.

Each lattice encodes a potentially astronomical number of hypotheses. In k-best list decoding, the lists are normally limited to a fixed depth for efficiency reasons. If the maximum list size is 1000, then in three-way combination only 333 hypotheses from each system are considered and the majority share of the evidence space is completely ignored (see the discussion of lattice and k-best list probability masses in Section 7.3.2.2 of Chapter 7).

The lattice-based MBR system combination decoder introduced in this chapter is very general. It can be used for efficient combination of any set of multiple lattices produced by any technique, so long as the lattices represent a translation of the same foreign language sentence. Instead of the multi-input and multi-source translation described in this chapter, the decoding framework could be used to combine multiple lattices produced by any number of machine translation systems, even systems with very different architectures.

Chapter 9

Hypothesis Space Constraints for Statistical Machine Translation Fluency

This chapter develops a novel and robust approach to improving the quality of statistical machine translation within a lattice minimum Bayes-risk decoding framework. Segmentation of first-pass word lattices according to confidence measures over the maximum likelihood translation hypothesis makes it possible to focus on regions with potential errors in translation. Hypothesis space constraints based on high-order n-gram coverage in a large monolingual text collection are applied to partial hypotheses in low confidence regions in order to improve overall translation fluency.

Weighted finite-state transducer approaches to language model rescoring (Chapter 5) and lattice minimum Bayes-risk decoding (Chapter 7) are synthesised in this chapter to form a novel framework for improving MT fluency. This framework constitutes a new approach to machine translation decoding with great potential for future research. As with phrasal segmentation models (Chapter 6), the techniques introduced in this chapter represent an effective new way in which statistical machine translation can benefit from the exploitation of monolingual data that is ordinarily used only for building language models.

The rest of this chapter is organised as follows. Section 9.1 discusses the problem of poor fluency in existing approaches to machine translation and motivates the new framework proposed in this chapter. Section 9.2 introduces a confidence measure based on n-gram path posterior probabilities; this confidence measure is used to identify 'trusted' subsequences in the maximum likelihood translation hypothesis. Sections 9.3 and 9.4 describe a general framework for improving machine translation fluency based on a segmentation of the lattice into regions of high and low confidence. An application of this framework, monolingual coverage constraints, is presented in Section 9.5. Lattice generation procedures are described in Section 9.6, and a human evaluation of translation fluency is presented in Section 9.7. The chapter concludes, with suggestions for future research, in Section 9.8.

9.1 Introduction and Motivation

Translation quality is often described in terms of *fluency* and *adequacy*. Fluency reflects the 'nativeness' of the translation, while adequacy indicates how well a translation captures the meaning of the original text (Ma and Cieri, 2006).

From a purely utilitarian point of view, adequacy should be more important than fluency. But fluency and adequacy are subjective and not easy to tease apart (Callison-Burch et al., 2009, 2006; Vilar et al., 2007). There is a human tendency to rate less fluent translations as less adequate. One explanation for this is that errors in grammar cause readers to be more critical. A related phenomenon is that the nature of translation errors changes as fluency improves so that whatever errors are present in fluent translations must necessarily be relatively subtle. It is therefore not enough to focus solely on adequacy in translation. SMT systems must also be fluent if they are to be accepted and trusted. It may be that the reliance on automatic metrics has led SMT researchers to pay insufficient attention to fluency. Automatic metrics such as BLEU (Papineni et al., 2002b), TER (Snover et al., 2006), and METEOR (Lavie and Denkowski, 2009) show broad correlation with human rankings of MT quality, but are not capable of fine distinctions between fluency and adequacy.

There is a growing concern that the fluency of current SMT systems is inadequate (Knight, 2007a). SMT is robust, in that a translation is nearly always produced. But unlike translators who should be skilled in at least one of the languages involved, SMT systems are limited in both source language and target language competence. SMT fluency and accuracy therefore tend to suffer together as translation quality degrades. This should not be the case. Ideally, an SMT system should never be any less fluent than the best stochastic text generation (STG) system available in the target language (Oberlander and Brew, 2000). What is needed is a good way to enhance the fluency of SMT hypotheses.

The maximum likelihood (ML) formulation (Brown et al., 1990) (Chapter 4, Section 4.1.1) of translation of a source language sentence F to a target language sentence $\hat{E}$

$$\hat{E} = \underset{E}{\operatorname{argmax}}\, P(F|E)P(E) \tag{9.1}$$

makes it clear why improving SMT fluency is a difficult modelling problem. The language model $P(E)$, the closest thing to a 'fluency component' in the original formulation, only affects candidates which are likely under the translation model $P(F|E)$. Given the weakness of current translation models this is a severe limitation. For example, it often happens that SMT systems assign $P(F|\bar{E}) = 0$ to a correct reference translation $\bar{E}$ of F. This is one of the

reasons why translation systems are usually not used to align parallel text and also why one SMT system often fails in re-scoring the hypotheses of a second SMT system. The cause is often as minor as one or two words or phrases which cannot be aligned or translated. This problem, sometimes called the 'reachability' problem, motivates the integration of natural language generation systems in statistical machine translation, an area for future work that is discussed in more depth in the conclusions to this thesis (Chapter 10, Section 10.3).

In such situations the reference translation is not even a valid candidate under the translation model. The problem is that in ML decoding the language model can only encourage the production of fluent translations; it cannot easily enforce constraints on fluency or introduce new hypotheses. Simply replacing the language model by a generation system will not overcome this limitation. This analysis applies to phrase-based SMT and the situation is similar in syntax-based SMT.

In syntax-based SMT, the primary role of syntax is to drive the translation process. The translations produced by these systems respect the syntax of their translation models, but this does not force them to be grammatical in the way that a typical human sentence is grammatical. In Hiero (Chiang, 2005, 2007), the grammar consists of hierarchical rules for the movement and translation of words and phrases (see Chapter 4, Section 4.3). Hiero allows complex long-range movement, but as a grammar it imposes little constraint on the generation of target language sentences. In tree transduction grammars (Knight and Graehl, 2005; Knight, 2007b), parse trees generated by source language parsers are mapped to trees in a target language grammar. The grammar must have broad enough coverage to accept the trees which are generated automatically by the stochastic analysis and translation processes. These systems are very powerful in the types of translations they support, but they allow many translations which are not fluent. The problem is the need for robustness. Generating fluent translations demands a tightly constraining target language grammar but such a grammar is at odds with the broad-coverage parsing needed for robust translation.

There are thus two main problems in translation fluency: (i) SMT may fail to generate fluent hypotheses and there is no simple way to introduce them into the search; (ii) SMT can produce many translations which are not fluent, but tightening syntactic constraints to improve fluency can hurt robustness. Both problems are rooted in the maximum likelihood decoding framework in which robustness and fluency are conflicting objectives.

This chapter proposes a novel decoding framework to improve the fluency of any SMT system, whether syntactic or phrase-based. The idea is to perform Minimum Bayes-risk search (Kumar and Byrne, 2004) over a space of fluent hypotheses $\mathcal{H}$:

$$\hat{E}_{\text{MBR}} = \underset{E' \in \mathcal{H}}{\text{argmin}} \sum_{E \in \mathcal{E}} L(E, E') P(E|F) \tag{9.2}$$

In this approach the MBR evidence space $\mathcal{E}$ is generated by an SMT system as a k-best list or lattice. The SMT system runs in its best possible configuration, thus ensuring both translation robustness and good baselines. Rather than constraining hypothesis search to the output of the SMT system, translations will be sought among the collection of fluent sentences which are close to the top SMT hypotheses as determined by the loss function $L(E, E')$.

Decoupling the MBR hypothesis space from first-pass translation offers great flexibility. Hypotheses in $\mathcal{H}$ may be arbitrarily constrained according to lexical, syntactic, semantic, or other considerations, with no effect on translation robustness. This is because constraints on fluency do not affect the production of the evidence space by the baseline system. Robustness and fluency are no longer conflicting objectives. This framework also allows the MBR

hypothesis space to be augmented or replaced with new hypotheses produced by a natural language generation system, with great potential for improved translation fluency.

This chapter focuses on searching out fluent strings amongst the vast number of hypotheses encoded in SMT lattices. Oracle BLEU scores computed over k-best lists have shown that many high quality hypotheses are produced by first-pass SMT decoding (Och et al., 2004). The difficulty of enhancing the fluency of complete hypotheses is reduced by first identifying regions of high-confidence in the ML translation and using these to guide the fluency refinement process. This has two advantages: (i) portions of the baseline hypotheses that are trusted are retained and alternatives searched for elsewhere, and (ii) the task is made much easier since the fluency of sentence fragments can be refined in the context of their high-confidence neighbours. Section 9.6 will show that the fluency of the MBR hypothesis space can be refined with no real degradation in the BLEU score compared to MBR decoding over an unconstrained first-pass lattice.

The formulation of the MBR decoder in Equation (9.2) separates the hypothesis space from the evidence space. Linearised lattice MBR (Tromble et al., 2008) rewrites the loss in terms of a gain and replaces the sum over hypotheses with a sum over lattice n-grams to give

$$\hat{E}_{\text{LMBR}} = \underset{E' \in \mathcal{H}}{\text{argmax}} \left\{ \theta_0 |E'| + \sum_{u \in \mathcal{N}} \theta_u \#_u(E') p(u|\mathcal{E}) \right\}, \tag{9.3}$$

where $\mathcal{H}$ is the hypothesis space, $\mathcal{E}$ is the evidence space, $\mathcal{N}$ is the set of all n-grams in $\mathcal{H}$ (typically, $n = 1 \ldots 4$), and θ are constants estimated on held-out data. The quantity $p(u|\mathcal{E})$ is the path posterior probability of the n-gram u

$$p(u|\mathcal{E}) = \sum_{E \in \mathcal{E}_u} P(E|F), \tag{9.4}$$

where $\mathcal{E}_u = \{E \in \mathcal{E} : \#_u(E) > 0\}$ is the subset of lattice paths containing n-gram u at least once. These path posterior n-gram probabilities can be efficiently calculated using general purpose WFST operations (Blackwood and Byrne, 2010) (Chapter 7, Section 7.2).

9.2 Posterior Probability Confidence Measures

In the formulation of Equations (9.3) and (9.4) the path posterior n-gram probabilities play a crucial role. Minimum Bayes-risk decoding under the linear approximation to BLEU is driven mainly by the presence of high posterior n-grams in the lattice; the low posterior n-grams contribute relatively little to the MBR decision criterion. Here, the predictive power of these statistics is investigated. The n-gram path posterior probabilities will be shown to be a good predictor as to whether or not an n-gram is to be found in a set of reference translations and hence whether it should be included in a translation hypothesis.

