LEVERAGING DIVERSE SOURCES IN STATISTICAL MACHINE TRANSLATION

by

Majid Razmara

B.Eng., Iran University of Science and Technology, 2005

M.Sc., Concordia University, 2008

a Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Summer 2013

APPROVAL

Name:	Majid Razmara
Degree:	Doctor of Philosophy
Title of Thesis:	Leveraging Diverse Sources in Statistical Machine Translation
Examining Committee:	Dr. Ramesh Krishnamurti, Professor Chair
	Dr. Anoop Sarkar, Associate Professor Computing Science, Simon Fraser University Senior Supervisor
	Dr. Fred Popowich, Professor Computing Science, Simon Fraser University Senior Supervisor
	Dr. Greg Mori, Associate Professor Computing Science, Simon Fraser University SFU Examiner
	Dr. Alon Lavie, Research Professor Computer Science, Carnegie Mellon University External Examiner
Date Approved:	July 29, 2013

Abstract

Statistical machine translation is often faced with the problem of having insufficient training data for many language pairs. In this thesis, several methods have been proposed to leverage other sources to enhance the quality of machine translation systems. Particularly, we propose approaches suitable in these four scenarios:

1. when an additional parallel corpus between the source and the target language is available (*ensemble decoding*);
2. when parallel corpora between the source language and a third language and between that language and the target language are available (*ensemble triangulation*);
3. when an abundant source language monolingual corpus is available (*graph propagation for paraphrasing out-of-vocabulary words*);
4. when no additional resource is available (*stacking*).

In the heart of these solutions lie two novel approaches: ensemble decoding and a graph propagation approach for paraphrasing out-of-vocabulary (oov) words.

Ensemble decoding combines a number of translation systems dynamically at the decoding step. We evaluate its performance on a domain adaptation setting where a model trained on a large parliamentary domain is adapted to the medical domain, we then translate sentences from the medical domain. Our experimental results show that ensemble decoding outperforms various strong baselines including mixture models, the current state-of-the-art for domain adaptation in machine translation.

We extend ensemble decoding to do triangulation on-the-fly when there exist parallel corpora between the source language and one or multiple pivot languages and between those and the target language. These triangulated systems are dynamically combined together and possibly to a direct source-target system. Experiments in 12 different language pairs show significant improvements over the baselines in terms of BLEU scores.

Ensemble decoding can also be used to apply *stacking* to statistical machine translation. Stacking is an ensemble learning approach that enhances the bias of the models. We show that stacking can consistently and significantly improve over the conventional SMT systems in two different language pairs and three different training sizes.

In addition to ensemble decoding, we propose a novel approach to mining translations for oov words using a monolingual corpus on the source-side language. We induce a lexicon by constructing a graph on the source language phrases using a monolingual text and employ a graph propagation technique in order to find translations for those phrases. Experimental results in two different settings, including a domain adaptation one, show that our graph propagation method significantly improves performance over two strong baselines under intrinsic and extrinsic evaluation metrics.

To My Love, Maryam!

Acknowledgements

I would like to express my heartfelt gratitude to my advisor, Dr. Anoop Sarkar, for his insightful guidance and warm encouragements. I appreciate all his contributions of time, ideas, and funding to make my experience productive and exciting. I would also like to thank my other advisor, Dr. Fred Popowich, for his support and encouragements. I am grateful of both of them not only because of their supervision, advice and enthusiasm, but also for their nice personalities. They made my PhD journey pleasant even during tough times.

I would also like to thank my examiners, Dr. Greg Mori and Dr. Alon Lavie. Greg is a wonderful mentor and is always open to help. Dr. Lavie has written many papers that I read during my PhD program and it is an honor to have him on the committee.

I learned a great deal of what I know in NLP from the students in the natlang lab at SFU. I would like to thank all of them for creating such a friendly, fun and encouraging environment for me in which many ideas were developed and refined: Reza Haffari, Baskaran Sankaran, Ann Clifton, Maryam Siahbani, Max Whitney, Ravikiran Vadlapudi, Milan Tofiloski, Marzieh Razavi and Rohit Dholakia. I am specially grateful to Reza Haffari for helpful discussions and his significant contribution to one of the ideas presented in this thesis.