For each sentence, let $\mathcal{N}_n$ denote the set of n-grams of order n in the first-pass ML translation 1-best hypothesis $\hat{E}$, and let $\mathcal{R}_n$ denote the set of n-grams of order n in the union of the references. For confidence threshold β, let $\mathcal{N}_{n,\beta} = \{u \in \mathcal{N}_n \colon p(u|\mathcal{E}) \geq \beta\}$ denote the set of all n-grams in $\mathcal{N}_n$ with posterior probability greater than or equal to β, where $p(u|\mathcal{E})$ is computed according to Equation (9.4). This is equivalent to identifying all substrings of length n in the translation hypotheses for which the system assigns a posterior probability of

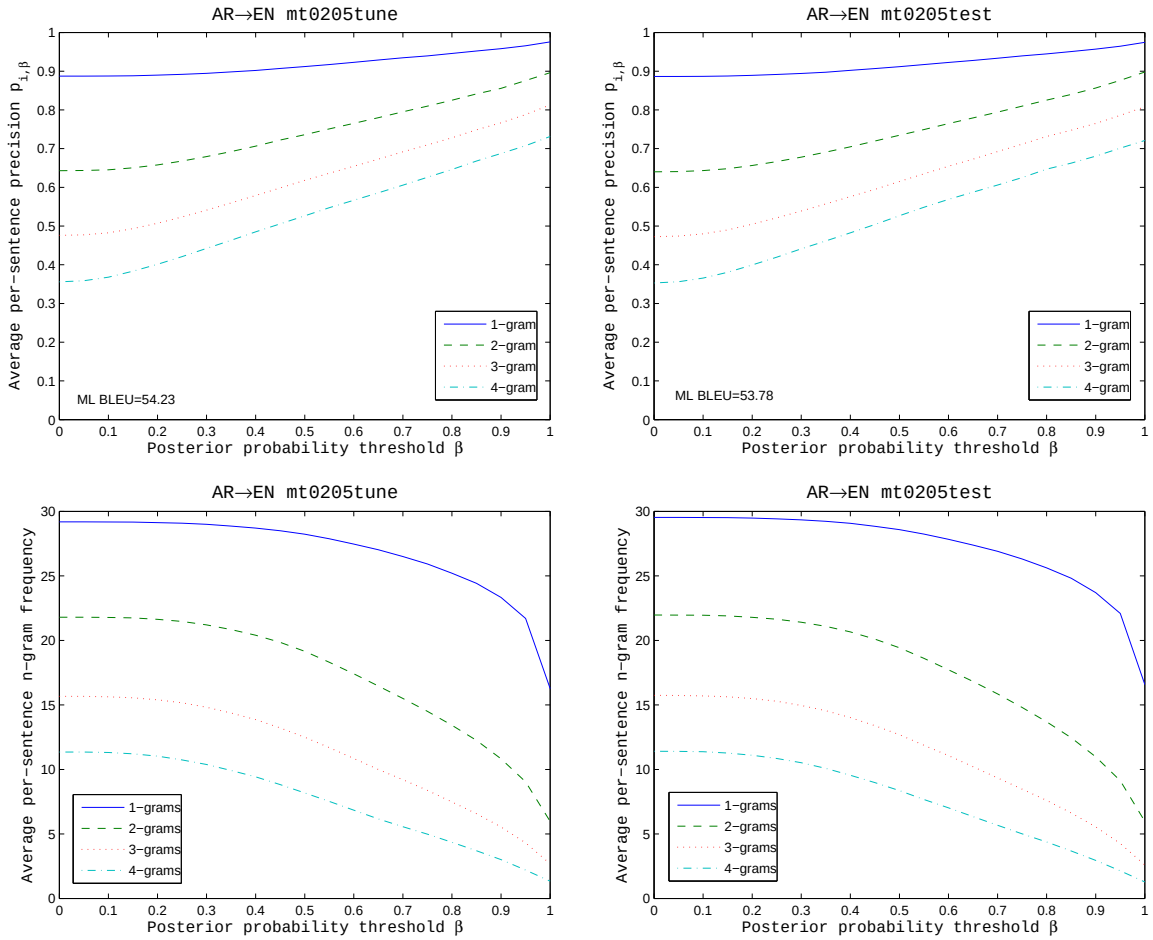


Figure 9.1: Average per-sentence n-gram precisions (top) and counts (bottom) for Arabic→English ML translations at a range of posterior probability thresholds $0 \leq \beta \leq 1$.

β or higher. The precision at order n for threshold β is the proportion of n-grams in $\mathcal{N}_{n,\beta}$ also present in the references:

$$\mathcal{P}_{n,\beta} = \frac{|\mathcal{R}_n \cap \mathcal{N}_{n,\beta}|}{|\mathcal{N}_{n,\beta}|} \tag{9.5}$$

9.2.1 Single-System Reference Precisions

The upper plots in Figure 9.1 show the average per-sentence n-gram precisions $\mathcal{P}_{n,\beta}$ at orders $1, 2, 3$, and 4 for the Arabic→English translation testsets mt0205tune and mt0205test, over a range of posterior probability thresholds $0 \leq \beta \leq 1$. Sentence start and end tokens are ignored when computing unigram precisions. The plots show that precisions at all orders improve considerably as the threshold β increases. This confirms that these intrinsic measures of translation confidence have strong predictive power. Note that the upper plots show at $\beta = 0$ the n-gram precisions used to compute the BLEU score of the ML baseline system.

The lower plots in the figure show the average number of n-grams per sentence at each order for the same range of β. For high β, there are relatively few n-grams with $p(u|\mathcal{E}) \geq \beta$;

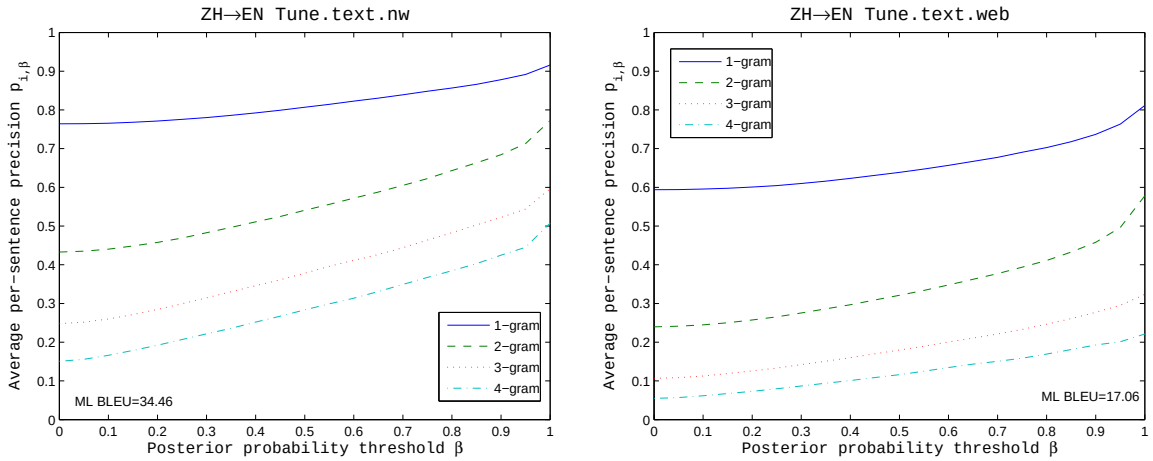


Figure 9.2: Average per-sentence n-gram precisions (top) and counts (bottom) for Chinese→English ML translations at a range of posterior probability thresholds $0 \leq \beta \leq 1$.

this is as expected. However, even at a high threshold of $\beta = 0.9$ there are still an average of three 4-grams per sentence with posterior probabilities that exceed that threshold. Therefore, even at very high levels of confidence, high posterior probability higher-order n-grams still occur frequently enough to be useful.

Precision plots for GALE Chinese→English newswire and web data translations are shown in Figure 9.2. The precision scores at all orders are considerably lower than those observed in Arabic→English translation. This is to be expected given the much lower BLEU score obtained in translation from Chinese. The web data translations in particular have very low 3-gram and 4-gram precisions. That the 3-gram and 4-gram precisions do not vary much as the posterior probability threshold β is increased suggests that these higher-order n-grams do not usefully discriminate amongst hypotheses during minimum Bayes-risk decoding.

The precision results presented in this section motivate the use of n-gram path posterior probabilities as a statistical machine translation confidence measure. Confidence $p(\hat{E}_i^j|\mathcal{E})$ is assigned to subsequences $\hat{E}_i \ldots \hat{E}_j$ of the ML translation hypothesis.

Prior work focuses on word-level confidence measures extracted from k-best lists and word graphs (Ueffing and Ney, 2005, 2007), while Zens and Ney (2006) rescore relatively shallow k-best lists with n-gram posterior probabilities. Similar experiments using different statistics and with a different motivation are reported by DeNero et al. (2009); they show that expected counts of n-grams obtained from a lattice can be used to predict which n-grams appear in the references.

9.2.2 Evidence Space Size and Reference Precisions

One of the main advantages of lattice minimum Bayes-risk decoding over a k-best implementation is that a much larger evidence space and hypothesis space can be considered. The following experiment shows that the larger evidence space of the lattice is useful for obtaining improved posterior probability estimates.

The Arabic→English mt0205tune and mt0205test testset 4-gram reference precisions at a range of posterior probability thresholds β are shown in Figure 9.3. The posterior probabilities

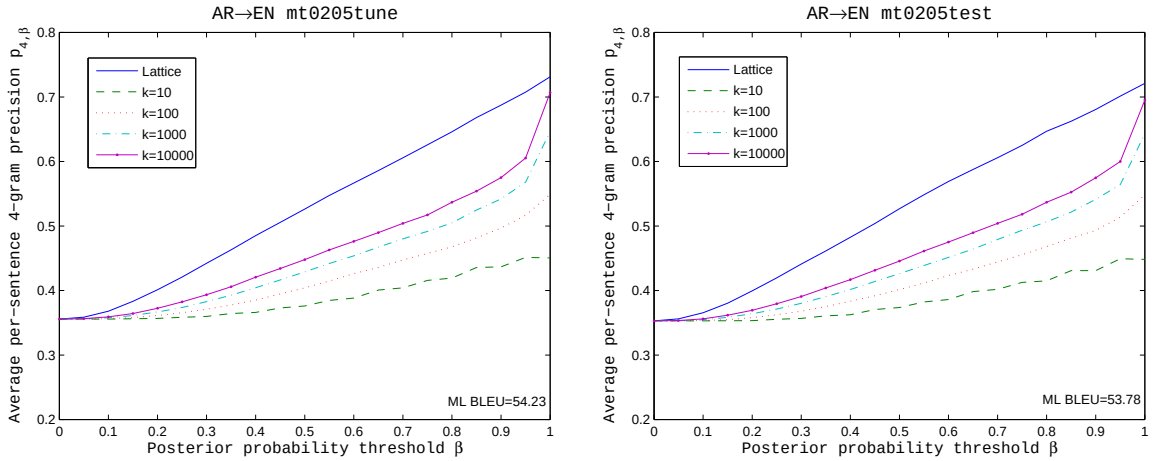


Figure 9.3: Average per-sentence 4-gram precisions for NIST MT08 Arabic→English ML 1-best computed using the full lattice and k-best lists of the specified sizes.

are computed using either the full lattice $\mathcal{E}$ or a k-best list of the specified size. The 4-gram precision of the 1-best translations is around 0.35. At higher values of β, the reference precision increases considerably to around 0.70. Expanding the k-best list size from 1000 to 10000 translation hypotheses only slightly improves the precision, but much higher precisions are observed when the full evidence space of the lattice is used. The improved level of 4-gram precision is a result of more accurate estimates of n-gram posterior probabilities using the full lattice. These precision plots re-emphasise the advantage of lattice-based decoding and rescoring procedures, previously shown in the comparison of evidence space size and MBR decoding performance in Chapter 7, Section 7.3.2.2.

These precision plots show that although any hypotheses beyond the 10000^{th} hypothesis in a k-best list might have a very low posterior probability, the aggregate probability of all hypotheses beyond the 10000^{th} is substantial and useful for accurate estimation of n-gram posterior probabilities.

9.2.3 System Combination Reference Precisions

Minimum Bayes-risk decoding of multiple translation lattices generated from alternative decompositions of the input sentence has been demonstrated to significantly improve the BLEU score (see Chapter 8, Section 8.3). This section shows that n-gram posterior probabilities computed from a combination of multiple lattices have higher reference precisions.