Outside our lab, I would also like to thank George Foster as well as people at the 2012 Summer workshop of Johns Hopkins University. I was fortunate to know and work with Hal Daume, Alex Fraser, Marine Carpuat and Chris Quirk and learn many things in such a short period from them. I am also grateful of Dr. Leila Kosseim, my Master's supervisor, whose support and help during tough times in my Masters will never be forgotten.

I would not have contemplated this road if not for my parents, their love and inspiration. I am grateful for their faith in me and their invaluable support. I also thank my friends for

providing support and friendship that I needed during my PhD program.

Last but not least, I would like to thank my lovely wife, Maryam Sadeghi, who has been always my source of energy and love. This PhD would not be possible without her unconditional love and support. I would also like to thank her wonderful supervisor, Dr. Stella Atkins, whose help and friendship made our life a better one.

Contents

Approval **ii**

Abstract **iii**

Dedication **v**

Acknowledgements **vi**

Contents **viii**

List of Tables **xii**

List of Figures **xiv**

1 Introduction **1**

1.1 Machine Translation 1

1.1.1 Log-linear Models 3

1.2 SMT Combination Approaches 4

1.3 Domain Adaptation 5

1.4 Thesis Outline 6

1.5 Research Contributions 7

2 Ensemble Decoding **8**

2.1 Introduction 8

2.2 Baselines 9

2.2.1 Log-Linear Mixture 10

2.2.2 Linear Mixture 10

2.3 Ensemble Decoding . . . 11
2.3.1 Mixture Operations . . . 12
2.3.2 A Semiring Definition . . . 14
2.3.3 Normalization . . . 16
2.4 Experiments & Results . . . 17
2.4.1 Baselines . . . 17
2.4.2 Experimental Setup . . . 18
2.4.3 Results . . . 19
2.4.4 WMT experiments . . . 20
2.5 Related Work . . . 23
2.5.1 Domain Adaptation . . . 23
2.5.2 System Combination . . . 24
2.6 Conclusion . . . 26
2.7 Future Directions . . . 26
2.7.1 Global Normalization . . . 26
2.7.2 Domain Mixing Scenario . . . 27
2.7.3 Mixture Operation Characteristics . . . 27
2.7.4 Consensus Ensemble Decoding . . . 28

3 Ensemble Triangulation **32**
3.1 Introduction . . . 32
3.2 Related Work . . . 34
3.2.1 System Cascades . . . 35
3.2.2 Corpus Synthesis . . . 35
3.2.3 Phrase-Table Triangulation . . . 35
3.3 Baselines . . . 36
3.4 Our Approach . . . 38
3.4.1 Dynamic Triangulation . . . 38
3.4.2 Combining Triangulated Systems . . . 39
3.4.3 Tuning Component Weights . . . 39
3.5 Experiments & Results . . . 40
3.5.1 Experimental Setup . . . 40
3.5.2 Results . . . 41

3.5.3 Phrase table coverage 42
3.6 Conclusion and Future Work 45

4 Stacking 47
4.1 Introduction 47
4.2 Ensemble Learning Methods 48
4.3 Our Approach 50
4.4 Experiments & Results 50
4.4.1 Training on dev set 51
4.4.2 Training on train+dev 52
4.4.3 Partitioning Methods 53
4.4.4 Decoding Time Overhead 54
4.5 Related Work 57
4.6 Conclusion 58

5 Graph Propagation for Paraphrasing OOVs 59
5.1 Introduction 59
5.2 Collocational Lexicon Induction 61
5.2.1 Baseline System 61
5.2.2 Association Measures 62
5.2.3 Similarity Measures 64
5.3 Graph-based Lexicon Induction 64
5.3.1 Label Propagation 66
5.3.2 Efficient Graph Construction 67
5.4 Experiments & Results 68
5.4.1 Experimental Setup 68
5.4.2 Phrase-table Integration 70
5.4.3 Evaluation 71
5.4.4 Intrinsic Results 72
5.4.5 Extrinsic Results 76
5.5 Related work 77
5.6 Conclusion 79

6 Conclusion 80
6.1 Summary . . . 80
6.2 Future directions . . . 81
6.2.1 Future Directions for Ensemble Decoding . . . 81
6.2.2 Future Directions for Ensemble Triangulation . . . 84
6.2.3 Future Directions for Stacking . . . 84
6.2.4 Future Directions for Graph Propagation for Paraphrasing OOVs . . . 85
6.3 Integration into a Single Model . . . 85

Appendix A Kriya - A Hierarchical Phrase-based MT System 88
A.1 Introduction . . . 89
A.2 Related Works . . . 90
A.3 Synchronous Context-Free Grammar . . . 91
A.4 Kriya . . . 92
A.4.1 Kriya Grammar Extractor . . . 93
A.4.2 Kriya Decoder . . . 95
A.5 Experiments . . . 98
A.6 Future Directions . . . 101
A.7 Conclusion . . . 101

Bibliography 103

List of Tables

2.1 Training, dev and test sets for EMEA. 18

2.2 The results of various baselines implemented in a phrase-based (PBS) and a Hiero SMT on EMEA. 19

2.3 The results of ensemble decoding on EMEA for *fr* → *en* when using uniform weights, tuned weights and normalization heuristic. The tuned BLEU scores are averaged over three runs with multiple initial points with the standard deviations in brackets. 19

2.4 Applying ensemble decoding with different mixture operations on the Test-11 dataset. 22

2.5 French-English BLEU scores. 22

3.1 Number of tokens in each language pair in the training data. 40

3.2 Results of i) single-pivot triangulation; ii) baseline systems including direct systems and linear mixture of triangulated phrase-tables; iii) ensemble triangulation results based on different mixture operations. The mixture and ensemble methods are based on multi-pivot triangulation. These methods are built on 10k sentence-pair corpora. 46

4.1 Statistics of the training set for different systems and different language pairs. 51

4.2 Test set BLEU scores when applying stacking on the dev set only (using no specific training set). 51

4.3 Test set BLEU scores for *fr* → *en* when using 10k and 100k sentence training sets along with the dev set. 52

4.4 Test set BLEU scores for *es* → *en* when using 10k and 100k sentence training sets along with the dev set. 52

4.5 Time overhead of stacking 11 base models over the baseline. 56

5.1 Statistics of training sets in different domains. 70

5.2 number of oovs in dev and test sets for Europarl and EMEA systems. 70

5.3 Results of intrinsic evaluations (MRR and Recall) on Europarl, window size 4 and paraphrase length 2 73

5.4 Intrinsic results of different types of graphs when using unigram nodes on Europarl. 75

5.5 Results on using unigram or bigram nodes. 76

5.6 BLEU scores for different domains with or without using oov translations. . . 76

5.7 Two examples of oov translations found by our method. 77

A.1 Corpus Statistics - English-French use a 4-gram LM and other pairs use 5-gram LM. Chinese-English experiments use four references for tuning and testing. 98

A.2 Model sizes and BLEU Scores - Model sizes are in millions of rules. Bold face indicates best BLEU score for each language pair and italicized figures point to statistically significant improvements assuming significance level $\alpha = 0.1$. . 99

A.3 Shallow-1 decoding vs. Compact (1 NT) model - Bold face indicates BLEU scores comparable to the original Hiero model in Table A.2. Size of the compact 1 NT model as a % of original Hiero model is given within the brackets. 100

A.4 Kriya Decoding time (in secs/word) for Chinese-English translation 101

List of Figures

2.1 The cells in the CKY chart are populated using rules from all component models as well as their sub-spans' cells. This figure shows two systems with different language and translation models and weight vectors that are being combined in a decoder. 12

2.2 Examples illustrating how this method is able to use expertise of both out-of-domain and in-domain systems. 21

3.1 Number of oovs when translating directly from *fr* to *en* (solid lines), triangulating through *es*, *de* or *it* individually (dotted lines), and when combining multiple triangulation systems with the direct system. These oov numbers are based on a multi-language parallel test set and the models are built on small corpora (10k sentence pairs), which are not multi-parallel. 34