Given evidence space $\mathcal{E} = \bigoplus_{i=1}^{M} \mathcal{E}^{(i)}$ formed from the union of M individual translation lattices $\mathcal{E}^{(1)}, \ldots, \mathcal{E}^{(M)}$, the interpolated n-gram posterior probability $p(u|\mathcal{E})$ can be computed using one of two methods:

Linear Interpolation The posterior probability of n-gram u computed according to a linear interpolation of the posterior probability in each of the M lattices is

$$p(u|\mathcal{E}) = \sum_{i=1}^{M} \lambda_i \, p_i(u|\mathcal{E}^{(i)}), \quad 0 \leq \lambda_i \leq 1, \quad \sum_{i=1}^{M} \lambda_i = 1. \tag{9.6}$$

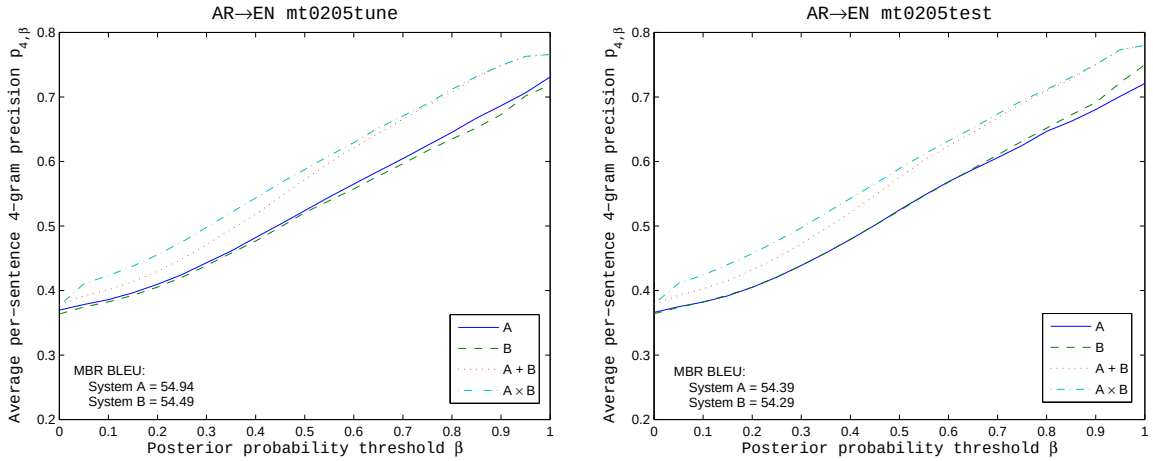


Figure 9.4: Average per-sentence 4-gram precisions for Arabic→English single-system system combination MBR 1-best translations at a range of posterior probability thresholds β.

Product Interpolation The posterior probability of n-gram u computed according to a product interpolation of the posterior probability in each of the M lattices is taken as

$$p(u|\mathcal{E}) = \prod_{i=1}^{M} p_i(u|\mathcal{E}^{(i)})^{\lambda_i}, \quad 0 \leq \lambda_i \leq 1, \quad \sum_{i=1}^{M} \lambda_i = 1. \tag{9.7}$$

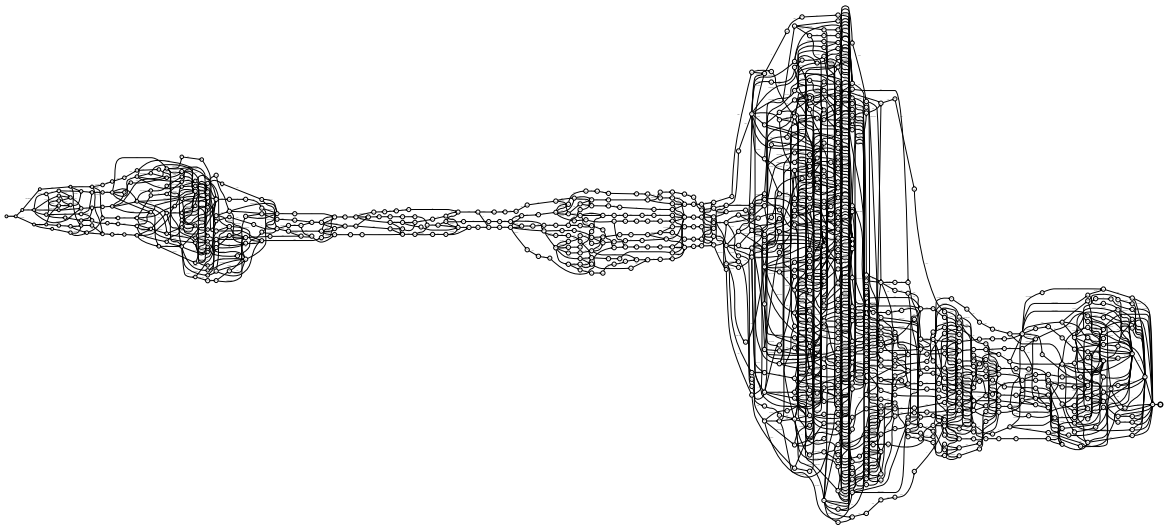
For the special case of combining two equally weighted lattices $\mathcal{E}^{(1)}$ and $\mathcal{E}^{(2)}$ the interpolation weights are $\lambda_1 = \lambda_2 = \frac{1}{2}$. The product interpolation simplifies to the geometric mean:

$$p(u|\mathcal{E}) = p_1(u|\mathcal{E}^{(1)})^{\frac{1}{2}} \times p_2(u|\mathcal{E}^{(2)})^{\frac{1}{2}} = \sqrt{p_1(u|\mathcal{E}^{(1)}) \times p_2(u|\mathcal{E}^{(2)})}. \tag{9.8}$$

Figure 9.4 shows how 4-gram precision varies as a function of β for two single systems A and B, and their combination using a linear interpolation of posterior probabilities (line $A+B$) or a product interpolation of posterior probabilities (line $A \times B$). The 4-gram precisions of the individual systems are very close over the full range of β. This is as expected given that the single-system BLEU scores are so similar; it is also evidence confirming that the optimal interpolation weights for this particular combination should be equal. The system combination 4-gram precisions are higher for both linear interpolation and product interpolation. For $\beta \geq 0.6$ there is no real difference between the two forms of interpolation. For lower values of β, the product interpolation has a higher precision than the linear interpolation. The precision obtained using n-gram posterior probabilities computed from the combined lattices is higher than that of the individual systems. A higher proportion of the n-grams assigned high posterior probability under the interpolated distribution are found in the references; this is one of the reasons for the improved BLEU score of lattice MBR system combination.

The precision results presented in this section show that the reliability of the n-gram path posterior probability confidence measure can be improved by interpolating the distribution over multiple first-pass translation lattices.

<s> the newspaper " constitution " quoted brigadier abdullah krishan **, the chief of police in** karak governorate **(521 km south @-@ west of amman) as saying that the seizure** took place after **police received information that** there were attempts by the group to sell for more than $ 100 thousand dollars , **the police rushed to** the arrest in possession . </s>



$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$	$\mathcal{H}_4$	$\mathcal{H}_5$	$\mathcal{H}_6$	$\mathcal{H}_7$	$\mathcal{H}_8$	$\mathcal{H}_9$
433	1	4	1	6	1	6860	1	76

Figure 9.5: ML translation $\hat{E}$, word lattice $\mathcal{E}$, and segmentation as a sequence of four string and five sublattice regions $\mathcal{H}_1 \ldots \mathcal{H}_9$ using n-gram posterior probability threshold $p(u|\mathcal{E}) \geq 0.8$.

9.3 Lattice Segmentation Under Posterior Distributions

The study of reference precisions in the previous section shows that current SMT systems, although flawed, can identify with confidence partial hypotheses that can be trusted. It is potentially useful to constrain MBR decoding to include these trusted partial hypotheses but otherwise allow decoding to consider alternatives in the regions of low confidence. In this way it is possible to improve the best possible output of the best available systems.

The n-gram path posterior probabilities of Equation (9.4) can be used to segment a lattice $\mathcal{E}$ into regions of high and low confidence. An example ML 1-best translation and segmented lattice is shown in Figure 9.5. The words of the ML 1-best covered by high confidence n-grams are marked in bold. The number of hypotheses in each region is also given. As this example shows, lattice segmentation is performed relative to the ML hypothesis $\hat{E}$, i.e. relative to the best path through $\mathcal{E}$ according to the first-pass translation decoder.

Lattice segmentation is performed in the following way. For confidence threshold β, find all 4-grams $u = \hat{E}_i, \ldots, \hat{E}_{i+3}$ in the ML translation hypothesis for which $p(u|\mathcal{E}) > \beta$. Then segment $\hat{E}$ into regions of high and low confidence where the high confidence regions are identified by consecutive, overlapping high confidence 4-grams. The high confidence regions are contiguous strings of words for which there is consensus amongst the translations in the lattice. If the path posterior n-gram probabilities are trusted, then any hypothesised translation should include these high confidence substrings.

The ML hypothesis string $\hat{E}$ is in this way segmented into R alternating subsequences of high and low confidence. The segment boundaries are i_r and j_r so that $\hat{E}_{i_r}^{j_r}$ is either a high confidence or a low confidence subsequence. Each subsequence is associated with an unweighted subspace $\mathcal{H}_r$. This subspace has the form of a string for high confidence regions and the form of a lattice for low confidence regions. Figure 9.5 shows the series of nine unweighted subspaces $\mathcal{H}_1, \ldots, \mathcal{H}_9$ obtained by segmenting the lattice using $\beta = 0.8$. This form of segmentation into regions of high and low confidence is related to segmental MBR for automatic speech recognition (Goel et al., 2004).

If the r^{th} segment is a high confidence region then $\mathcal{H}_r$ accepts only the string $\hat{E}_{i_r}^{j_r}$. If the r^{th} segment is a region of low confidence, then $\mathcal{H}_r$ is built to accept relevant substrings from $\mathcal{E}$. It is constructed as follows. The r^{th} low confidence region $\hat{E}_{i_r}^{j_r}$ has a high confidence left context $\hat{e}_{r-1}$ and a high confidence right context $\hat{e}_{r+1}$ formed from subsequences of the ML translation hypothesis $\hat{E}$ as

$$\hat{e}_{r-1} = \hat{E}_{i_{r-1}}^{j_{r-1}} \tag{9.9}$$

$$\hat{e}_{r+1} = \hat{E}_{i_{r+1}}^{j_{r+1}} \tag{9.10}$$

When $r = 1$ the left context $\hat{e}_{r-1}$ is defined to be the empty string and when $r = R$ the right context $\hat{e}_{r+1}$ is defined to be the empty string. A transducer $\mathcal{T}_r$ for the regular expression $/.*\hat{e}_{r-1}(.*)\hat{e}_{r+1}.*/\backslash 1/$ is constructed for finding all subsequences in $\mathcal{E}$ associated with $\mathcal{H}_r$. In this notation, parentheses indicate string matches. For example $/.*y(a*)w.*/\backslash 1/$ applied to *xyaaawzz* yields *aaa*. Composition with $\mathcal{E}$ yields $\mathcal{H}_r = \mathcal{E} \circ \mathcal{T}_r$, so that $\mathcal{H}_r$ contains all the reasonable alternatives to $\hat{E}_{i_r}^{j_r}$ in $\mathcal{E}$ consistent with the high confidence left and right string contexts $\hat{e}_{r-1}$ and $\hat{e}_{r+1}$ in the ML 1-best translation.