3.2 The average BLEU scores of the direct system, mixture models and *wmax* ensemble triangulation approach over all 12 language pairs. 42

3.3 Meteor score difference between mixture models and direct systems as well as the difference between ensemble decoding approach with *wmax* and the direct system. 42

3.4 Coverage for i) direct system; ii) combined triangulated system with three 3 languages; and iii) the combination of the triangulated phrase-tables and the direct one. The table shows the number of rules for each system and language pair after filtering based on the source side of the test set. 43

3.5 Example translation output of the two baselines: direct and mixture and our approach: ensemble on *fr* → *en*. 44

4.1 BLEU scores for all the base models and stacked models on the *fr* → *en* dev set with 20-fold cross validation. The horizontal line shows the mean of base models' scores. 53
4.2 Results when using sampling without or without replacement and cross validation when the tuning set is fixed or changing (our approach) on the *fr* → *en* 10k sentence-pair data. *Mean* refers to the mean of the BLEU scores of the base models in each partitioning method. 54
4.3 Time comparison between different mixture operations of ensemble decoding where 11 base models are combined. 55
4.4 Stacking time ratio with regard to the number of base models. 56

5.1 A tripartite graph between oov, labeled and unlabeled nodes. Translations propagate either directly from labeled nodes to oov nodes or indirectly via unlabeled nodes. 66
5.2 A portion of unigram graph constructed on the French side of Europarl. . . . 69
5.3 Effects of different window sizes and paraphrase length on the MRR of the dev set. 74
5.4 Effect of increasing the monolingual text size on MRR and Recall. 75

Chapter 1

Introduction

1.1 Machine Translation

Automatic text translation, commonly called *machine translation*, is one of the oldest yet unsolved problems of natural language processing and artificial intelligence. Hand-crafted rule-based machine translation systems (e.g. SYSTRAN) were first attempts to attack this problem. In the early 1990s, Brown et al. (1990) proposed statistical machine translation models based on finite state machines and since then for almost a decade those were the dominant approaches for machine translation. But string-based models were not able to capture many of the natural language properties. Phrase-based machine translation models came into play and to some degree fixed some of the inadequacies of word-based models, e.g. local word-reorderings. However, the need to use the structures of sentences led to introducing syntax-based models.

The translation process may be seen as decoding the meaning of the source text and re-encoding it into the target language (Koehn, 2010). There are a number of different approaches to machine translation, among which statistical machine translation (SMT) is the state-of-the-art.

In statistical machine translation, the most common approach for estimating $p(e|f)$ (i.e. the probability that a string e in the target language is the translation of a string f in the source language) is using the *noisy channel* model:

$$\begin{aligned}\hat{e} &= \operatorname*{argmax}_{e} p(\,e \,|\, f\,) \\ &= \operatorname*{argmax}_{e} \quad \underbrace{p(\,f \mid e\,)}_{\text{translation model}} \quad . \quad \underbrace{p(\,e\,)}_{\text{language model}}\end{aligned}$$

Using Bayes' rule, the problem is divided into two subproblems, one of which is responsible for generating an adequate translation, while the other one is responsible for the fluency of it in the target language. There are different approaches for modeling languages. But the most common one is using n-gram models (i.e. unigram, bigram, trigram, etc.). Similarly, there are different ways for modelling the translation:

Word-Based Models: The translation units in these models are words. They are first introduced by Brown et al. (1990); Brown et al. (1993) through IBM Models 1-5. Model 6 was also suggested later by Och (2003a). Another commonly used word-based model incorporates a Hidden Markov Model over word alignments, known as the HMM model (Vogel et al., 1996).

Phrase-Based Models: The translation units in phrase-based translation models are sequences of words (i.e. phrases) rather than single words. They were first introduced by Och et al. (1999) in the alignment template model and had variations in Marcu and Wong (2002); Koehn et al. (2003); Och and Ney (2004). Phrase-based models produce higher quality translations compared to word-based models.

Phrase-based models address two types of shortages in word-based models: local reordering and idiomatic expressions. However, they are unable to model long-distance reorderings. To model this linguistic phenomenon, models have been proposed that take into account the structure of the sentences as well. These models can be divided into two categories: syntax-based models and pseudo-syntax-based models.

Pseudo-Syntax-Based Models: These models take into account the hierarchical characteristics of natural languages. However, they do not explicitly take into account the syntax parse trees of sentences. Two well-known examples of this category are hierarchical phrase-based models (Chiang, 2005a; Chiang, 2007a) and bracketing-grammar-based models (Wu, 1995; Wu, 1997).

Syntax-Based Models: Syntax-based models incorporate the linguistic syntax of sentences in translation. This category of translation models can further be divided into two subcategories: synchronous-grammar-based models and tree-transducer-based models. Several synchronous grammars have been successfully used in machine translation: Synchronous context-free grammars (SCFG) (Zollmann and Venugopal, 2006), synchronous tree-substitution grammars (Eisner, 2003), synchronous tree-adjoining grammars (STAG) (Shieber and Schabes, 1990; DeNeefe and Knight, 2009) and generalized multi-text grammars (GMTG) (Melamed et al., 2004). Tree transducers, on the other hand, have been used in several syntax-based machine translation, such as Yamada and Knight (2001); Gildea (2003); Galley et al. (2004); Galley et al. (2006) and are rather new to machine translation.

1.1.1 Log-linear Models

In the state-of-the-art statistical machine translation systems, the log-linear framework superseded the noisy-channel models with the rise of phrase-based models. In log-linear models, different model components such as language model, translation model and reordering model are used as feature functions (ϕ_i) with appropriate weights (w_i).

$$\begin{aligned}\hat{e} &= \text{argmax } p(\bar{e} \mid \bar{f}) \\ &= \text{argmax } \frac{\exp\left(\sum_i w_i \phi_i(\bar{e}, \bar{f})\right)}{\sum\limits_{\bar{e'}} \exp\left(\sum_i w_i \phi_i(\bar{e'}, \bar{f})\right)} \\ &= \text{argmax } \exp\left(\sum_i w_i \phi_i(\bar{e}, \bar{f})\right)\end{aligned}$$

This framework allows an elegant integration of arbitrary features such as number of words or phrases generated on the target side in the SMT model. In addition, each feature function has its own weight that signifies its importance in the whole model. In the original noisy-channel models, there were only two models involved (e.g. translation model and language model) with equal contribution. Finding (sub)optimal values for these feature weights, i.e. *tuning*, generally leads to an improvement in translation quality compared to using the equal weights.

1.2 SMT Combination Approaches

The state-of-the-art SMT paradigms for machine translation, namely phrase-based, hierarchical phrase-based and syntax-based systems encode different levels of syntactic information and have their own advantages and disadvantages. For example, syntax-based systems are able to capture most linguistic phenomena while the decoding is slower than that of phrase-based systems. Though, for some distant language pairs, syntax-based systems outperform phrase-based and hierarchical phrase-based systems. It would be very advantageous to have combination approaches that combine these heterogeneous systems to further improve the translation quality.

Furthermore, there may exist multiple parallel training sets available for a language pair with different genres and sizes. Training the same SMT engine on each training set results in a different model with possibly large gap in translation quality. Again, it is interesting to explore what the best practice is to take advantage of all available data.

The current combination approaches for SMT can be categorized into four classes:

Data Concatenation: A natural combination approach when having multiple training sets is to simply concatenate them into a single corpus and use it to train an SMT model. Although this method is appropriate when the training sets are from the same genre/topic, when they are from different genres and sizes, it does not work as expected. When combining corpora with quite different sizes, the concatenated corpus will be most similar to the biggest corpus and the information in the small ones will be washed out. Different approaches have been proposed to tackle this problem, including selecting subsets of corpora that match the domain of the test set (Yasuda et al., 2008), corpus/sentence level weighting of the training set (Matsoukas et al., 2009), and phrase-level weighting (Foster et al., 2010).