If $\mathcal{H}_r$ is aligned to a high confidence subsequence of $\hat{E}$, it is called a *string region* since it contains only a single string; if $\mathcal{H}_r$ is aligned to a low confidence region, then it is a lattice of partial translation hypotheses and is called a *sublattice region*. The series of high and low confidence subspace regions $\mathcal{H}_1, \ldots, \mathcal{H}_R$ defines the segmentation of the lattice. The segmentation example in Figure 9.5 contains four string and five sublattice regions.

9.3.1 Segmentation Transducers

WFST operations can be used to efficiently segment the lattice by extracting sublattices corresponding to low confidence regions of the ML translation. If the lattice $\mathcal{E}$ contains at least one high confidence string region, then each low confidence sublattice region occurs in one of three possible orientations: (i) to the left of a string region; (ii) to the right of a string region; or (iii) between two string regions. For the lattice segmentation shown in Figure 9.5, the sublattice $\mathcal{H}_1$ occurs to the left of a string region, the sublattice $\mathcal{H}_9$ occurs to the right of a string region, and the other sublattices all occur between two string regions.

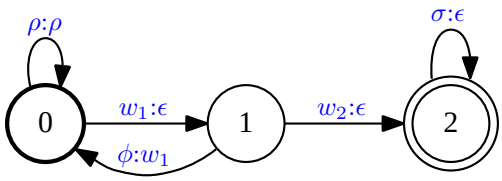


Figure 9.6: Example transducer M_u^L for matching sublattice region to the left of $u = w_1 w_2$.

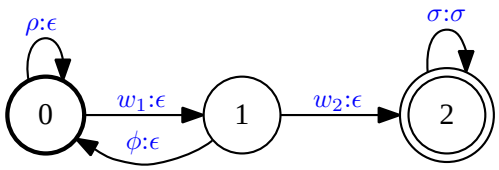


Figure 9.7: Example transducer M_u^R for matching sublattice region to the right of $u = w_1 w_2$.

Transducers M_u^L and M_u^R are introduced for extracting sublattice regions relative to a specified n-gram u. M_u^L extracts the sublattice region consisting of all partial path prefixes that occur before (i.e. to the left of) u. M_u^R extracts the partial path suffixes that occur after (i.e. to the right of) u. Both transducers work by mapping the n-gram u and all symbols not in the required sublattice region to ϵ-arcs.

Sublattice regions are extracted by composing $\mathcal{E} \circ M_u^L$ or $\mathcal{E} \circ M_u^R$, removing the weights, projecting on the output labels, removing ϵ-arcs, and then determinizing and minimising. The resulting acceptor represents the application of the subsequence regular expression to the lattice $\mathcal{E}$. It contains the set of all partial translation alternatives corresponding to the words of the ML 1-best that cover the low confidence region. Figures 9.6 and 9.7 show transducer examples M_u^L and M_u^R for the bigram $u = w_1 w_2$.[1]

Sublattice regions located between two high posterior n-grams u_1 and u_2 are extracted by composing $\mathcal{E} \circ M_{u_1}^R \circ M_{u_2}^L$, followed by output label projection and the same sequence of WFST optimisation operations.

9.4 Hypothesis Space Construction

This section introduces a general framework for improving the fluency of the hypothesis space $\mathcal{H}$ in lattice MBR decoding. The lattice segmentation process described in Section 9.3 considerably simplifies the problem of improving the fluency of $\mathcal{H}$ since each region of low confidence may be considered independently. The low confidence regions can be transformed one-by-one, and then reassembled to form an improved MBR hypothesis space.

In order to transform the sublattice region $\mathcal{H}_r$ it is important to know the context in which it occurs, i.e. the sequences of words that form its prefix and suffix. Some transformations might need only a short context; others might need a sentence-level context, i.e. the full sequence of ML words $\hat{E}_1^{j_{r-1}}$ and $\hat{E}_{i_{r+1}}^N$ to the left and right of the region $\mathcal{H}_r$ to be transformed.

[1]The special symbol σ (all) matches and consumes any arc during composition; ρ (rest) matches and consumes any arc other than those with an explicit transition from the state. ϕ (fail) is the non-consuming equivalent of ρ. Special label matching is described in Chapter 2, Section 2.4.2

To put this formally, each low confidence sublattice region is transformed by the application of some function Ψ:

$$\mathcal{H}_r \leftarrow \Psi(\hat{E}_1^{j_{r-1}},\ \mathcal{H}_r,\ \hat{E}_{i_{r+1}}^{N}) \tag{9.11}$$

The hypothesis space is then constructed from the concatenation of high confidence string regions and transformed low confidence sublattice regions:

$$\mathcal{H} = \mathcal{E} \circ \left\{ \bigotimes_{1 \le r \le R} \mathcal{H}_r \right\} \tag{9.12}$$

The composition with the original lattice $\mathcal{E}$ discards any new hypotheses that might be introduced via the unconstrained concatenation of strings from the regions $\mathcal{H}_r$. It may be that in some circumstances the introduction of new paths is good, but the experiments that follow test the ability to improve fluency by searching among existing hypotheses; the composition with $\mathcal{E}$ in Equation (9.12) ensures that no new hypotheses are created.

Section 9.5 describes an implementation of the function Ψ based on monolingual coverage constraints in a large collection of target language text.

9.4.1 Segmented Hypothesis Space Size

If no new hypotheses are introduced by the operation of Ψ, the size of the hypothesis space $\mathcal{H}$ is determined by the path posterior n-gram probability threshold β. Only the ML hypothesis remains at $\beta = 0$, since all of its subsequences are of high confidence, i.e. can be covered by n-grams with non-zero path posterior probability. At the other extreme, for $\beta = 1$, it follows that $\mathcal{H} = \mathcal{E}$ and no paths are removed, since either no n-gram subsequences have posterior probability equal to 1, or any subsequences u with $p(u|\mathcal{E}) = 1$ occur on every path in $\mathcal{E}$.

The threshold β can be viewed as a 'tunable knob' that can be used to tighten or relax constraints on the LMBR hypothesis space. For $\beta = 0$, LMBR decoding returns only the ML hypothesis; for $\beta = 1$, LMBR is performed over the full translation lattice. The effect of β on the BLEU score obtained by LMBR decoding is investigated in Section 9.6.

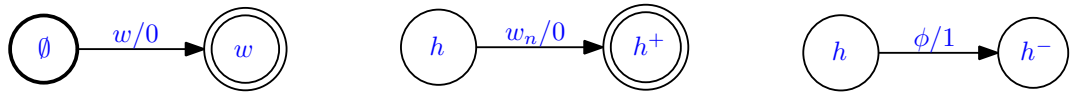
The size of the hypothesis space at a given value of β is the product of the number of sequences in the sublattice regions. If $|\mathcal{H}_r|$ denotes the number of hypotheses in region r, then the total number of hypotheses in the MBR hypothesis space is the product $\prod_{i=1}^{R} |\mathcal{H}_r|$. For Figure 9.5 at $\beta = 0.8$, this product is more than 5.4 billion hypotheses, showing that even for fairly aggressive constraints on the hypothesis space, many hypotheses remain.

9.5 Monolingual Coverage Constraints for Translation Fluency

This section describes a simple implementation of the transformation function Ψ that results in improved machine translation fluency. This transformation is based on n-gram coverage in a large target language text collection: where possible, the sublattice regions are filtered so that they contain only long-span n-grams which have been previously observed in the target language text. The motivation is that large monolingual text collections are good guides to fluency. If a partial hypothesis is composed entirely of previously seen higher-order n-grams, it is likely to be fluent and should therefore be favoured over other partial hypotheses with only lower-order n-gram coverage.

Translation hypothesis E and n-gram orders used by the LM to score each word	Score
<s>$_1$ the$_2$ reactor$_3$ produces$_3$ plutonium$_2$ *needed*$_2$ *to*$_3$ *manufacture*$_4$ atomic$_3$ bomb$_2$.$_3$ </s>$_4$	-22.59
<s>$_1$ the$_2$ reactor$_3$ produces$_3$ plutonium$_2$ *needed*$_2$ *to*$_3$ *manufacture*$_4$ *the*$_4$ atomic$_2$ bomb$_3$.$_4$ </s>$_4$	-23.61
<s>$_1$ the$_2$ reactor$_3$ produces$_4$ plutonium$_5$ *needed*$_3$ *to*$_3$ *manufacture*$_4$ atomic$_5$ bomb$_2$.$_3$ </s>$_4$	-16.04
<s>$_1$ the$_2$ reactor$_3$ produces$_4$ plutonium$_5$ *needed*$_3$ *to*$_3$ *manufacture*$_4$ *the*$_4$ atomic$_4$ bomb$_5$.$_4$ </s>$_5$	-17.96

Figure 9.8: Scores and n-gram orders for hypotheses using 4-gram Kneser-Ney (top) and 5-gram stupid-backoff (bottom) language models. Low confidence regions are in italics.



(a) (b) (c)

Figure 9.9: Unigram arcs (a), higher-order n-gram arcs (b), and backoff arcs (c) required for the implementation of the maximum order n monolingual coverage constraints acceptor $\mathcal{C}_n$.

Initial attempts to identify fluent hypotheses in the sublattice regions by ranking hypotheses according to n-gram language model scores were not effective. Figure 9.8 provides an example of the difficulties. For both the first-pass 4-gram Kneser-Ney (Kneser and Ney, 1995) language model (estimated over 1.1 billion tokens) and second-pass 5-gram stupid-backoff (Brants et al., 2007) language model (estimated over 6.6 billion tokens), the LM score $-\log P(E)$ favours the shorter but disfluent hypothesis; normalising by length did not help. However, the stupid-backoff LM has better coverage and the backing-off behaviour is a clue to the presence of a disfluency. Similar cues have been observed in ASR analysis (Chase, 1997). The shorter hypothesis backs off to a bigram for "atomic bomb", whereas the longer hypothesis covers the same words with higher order n-grams. The LM scores are therefore disregarded and the transformation Ψ is instead implemented using n-gram coverage. This is an example where robustness and fluency are at odds. The backoff n-gram models are robust, but are often found to favour less fluent hypotheses.

Let $\mathcal{S}$ denote the set of all n-grams in the monolingual training data. To identify partial hypotheses in sublattice regions that have complete monolingual coverage at some maximum order n, a coverage acceptor $\mathcal{C}_n$ is constructed with a similar form to the WFST representation of a backoff n-gram language model (Allauzen et al., 2003) (see Chapter 3, Section 3.4). $\mathcal{C}_n$ assigns a penalty to every n-gram not found in $\mathcal{S}$. Word arcs in $\mathcal{C}_n$ have no cost and backoff arcs are assigned a fixed cost of 1. Firstly, arcs from the null history start state $\emptyset$ are added for unigrams $u \in \mathcal{N}_1$ with the form shown in Figure 9.9 (a). Then, for higher-order n-grams $u \in \{\mathcal{S} \cap \{\cup_{i=2}^{n} \mathcal{N}_i\}\}$, where $u = w_1^n$ with history $h = w_1^{n-1}$ and target word w_n, arcs are added with the form shown in Figure 9.9 (b), where $h^+ = w_2^{n-1}$ if u has order n, and $h^+ = w_1^n$ if u has order less than n. Backoff arcs that implement the fixed penalty are then added for each u as shown in Figure 9.9 (c), where $h^- = w_2^{n-1}$ for $|u| > 2$ represents the state encoding the backed-off history; bigrams backoff to the null history start state $\emptyset$.

Each sublattice region $\mathcal{H}_r$ should penalise paths proportionally to the number of n-grams on the path not found in the monolingual text collection $\mathcal{S}$. This should be done in context, so that the effect of the neighbouring high confidence regions $\mathcal{H}_{r-1}$ and $\mathcal{H}_{r+1}$ is incorporated. Given that n-grams are counted at order n, a left context machine $\mathcal{L}_r$ is constructed to accept the *last* $n-1$ words in $\mathcal{H}_{r-1}$; similarly, $\mathcal{R}_r$ is constructed to accept the *first* $n-1$ words of

Generated String
for the manufacture of atomic bombs
manufacture of atomic bombs . </s>
needed for the manufacture of atomic
needed for the manufacture of the
plutonium for the manufacture of atomic
plutonium for the manufacture of the
required for the manufacture of atomic
the manufacture of atomic bombs .
the reactor produced plutonium for the
to manufacture atomic bombs . </s>
for the manufacture of atomic bombs .
plutonium for the manufacture of atomic bombs
required for the manufacture of atomic bombs
the manufacture of atomic bombs . </s>
for the manufacture of atomic bombs . </s>
needed for the manufacture of atomic bombs .
plutonium for the manufacture of atomic bombs .
required for the manufacture of atomic bombs .
needed for the manufacture of atomic bombs . </s>
plutonium for the manufacture of atomic bombs . </s>
required for the manufacture of atomic bombs . </s>

Figure 9.10: Example strings generated using monolingual coverage acceptor $\mathcal{C}_n$.

$\mathcal{H}_{r+1}$. The concatenation of unweighted acceptors

$$\mathcal{X}_r = \mathcal{L}_r \otimes \mathcal{H}_r \otimes \mathcal{R}_r \tag{9.13}$$

represents the partial translation hypotheses in $\mathcal{H}_r$ padded with $n-1$ words of left and right context from the neighbouring high confidence regions. Composing $\mathcal{X}_r \circ \mathcal{C}_n$ assigns to each partial hypothesis a cost equal to the number of times it was necessary to back off to lower order n-grams while reading each string in $\mathcal{X}_r$. A partial hypothesis with a cost of 0 did not back off at all and contains only n-grams of the longest possible order.

In the following experiments, the unweighted composition $\mathcal{X}_r \circ \mathcal{C}_n$ is applied in each sublattice region. If there are paths with cost zero, then only these are kept and all others are discarded. This procedure is introduced as a constraint on the hypothesis space which will be evaluated for improvement in fluency. Here the transformation function Ψ returns $\mathcal{H}_r$ as $\mathcal{X}_r \circ \mathcal{C}_n$ after pruning all paths with a cost greater than zero. If $\mathcal{X}_r \circ \mathcal{C}_n$ has no zero cost paths, the transformation function Ψ returns $\mathcal{H}_r$ as it is found, since there is not enough coverage in the monolingual text to guide the selection of more fluent hypotheses. After applying monolingual coverage constraints in each region, the modified hypothesis space used for MBR search is formed by concatenation of regions using Equation (9.12).

Note that $\mathcal{C}_n$ can be viewed as a very simplistic natural language generation system. It generates strings by concatenating n-grams found in $\mathcal{S}$. It is not allowed to run 'open loop' in the following experiments, but is instead used to find the strings in $\mathcal{X}_r$ with good n-gram coverage. Figure 9.10 shows examples of some of the longer strings generated by running $\mathcal{C}_n$ in open loop mode for the set of n-grams in an unconstrained Arabic→English translation lattice. These partial hypothesis strings are generated by concatenation of overlapping n-grams found in the training data, without ever needing to back off to lower order n-grams. Strings generated in this way are seen to have a very high level of fluency.

9.6 Lattice Minimum Bayes-Risk Decoding Over Segmented Lattices

The effect of fluency constraints on lattice MBR decoding is evaluated in the context of the NIST Arabic→English machine translation task.[1] The set mt0205tune is formed from the odd numbered sentences of the MT02–MT05 testsets; the even numbered sentences form mt0205test. Performance on mt08nw (newswire) and mt08ng (newsgroup) data is also reported. The first-pass decoder, segmentation process, and lattice MBR procedures are all implemented using OpenFst (Allauzen et al., 2007).

First-pass translation is performed using HiFST (Iglesias et al., 2009b), a hierarchical phrase-based decoder. Word alignments are generated using MTTK (Deng and Byrne, 2008) over approximately 150M words of parallel text specified for the constrained NIST MT08 Arabic→English track. Hierarchical rules are extracted using the constraints of Chiang (2007) with the additional count and pattern filters described in Iglesias et al. (2009a). In decoding, a Shallow-1 grammar with a single level of rule nesting is used and no pruning is required during search (Iglesias et al., 2009b).

The first-pass language model is a modified Kneser-Ney (Kneser and Ney, 1995) 4-gram estimated over the English side of the parallel text and an 881M word subset of the English GigaWord 3^{rd} Edition (Graff et al., 2007). Prior to LMBR, the first-pass lattices are rescored with zero-cutoff stupid-backoff 5-gram language models (Brants et al., 2007) estimated over more than six billion words of English text. These are the language models used in the example of Figure 9.8. The factors $\theta_0, \ldots, \theta_4$ of the LMBR decoder are set as in Tromble et al. (2008) using unigram precision $p = 0.85$ and average recall ratio $r = 0.74$.

The effect of performing lattice MBR over the segmented hypothesis space is shown in Table 9.1. The individual hypothesis spaces $\mathcal{H}_r$ are constructed at various confidence thresholds as described in Section 9.3, with $\mathcal{H}$ formed via Equation (9.12); no coverage constraints are applied at this stage. At confidence threshold $\beta = 0.6$, it appears that constraining the search space to contain n-grams with posterior probability greater than or equal to β leads to little degradation in LMBR performance under BLEU. This shows that the lattice segmentation process works as intended.

9.6.1 Decoding with Coverage Constraints

The effect on the BLEU score of applying monolingual coverage constraints is now investigated. The acceptors $\mathcal{C}_n$ are constructed as described in Section 9.5 with $\mathcal{S}$ consisting of all n-grams (orders $n = 1 \ldots 5$) in the English GigaWord Third Edition text collection (approximately 3.6 billion words). At $\beta = 0.6$ there are 181 sentences in mt08nw with sublattices $\mathcal{H}_r$ that can be completely spanned by n-grams of the maximum possible order from $\mathcal{S}$, i.e. for which $\mathcal{X}_r \circ \mathcal{C}_n$ contains paths with zero cost; these regions are filtered as described. LMBR over the concatenation of these coverage-constrained sublattices is denoted LMBR+CC. On mt08nw the BLEU score for LMBR+CC is 52.0 which is +0.7 over ML decoding and only -0.2 BLEU below the unconstrained LMBR decoding. These results show that constraining partial hypotheses in low confidence regions to have no backing off using n-grams from the GigaWord causes little change in the size of the gain obtained through LMBR decoding.

[1] http://www.itl.nist.gov/iad/mig/tests/mt

		mt0205tune	mt0205test	mt08nw	mt08ng
ML		54.2	53.8	51.3	36.3
β	0.0	54.2	53.8	51.3	36.3
	0.2	54.3	53.8	51.3	36.3
	0.4	54.6	54.2	51.6	36.7
	0.6	54.9	54.4	52.1	36.6
	0.8	54.9	54.4	52.1	36.6
	1.0	54.9	54.4	52.2	36.7
LMBR		54.9	54.4	52.2	36.8

Table 9.1: Arabic→English maximum likelihood (ML) and lattice minimum Bayes-risk (LMBR) decoding BLEU scores over n-gram posterior probability thresholds $0 \leq \beta \leq 1$.

At this value of β, 116 of the 813 mt08nw sentences have a low confidence region (i) completely covered by 5-grams, and (ii) within which the ML hypothesis and the LMBR+CC hypothesis differ. It is these regions which will be inspected for improved fluency.

9.6.2 Reference Translation Coverage Statistics

The effectiveness of monolingual coverage constraints as a method for improving machine translation fluency depends crucially on the level of n-gram coverage in fluent monolingual data. Table 9.2 shows n-gram coverage statistics by order for the union of the four reference translations of the Arabic→English mt0205tune and mt0205test testsets. These coverage statistics are computed with respect to approximately 3.6 billion words of tokenized data in the English GigaWord Third Edition (Graff et al., 2007). An n-gram is considered covered if it occurs at least once in the monolingual training data.

Order	mt0205tune		mt0205test	
	n-grams	coverage (%)	n-grams	coverage (%)
1	10566	99.3	10449	99.3
2	77638	95.6	76642	95.8
3	150062	82.6	149070	82.9
4	189172	58.5	188672	58.7
5	206802	34.2	206760	34.4

Table 9.2: Total number of unique n-grams and GigaWord Third Edition coverage statistics by order for the union of the four reference translations of mt0205tune and mt0205test.

The table shows that there is good coverage of unigrams and bigrams, but the coverage falls off rapidly at higher orders. Only around 59% of 4-grams and 34% of 5-grams in the references are found in the fluent monolingual text collection.

9.7 Human Fluency Evaluation

Since it is difficult to reliably assess the fluency of MT output using automatic metrics such as BLEU and TER, 17 native speakers of English were asked to judge the fluency of sentence fragments from the Arabic→English mt08nw testset. 116 sentence fragments from the

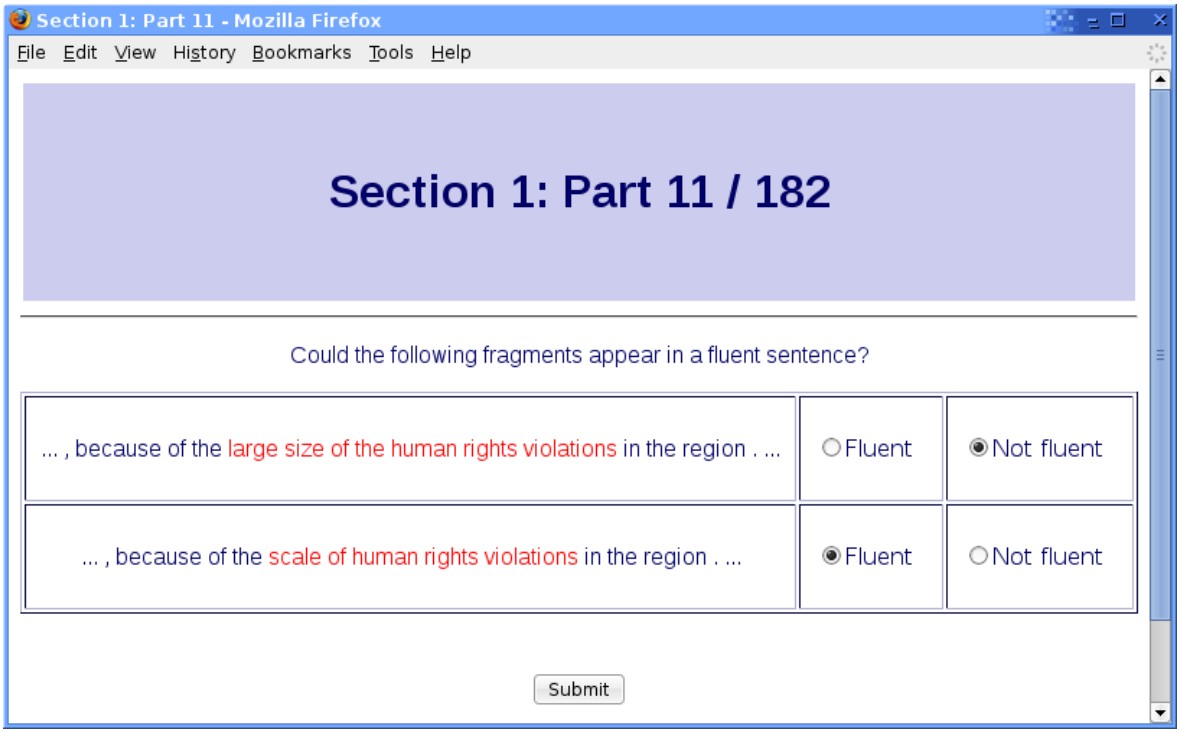


Figure 9.11: Human fluency evaluation web application.

maximum likelihood decoder (ML) and the lattice MBR decoder with monolingual coverage constraints (LMBR+CC) were compared. Each fragment consisted of the partial translation hypothesis from a low confidence sublattice region together with its left and right high confidence contexts (examples are given in Figure 9.12, with low confidence regions marked in italics). For each pair of sentence fragments, judges were asked: "Could the following fragments appear in a fluent sentence?". The evaluation was based on a modified version of the web application used for the Blizzard Challenge (Black and Tokuda, 2005). A screen-shot of the web application presented to the human judges is shown in Figure 9.11.

The results of the evaluation are shown in Table 9.3. Most of the time, the ML and LMBR+CC sentence fragments were both judged to be fluent; it often happened that they differed by only a single noun or verb substitution which did not affect fluency. In a small number of cases, both the ML and LMBR+CC fragments were judged to be disfluent. The most interesting cases are the 'off-diagonal' cases. When one system was judged to be fluent and the other was not, LMBR+CC was preferred more than twice as often as the ML baseline (26.9% to 9.7%). In other words, the application of hypothesis space constraints based on monolingual coverage was judged to have improved the fluency of partial hypotheses in low confidence regions more than twice as often as fluent hypotheses were made disfluent.

Some examples of improved fluency are shown in Figure 9.12. Although both the ML and unconstrained LMBR hypotheses could probably be said to satisfy translation adequacy, they lack the fluency of the LMBR+CC hypotheses. In the first of these examples, the LMBR+CC sentence fragment is made perfectly fluent by replacing the single word 'open' with the three word phrase 'opening of the'. The resulting 14 word sentence fragment is then completely covered by a series of 10 overlapping 5-grams in the monolingual text collection.

		LMBR+CC	
		Fluent	Not Fluent
ML	Fluent	1175 (59.6%)	192 (9.7%)
	Not Fluent	530 (26.9%)	75 (3.8%)

Table 9.3: Partial hypothesis fluency judgements by 17 native speakers of English.

Decoder	Partial Translation Hypothesis
ML	... view , especially with *the open chinese economy* to the world and ...
+LMBR	... view , especially with *the open chinese economy* to the world and ...
+LMBR+CC	... view , especially with *the opening of the chinese economy* to the world and ...
ML	... change the position of *iran nuclear . </s>*
+LMBR	... change the position of *the iranian nuclear . </s>*
+LMBR+CC	... change the position of *iran . </s>*
ML	... revision of the constitution *of the japanese public* , which dates back ...
+LMBR	... revision of the constitution *of the japanese public* , which dates back ...
+LMBR+CC	... revision of the constitution *of japan* , which dates back ...
ML	<s> it should be *remembered* the benefits of the ...
+LMBR	<s> it should be *recalled* the benefits of the ...
+LMBR+CC	<s> it should be *a reminder of* the benefits of the ...

Figure 9.12: Improved fluency through the application of monolingual coverage constraints to the hypothesis space in lattice MBR decoding of the Arabic→English mt08nw testset.

The availability of such higher-order n-gram coverage is the basis for constraints on the MBR search space that lead to improved fluency.

The second example shows one possible problem with hypothesis space constraints based on coverage: filtering sublattice regions to retain only hypotheses completely covered by high order n-grams may lead to the deletion of content. Here, the word 'nuclear', which is present in both the ML and unconstrained LMBR translation, has been deleted from the LMBR+CC hypothesis. It was deleted since its inclusion in these partial translation hypotheses is not fluent and maximum order n-grams cannot be found to completely cover the fragments. From a fluency perspective, the word should be deleted. From an adequacy perspective, however, the deleted word might represent a loss of information content. Any approach to improving MT fluency must be careful not to adversely impact translation adequacy.

9.8 Summary and Conclusions

This chapter has described a novel general framework for improving the fluency of machine translation output. By decoupling the hypothesis space from the evidence space, there is great potential for flexibility in lattice minimum Bayes-risk search.

The approach to improving fluency described in this chapter is motivated by the analysis in Section 9.2, where it was shown that high path posterior probability n-grams in the ML translation hypothesis can be used to guide the segmentation of a lattice into regions of high and low confidence. Lattice MBR decoding can be performed over such segmented lattices with little or no degradation in performance relative to unconstrained LMBR.

The segmentation of the lattice into alternating regions of high and low confidence considerably simplifies the process of refining the hypothesis space since low confidence regions

can be refined in the context of the high confidence strings that surround them. This can be done independently before reassembling the hypothesis space as the concatenation of refined regions. The use of general purpose weighted finite-state transducer methods allows for efficient identification of high and low confidence regions. The lattice segmentation process, implementation of the transformation function, and hypothesis space reconstitution procedures are also greatly simplified. This form of lattice segmentation facilitates the targeted application of techniques intended to address specific deficiencies in SMT.

As one example of the use of this framework, hypothesis space constraints were applied to low confidence regions based on maximum order n-gram coverage in a large monolingual text collection. An evaluation of the constrained regions by native speakers showed improved translation fluency without a significant degradation in BLEU score relative to LMBR decoding in the unconstrained hypothesis space.

The effectiveness of this particular approach to improving the fluency of the MBR search space is plainly limited by the coverage of low confidence sublattice regions using fluent monolingual text. This is expected to improve with larger text collections such as the vast library of fluent text contained in the Google Books project[1], or in tightly focused scenarios where in-domain text is less diverse.

However, machine translation fluency will be best improved by integrating more sophisticated natural language generation systems. NLG systems capable of generating sentence fragments in context can be incorporated directly into this framework. This framework could also be used to improve the fluency of automatic speech recognition (Huang et al., 2001), optical character recognition (Mori et al., 1999), and other language processing tasks in which the objective is to produce fluent output.

[1] http://books.google.com

Chapter 10
Conclusions

Lattice rescoring methods offer great potential for improving the quality of statistical machine translation. Complex models and processes that are difficult or impossible to integrate in first-pass translation decoding can be efficiently applied to large lattices of the most likely translation hypotheses. This form of multi-pass translation is of increasing interest to the statistical machine translation community (Rosti et al., 2007a,b; Tromble et al., 2008; Kumar et al., 2009; Li et al., 2009; DeNero et al., 2009; Allauzen et al., 2010).

The original contributions are reviewed in Section 10.1; the publications and presentations resulting from the research described in this thesis are listed in Section 10.2. Section 10.3 discusses possible extensions and suggested areas for future research, building on the core ideas developed in the later chapters of this thesis.

10.1 Review of Work

This thesis developed an inventory of robust and effective lattice rescoring methods for large-scale statistical machine translation. Efficient realisations of these rescoring methods in terms of general purpose weighted finite state transducer operations and algorithms were demonstrated to lead to significant improvements in the quality of state-of-the-art statistical machine translation systems. The rescoring methods described in this work have been used extensively for translation research at CUED, and contributed to significant gains in highly-ranked submissions to the NIST and WMT evaluations of SMT quality. This section reviews the original contributions of this thesis.

10.1.1 Large Language Model Rescoring

In order to establish a high quality baseline for investigations into more sophisticated lattice rescoring methods, Chapter 5 presented a detailed empirical investigation of SMT lattice rescoring with high-order n-gram language models estimated over multi-billion word corpora. 5-gram and 6-gram zero-cutoff language models estimated over more than 10 billion words of English monolingual training data were demonstrated to lead to substantial improvements in the quality of Arabic→English and Chinese→English translation. French→English and Spanish→English lattice rescoring experiments were also reported.

The simple dependency structure of stupid-backoff smoothing (Brants et al., 2007) allows for an efficient, low-memory streaming algorithm to be used to filter counts for relevancy; this allows large LMs to be applied in lattice rescoring without a distributed client-server architecture. An efficient rescoring framework based on encoding only the subset of model parameters required to rescore each lattice as a weighted finite-state acceptor was described. Limitations of data sparsity and poor coverage of higher-order n-grams in lattice hypotheses mean that the 6-gram models are currently no better than the 5-gram models for SMT lattice rescoring.

10.1.2 Phrasal Segmentation Models

Phrasal segmentation models were proposed and developed as a simple but effective stochastic model of the segmentation process in phrase-based statistical machine translation (Blackwood et al., 2008b). Chapter 6 described how the parameters of a phrasal segmentation model can be estimated from naturally occurring phrase sequence examples in a large monolingual training corpus. First-order phrasal segmentation model rescoring of Arabic→English lattices was shown to result in significant complementary gains in BLEU score with respect to large 5-gram and 6-gram zero-cutoff language models. Phrasal segmentation model rescoring improves phrase-based SMT quality by exploiting the same abundantly available monolingual data that is normally used only for building word-based language models.

Although there is no explicit model of the segmentation process in hierarchical phrase-based SMT, phrases extracted from contiguous strings of terminals in the rules of the grammar allow phrasal segmentation models to be used to rescore lattices produced by a hierarchical decoder. This approach, however, yielded only moderate gains when applied to Arabic→English and Chinese→English lattice rescoring. As described in the conclusions to Chapter 6, a parameter estimation procedure that correctly accounts for the hierarchical nature of phrasal substitutions and reordering may be required before phrasal segmentation models can be effectively applied to the output of a hierarchical phrase-based decoder.

10.1.3 Efficient Lattice Minimum Bayes-Risk Decoding

Chapter 7 proposed a fast and exact formulation of linearised lattice minimum Bayes-risk decoding based on efficient path counting transducers (Blackwood and Byrne, 2010). Mapping word sequences to sequences of n-grams considerably simplifies the extraction of the higher-order statistics. Weighted path counting transducers allow for all n-gram path posterior probabilities of a given order to be computed in a single weighted composition; analysing decoding times showed this approach to be more than twice as fast as the sequential method of computing n-gram posterior probabilities one-by-one in series (Tromble et al., 2008).

Comprehensive MBR decoding experiments showed the new decoder to perform well in rescoring of large Arabic→English and Chinese→English SMT lattices, leading to large gains in BLEU score over the maximum likelihood translations. The importance of using as large an evidence space as possible was demonstrated by comparing decoding performance over lattices and k-best lists. An analysis of k-best list evidence space sizes showed that, for many longer sentences, a surprisingly small proportion of the total lattice probability mass is represented by large lists of the top 20000 hypotheses.

10.1.4 Multiple Lattice Minimum Bayes-Risk Combination

Chapter 8 introduced an efficient multiple-lattice generalisation of the lattice MBR decoder described in Chapter 7. Multiple lattice MBR decoding allows the full evidence space of each individual lattice to contribute to the calculation of the expected risk. This is a significant advantage over alternative combination techniques which are often limited to relatively shallow k-best lists for efficiency reasons. Multiple evidence spaces provide greater robustness since aspects of translation that are poorly handled in one set of lattices may be compensated for by better handling in another set of lattices.

Multiple-lattice MBR decoding was evaluated on two separate combination tasks. Multi-input translation of alternative decompositions of the source language sentence was shown to be effective in two-way and three-way combination of lattices generated from different Arabic morphological analyses (de Gispert et al., 2010). Similar gains were observed through the combination of lattices generated from alternative Chinese word segmentations. Multi-source translation of French→English and Spanish→English lattices was shown to lead to very large gains in BLEU score with respect to the maximum likelihood translations of the best of the individual systems.

10.1.5 Posterior-Based Lattice Segmentation

The analysis of n-gram precisions in Chapter 9 showed that high probability n-grams in the maximum likelihood translation are more likely to be found in the human reference translations. These results motivated the use of n-gram path posterior probabilities to guide the segmentation of a lattice into regions of high and low confidence. Efficient segmentation procedures were described in terms of weighted finite state transducers. MBR decoding over the segmented lattice was shown to result in little or no degradation in the BLEU score compared to MBR decoding over the unsegmented lattice.

Lattice segmentation considerably simplifies the process of refining the MBR hypothesis space since low confidence regions can be refined in the context of their high confidence neighbours. This can be done independently in each region of low confidence before reassembling the refined regions. Lattice segmentation is proposed as a general technique for facilitating the targeted application of specific post-processing methods intended to address deficiencies in translation.

10.1.6 Hypothesis Space Constraints

Chapter 9 proposed a novel framework for improving the fluency of statistical machine translation based on a separation of the hypothesis space and evidence space in a minimum Bayes-risk decoder. Segmenting first-pass translation lattices using an n-gram path posterior probability

confidence measure allows the hypothesis space to be refined by enforcing hypothesis space constraints in the low confidence regions. Constraints based on high-order n-gram coverage in a large target-language monolingual text collection were demonstrated to lead to improved machine translation fluency. This framework potentially allows for robust integration of natural language generation in statistical machine translation, an area for future work that is discussed in more detail in Section 10.3.

10.2 Publications and Presentations

The research described in this thesis has led to the following publications and presentations:

1. G. Blackwood and W. Byrne. Efficient path counting transducers for minimum Bayes-risk decoding of statistical machine translation lattices. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, (accepted for publication) 2010.

2. A. de Gispert, G. Iglesias, G. Blackwood, E. R. Banga, and W. Byrne. Hierarchical phrase-based translation with weighted finite state transducers and shallow-n grammars. In Computational Linguistics. Association for Computational Linguistics, (accepted for publication) 2010.

3. A. de Gispert, G. Iglesias, G. Blackwood, J. Brunning, and W. Byrne. The CUED NIST 2009 Arabic-English SMT System. NIST Open Machine Translation 2009 Evaluation (MT09) Workshop, August 2009.

4. M. Kurimo, S. Virpioja, V. T. Turunen, G. Blackwood, and W. Byrne. Overview and results of morpho challenge 2009. In Multilingual Information Access Evaluation Vol. I-II, 10th Workshop of the Cross-Language Evaluation Forum, CLEF 2009, Corfu, Greece. Springer Lecture Notes in Computer Science, 2009.

5. G. Blackwood, A. de Gispert, J. Brunning, and W. Byrne. Large-scale statistical machine translation with weighted finite state transducers. In Proceedings of FSMNLP 2008: Finite-State Methods and Natural Language Processing, Ispra, Lago Maggiore, Italy, September 2008.

6. G. Blackwood, A. de Gispert, and W. Byrne. Phrasal segmentation models for statistical machine translation. In Proceedings of the 22nd International Conference on Computational Linguistics, Manchester, UK, August 2008.

7. G. Blackwood, A. de Gispert, J. Brunning, and W. Byrne. European language translation with weighted finite state transducers: The CUED MT system for the 2008 ACL workshop on statistical machine translation. In Proceedings of the ACL 2008 Third Workshop on Statistical Machine Translation, June 2008.

8. A. de Gispert, G. Blackwood, J. Brunning, and W. Byrne. The CUED NIST 2008 Arabic-English SMT System. Presented at NIST MT Workshop, March 2008.

10.3 Future Work

The minimum Bayes-risk decoder described in Chapter 9 proposed decoupling the hypothesis space from the evidence space so as to allow for greater flexibility in search. The fluency of statistical machine translation was improved within this framework by applying hypothesis space constraints to the low confidence regions of first-pass translation lattices. These constraints discard partial hypotheses whenever fluent alternatives can be found, where the degree of fluency was determined by considering coverage of high-order n-grams in monolingual data. Coverage constraints are one simple way of improving SMT fluency but depend critically on coverage of partial hypothesis n-grams in monolingual training data.

More sophisticated methods for refining the MBR hypothesis space include re-decoding low-confidence regions with linguistically motivated features, hypothesis combination strategies (Rosti et al., 2007a,b), and long-span language models estimated over massive monolingual text collections such as the Google Books[1] project. Hypothesis space constraints derived from statistical parsing (Charniak, 1997; Collins, 1999) of partial hypotheses may also lead to higher levels of SMT quality and fluency.

An alternative method for improving SMT fluency is to augment or replace the MBR hypothesis space with new hypotheses produced by a state-of-the-art natural language generation system. NLG systems capable of generating sentence fragments in context can be incorporated directly into the MBR decoding framework of Chapter 9. Decoding in a generated hypothesis space searches for translations in the space of fluent sentences close to the hypotheses of the baseline system, as determined by the MBR loss function $L(E, E')$.

Following on from the discussion of the limitations of the source-channel model of SMT in the introduction to Chapter 9, if the MBR hypothesis space $\mathcal{H}$ contains a generated hypothesis $\bar{E}$ for which $P(F|\bar{E}) = 0$, it is still possible for $\bar{E}$ to be produced as a translation, since it can be 'voted for' by nearby hypotheses produced by the underlying system. The hypothesis space can therefore be augmented by new hypotheses without compromising the robustness of statistical machine translation.

As evidence of the need for a richer hypothesis space, the reachability problem (Chapter 9, Section 9.1) in SMT is briefly analysed. Table 10.1 shows the proportion of NIST Arabic→English testset sentences that can be successfully aligned to any one of the available human reference translations using our high quality baseline hierarchical decoder and a powerful grammar (Iglesias et al., 2009b). The low levels of reachability suggest that without some form of natural language generation or other augmentation of the hypothesis space it will be difficult to achieve high levels of translation quality and fluency. Other posterior-based lattice rescoring methods such as the approaches of Kumar et al. (2009) and Li et al. (2009) will also benefit from NLG whenever the baseline is incapable of generating the reference.

[1] http://books.google.com

Testset	Sentences	Reachability
mt0205tune	2075	15%
mt0205test	2040	14%
mt08nw	813	11%
mt08ng	547	9%

Table 10.1: NIST Arabic→English reference translation reachability.

References

Adams, D. (1979). *The Hitchhiker's Guide To The Galaxy*. Pan Macmillan.

Allauzen, C., Kumar, S., Macherey, W., Mohri, M., and Riley, M. (2010). Expected sequence similarity maximization. In *Human Language Technologies 2010: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, California.

Allauzen, C., Mohri, M., and Roark, B. (2003). Generalized algorithms for constructing statistical language models. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pages 557–564.

Allauzen, C., Riley, M., Schalkwyk, J., Skut, W., and Mohri, M. (2007). OpenFst: A general and efficient weighted finite-state transducer library. In *Proceedings of the Ninth International Conference on Implementation and Application of Automata (CIAA 2007)*, pages 11–23. Springer Lecture Notes in Computer Science.

Bacchiani, M., Roark, B., and Saraclar, M. (2004). Language model adaptation with map estimation and the perceptron algorithm. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 21–24, Boston, Massachusetts, USA. Association for Computational Linguistics.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (1990). A maximum likelihood approach to continuous speech recognition. *Readings in Speech Recognition*, pages 308–319.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Association of Computational Linguistics Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Bangalore, S., Bordel, G., and Riccardi, G. (2001). Computing consensus translation from multiple machine translation systems. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.

Bell, T., Cleary, J., and Witten, I. (1990). *Text Compression*. Prentice Hall.

Bellegarda, J. R. (2004). Statistical language model adaptation: review and perspectives. *Speech Communication*, 42:93–108.

Bender, O., Matusov, E., Hahn, S., Hasan, S., Khadivi, S., and Ney, H. (2007). The RWTH Arabic-to-English spoken language translation system. In *Proceedings of the 2007 Automatic Speech Understanding Workshop*, pages 396–401.

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

Black, A. and Tokuda, K. (2005). The Blizzard challenge 2005: Evaluating corpus-based speech synthesis on common datasets. In *Proceedings of Interspeech 2005*.

Blackwood, G. and Byrne, W. (2010). Efficient path counting transducers for minimum Bayes-risk decoding of statistical machine translation lattices (to appear). In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Blackwood, G., de Gispert, A., Brunning, J., and Byrne, W. (2008a). European language translation with weighted finite state transducers: The CUED MT system for the 2008 ACL workshop on SMT. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 131–134, Columbus, Ohio. Association for Computational Linguistics.

Blackwood, G., de Gispert, A., Brunning, J., and Byrne, W. (2009). Large-scale statistical machine translation with weighted finite state transducers. In *Post Proceedings of the 7th International Workshop on Finite-State Methods and Natural Language Processing, FSMNLP 2008*, pages 39–49, Amsterdam, The Netherlands. IOS Press.

Blackwood, G., de Gispert, A., and Byrne, W. (2008b). Phrasal segmentation models for statistical machine translation. In *Coling 2008: Companion volume: Posters and Demonstrations*, pages 17–20, Manchester, UK. Coling 2008 Organizing Committee.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 858–867.

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

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

Brunning, J., de Gispert, A., and Byrne, W. (2009). Context-dependent alignment models for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 110–118, Boulder, Colorado. Association for Computational Linguistics.

Burch, C. C., Osborne, M., and Koehn, P. (2006). Re-evaluating the role of BLEU in machine translation research. In *Proceedings of European Chapter of the Association for Computational Linguistics*, volume 2006, pages 249–256.

Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece. Association for Computational Linguistics.

Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluating the role of BLEU in machine translation research. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256. Association for Computational Linguistics.

Chang, P.-C., Galley, M., and Manning, C. D. (2008). Optimizing chinese word segmentation for machine translation performance. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 224–232, Morristown, NJ, USA. Association for Computational Linguistics.

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

Charniak, E. (1997). Statistical techniques for natural language parsing. *AI Magazine*, 18:33–44.

Chase, L. L. (1997). Error-responsive feedback mechanisms for speech recognizers, Ph.D. Thesis, School of Computer Science, Carnegie Mellon University.

Chen, S. F. and Goodman, J. (1998). An empirical study of smoothing techniques for language modeling. In *Technical Report TR-10-98*, Harvard University. Computer Science Group.

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

Chiang, D. (2007). Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Collins, M. (1999). Head-driven statistical models for natural language parsing, Ph.D. Thesis, University of Pennsylvania.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to Algorithms – Second Edition*. The MIT Press.

Darroch, J. and Ratcliff, D. (1972). Generalised iterative scaling for log-linear models. In *Annals of Mathematical Statistics*, pages 43:1470–1480.

de Gispert, A., Iglesias, G., Blackwood, G., Banga, E. R., and Byrne, W. (2010). Hierarchical phrase-based translation with weighted finite state transducers and shallow-n grammars (accepted for publication April 2010). In *Computational Linguistics*. Association for Computational Linguistics.

de Gispert, A., Virpioja, S., Kurimo, M., and Byrne, W. (2009). Minimum Bayes risk combination of translation hypotheses from alternative morphological decompositions. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 73–76, Boulder, Colorado. Association for Computational Linguistics.

Deligne, S. and Bimbot, F. (1995). Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams. In *International Conference on Acoustics, Speech, and Signal Processing, 1995. ICASSP-95*, pages 169–172, Detroit, MI.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

DeNero, J., Chiang, D., and Knight, K. (2009). Fast consensus decoding over translation forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 567–575, Suntec, Singapore. Association for Computational Linguistics.

Deng, Y. and Byrne, W. (2008). HMM word and phrase alignment for statistical machine translation. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(3):494–507.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification – Second Edition*. Wiley-Interscience.

Ehling, N., Zens, R., and Ney, H. (2007). Minimum bayes risk decoding for BLEU. In *Proceedings of the 45th Annual Meeting of the ociation for Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 101–104, Morristown, NJ, USA. Association for Computational Linguistics.

Emami, A., Papineni, K., and Sorensen, J. (2007). Large-scale distributed language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing 2007*.

Fiscus, J. G. (1997). A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 347–352.

Gale, W. and Church, K. (1994). What's wrong with adding one? In *Corpus-Based Research into Languge*, pages 189–200.

Germann, U. (2001). Aligned hansards of the 36th parliament of canada, Natural Language Group of the USC Information Sciences Institute.

Goel, V. and Byrne, W. J. (2000). Minimum bayes-risk automatic speech recognition. *Computer Speech and Language*, 14(2):115–135.

Goel, V., Kumar, S., and Byrne, W. (2004). Segmental minimum Bayes-risk decoding for automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, 12:234–249.

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:327–264.

Graff, D. (1994). UN parallel text (complete), Linguistic Data Consortium.

Graff, D., Kong, J., Chen, K., and Maeda, K. (2007). English Gigaword Third Edition, Linguistic Data Consortium.

Habash, N. and Rambow, O. (2005). Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the Association for Computational Linguistics*, pages 573–580.

Huang, X., Acero, A., and Hon, H.-W. (2001). *Spoken Language Processing, A Guide to Theory, Algorithm and System Development*. Prentice Hall PTR.

Iglesias, G., de Gispert, A., Banga, E. R., and Byrne, W. (2009a). Rule filtering by pattern for efficient hierarchical translation. In *Proceedings of the 12th Conference of the European Chapter of the Association of Computational Linguistics*, pages 380–388, Athens, Greece. Association for Computational Linguistics.

Iglesias, G., de Gispert, A., R. Banga, E., and Byrne, W. (2009b). Hierarchical phrase-based translation with weighted finite state transducers. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 433–441, Boulder, Colorado. Association for Computational Linguistics.

Jelinek, F. (1998). *Statistical Methods for Speech Recognition*. The MIT Press.

Jurafsky, D. and Martin, J. H. (2008). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Second Edition)*. Prentice Hall.

Katz, S. M. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, pages 400–401.

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *International Conference on Acoustics, Speech, and Signal Processing, 1995. ICASSP-95*, volume 1, pages 181–184.

Knight, K. (2007a). Automatic language translation generation help needs badly. In *MT Summit XI Workshop on Using Corpora for NLG: Language Generation and Machine Translation, Keynote Address*.

Knight, K. (2007b). Capturing practical natural language transformations. *Machine Translation*, 21(2).

Knight, K. and Graehl, J. (2005). An overview of probabilistic tree transducers for natural language processing. In *Proceedings of the Conference on Intelligent Text Processing and Computational Linguistics (CICLING)*.

Koehn, P. (2004). Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *The Association for Machine Translation in the Americas*.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *Machine Translation Summit 2005*.

Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press.

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

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference for Computational Linguistics on Human Language Technology*, pages 48–54, Morristown, NJ, USA.

Kumar, S. and Byrne, W. (2003). A weighted finite state transducer implementation of the alignment template model for statistical machine translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 63–70, Morristown, NJ, USA. Association for Computational Linguistics.

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

Kumar, S. and Byrne, W. (2005). Local phrase reordering models for statistical machine translation. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 161–168.

Kumar, S., Deng, Y., and Byrne, W. (2006). A weighted finite state transducer translation template model for statistical machine translation. *Natural Language Engineering*, 12(1):35–75.

Kumar, S., Macherey, W., Dyer, C., and Och, F. (2009). Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 163–171, Morristown, NJ, USA. Association for Computational Linguistics.

Kuo, H.-K. J. and Reichl, W. (1999). Phrase-based language models for speech recognition. In *Sixth European Conference on Speech Communication and Technology*, pages 1595–1598.

Kurimo, M., Virpioja, S., Turunen, V. T., Blackwood, G. W., and Byrne, W. (2009). Overview and results of morpho challenge 2009. In *Multilingual Information Access Evaluation Vol. I-II, 10th Workshop of the Cross-Language Evaluation Forum, CLEF 2009*, Corfu, Greece. Springer Lecture Notes in Computer Science.

Lari, K. and Young, S. J. (1990). The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

Lavie, A. and Agarwal, A. (2007). METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of Workshop on Statistical Machine Translation at the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics.

Lavie, A. and Denkowski, M. J. (2009). The METEOR metric for automatic evaluation of machine translation. *Machine Translation Journal*.

Lewis, II, P. M. and Stearns, R. E. (1968). Syntax-directed transduction. *Journal of the ACM*, 15(3):465–488.

Li, Z., Eisner, J., and Khudanpur, S. (2009). Variational decoding for statistical machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 593–601, Morristown, NJ, USA. Association for Computational Linguistics.

Ma, X. and Cieri, C. (2006). Corpus support for machine translation at LDC. In *Proceedings of LREC 2006: Fifth International Conference on Language Resources and Evaluation*.

Macherey, W. and Och, F. J. (2007). An empirical study on computing consensus translations from multiple machine translation systems. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 986–995.

Manning, C. D. and Schutze, H. (1999). *Foundations of Statistical Natural Language Processing*. The MIT Press.

Mathias, L. and Byrne, W. (2006). Statistical phrase-based speech translation. In *2006 IEEE International Conference on Acoustics, Speech and Signal Processing*.

Matusov, E., Ueffing, N., and Ney, H. (2006). Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment. In *Proceedings of the European Association for Computational Linguistics*.

Mohri, M. (1997). Finite-state transducers in language and speech processing. In *Computational Linguistics*, volume 23, pages 269–311.

Mohri, M. (2002). Semiring frameworks and algorithms for shortest-distance problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321–350.

Mohri, M., Pereira, F., and Riley, M. (2000). The design principles of a weighted finite-state transducer library. *Theoretical Computer Science*, 231(1):17–32.

Mohri, M., Pereira, F., and Riley, M. (2008). Speech recognition with weighted finite-state transducers. *Handbook on Speech Processing and Speech Communication*.

Mori, S., Nishida, H., and Yamada, H. (1999). *Optical Character Recognition*. Wiley-Interscience.

Oberlander, J. and Brew, C. (2000). Stochastic text generation. In *Philosophical Transactions of the Royal Society*, volume 358, pages 1373–1387.

Och, F., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., Jain, V., Jin, Z., and Radev, D. (2004). A smorgasbord of features for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Och, F. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302.

Och, F. J. (2002). Statistical machine translation: From single word models to alignment templates, Ph.D. Thesis, RWTH Aachen.

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

Och, F. J. and Ney, H. (2001). Statistical multi-source translation. In *Machine Translation Summit 2001*, pages 253–258.

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

Och, F. J. and Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

Papineni, K., Roukos, S., and Ward, R. (1998). Maximum likelihood and discriminative training of direct translation models. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pages 189–192.

Papineni, K., Roukos, S., Ward, T., Henderson, J., and Reeder, F. (2002a). Corpus-based comprehensive and diagnostic MT evaluation: initial Arabic, Chinese, French, and Spanish results. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 132–137, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

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

Parker, R., Graff, D., Kong, J., Chen, K., and Maeda, K. (2009). English Gigaword Fourth Edition, Linguistic Data Consortium.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2002). *Numerical Recipes in C++*. Cambridge University Press.

Ries, K., Bu, F. D., and Waibel, A. (1996). Class phrase models for language modeling. In *Proceedings of the 4th International Conference on Spoken Language Processing*.

Riley, M., Allauzen, C., and Jansche, M. (2009). OpenFst: An Open-Source, Weighted Finite-State Transducer Library and its Applications to Speech and Language. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Tutorial Abstracts*, pages 9–10, Boulder, Colorado. Association for Computational Linguistics.

Rosti, A.-V. I., Ayan, N. F., Xiang, B., Matsoukas, S., Schwartz, R., and Dorr, B. J. (2007a). Combining outputs from multiple machine translation systems. In *Proceedings of the North American Chapter of the Association for Computational Linguistics Human Language Technologies*, pages 228–235, Rochester, NY. Association for Computational Linguistics.

Rosti, A.-V. I., Matsoukas, S., and Schwartz, R. (2007b). Improved word-level system combination for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 312–319, Prague, Czech Republic. Association for Computational Linguistics.

Schroeder, J., Cohn, T., and Koehn, P. (2009). Word lattices for multi-source translation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 719–727, Athens, Greece. Association for Computational Linguistics.

Schwartz, L. (2008). Multi-source translation methods. In *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, pages 279–288. AMTA.

Sim, K. C., Byrne, W. J., Gales, M. J. F., Sahbi, H., and Woodland, P. C. (2007). Consensus network decoding for statistical machine translation. In *IEEE Conference on Acoustics, Speech and Signal Processing*.

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

Snover, M., Madnani, N., Dorr, B., and Schwartz, R. (2009). Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 259–268, Athens, Greece. Association for Computational Linguistics.

Stolcke, A. (1998). Entropy-based pruning of backoff language models. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 270–274.

Stolcke, A. (2002). SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904.

Talbot, D. and Osborne, M. (2007). Smoothed bloom filter language models: Tera-scale LMs on the cheap. In *Proceedings of the Conference of the Association for Computational Linguistics*, pages 468–476. Association for Computational Linguistics.

Tillmann, C. (2004). A unigram orientation model for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 101–104, Morristown, NJ, USA. Association for Computational Linguistics.

Tromble, R., Kumar, S., Och, F., and Macherey, W. (2008). Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 620–629, Honolulu, Hawaii. Association for Computational Linguistics.

Ueffing, N. and Ney, H. (2005). Word-level confidence estimation for machine translation using phrase-based translation models. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 763–770, Morristown, NJ, USA. Association for Computational Linguistics.

Ueffing, N. and Ney, H. (2007). Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40.

Ueffing, N., Ueng, N., Och, F. J., and Ney, H. (2002). Generation of word graphs in statistical machine translation. In *2002 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Vilar, D., Leusch, G., Ney, H., and Banchs, R. (2007). Human evaluation of machine translation through binary system comparisons. In *Proceedings of the Second Workshop on Statistical Machine Translation*.

Zens, R. and Ney, H. (2006). N -gram posterior probabilities for statistical machine translation. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: Proceedings of the Workshop on Statistical Machine Translation*.

Zhang, Y. and Clark, S. (2007). Chinese segmentation using a word-based perceptron algorithm. In *Proceedings of the Conference of the Association for Computational Linguistics*, Prague, Czech Republic.

Zhang, Y., Hildebrand, A. S., and Vogel, S. (2006). Distributed language modeling for N-best list re-ranking. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 216–223, Morristown, NJ, USA. Association for Computational Linguistics.