Output Concatenation: Early approaches in system combination combine the output of different systems. These combination approaches assume no information regarding the nature of the component systems, nor the posterior distribution over the outputs. The combination of outputs can happen at the word, phrase or sentence level. In the word or phrase-level combination, the common practice is that one of the translations is selected as the backbone and other candidate translations are aligned to this backbone. A graph called *confusion network* is constructed by combining these aligned sentences (Bangalore et al., 2001). Then, a path through this network is chosen using a voting approach and the sentence corresponding to that path is used as the consensus translation. This method is able to generate new sentences that none of the system can generate. Rosti et al. (2007) showed that word-level combination provides the most robust gains but the best results were achieved by combining all the three levels (i.e. word, phrase and sentence).

Mixture Models: Separate translation and/or language models can be trained on each training set and they can be combined using mixture models to form a single one. The combined TM/LM models are fed into the decoder and the translation process proceeds as before. Two well-known mixture models for SMT are *log-linear mixture models* and *linear mixture models* (Foster and Kuhn, 2007). Details of these models are explained in Section 2.2.

Collaborative Decoding: Decoding is the process of searching for the best output $\hat{e}$ given the input $\bar{f}$ under the probabilistic model. In this approach, multiple translation models are fed into the decoder and the decoder uses hypotheses of all models to find the best derivation and translation. The resulted hypothesis search space is either the union of each model's search space (as in model combination approach of DeNero et al. (2010)) or the intermix of them (as in Koehn and Schroeder (2007) and Liu et al. (2009)).

One of the main applications of system combination is domain adaptation.

1.3 Domain Adaptation

Statistical machine translation (SMT) systems require large parallel corpora in order to obtain a reasonable translation quality. In statistical learning theory, it is assumed that the training and the test datasets are drawn from the same distribution, or in other words, they are from the same domain. However, bilingual corpora are only available in very limited domains and building bilingual resources in a new domain is usually very expensive. It is an interesting question whether a model that is trained on an existing large bilingual corpus in a specific domain can be adapted to another domain for which little parallel data is present. Domain adaptation techniques aim at finding ways to adjust an *out-of-domain* (OUT) model to represent a target domain (*in-domain* or IN).

Common techniques for model adaptation adapt two main components of contemporary state-of-the-art SMT systems: the language model and the translation model. However, language model adaptation is a more straight-forward problem compared to translation model adaptation, because various measures such as perplexity of adapted language models can be easily computed on the data in the target domain. As a result, language model adaptation has been well studied in various work (Clarkson and Robinson, 1997; Seymore and Rosenfeld, 1997; Bacchiani and Roark, 2003; Eck et al., 2004) both for speech recognition and for machine translation. It is also easier to obtain monolingual data in the target domain, compared to bilingual data which is required for translation model adaptation. We expect domain adaptation for machine translation can be improved further by combining orthogonal techniques for translation model adaptation combined with language model adaptation.

1.4 Thesis Outline

Chapter 2: Ensemble Decoding This chapter introduces a novel collaborative decoding approach to combine multiple translation models in the decoder. We evaluate performance on a domain adaptation setting where we translate sentences from the medical domain. Our experimental results show that ensemble decoding outperforms various strong baselines including mixture models, the current state-of-the-art for domain adaptation in machine translation.

Chapter 3: Ensemble Triangulation This chapter uses the *ensemble decoding* approach to alleviate the problem of scarce parallel corpora for resource-poor languages by *triangulation*. Triangulation uses a third language as a pivot through which another source-target translation system can be built. We dynamically create multiple such triangulated systems and combine them in the decoder. Experimental results of this approach show significant improvements in the BLEU score over the direct source-target system. Our approach also outperforms a strong linear mixture baseline.

Chapter 4: Stacking We propose the use of *stacking*, an ensemble learning technique, to the statistical machine translation (SMT) models. A diverse ensemble of weak learners is created using the same SMT engine (a hierarchical phrase-based system) by manipulating the training data and a strong model is created by combining the weak models on-the-fly. Experimental results on two language pairs and three different sizes of training data show significant improvements of up to 4 BLEU points over a conventionally trained SMT model.

Chapter 5: Graph Propagation for Paraphrasing OOV Words In this chapter, we propose a novel approach to finding translations for out-of-vocabulary (oov) words by constructing a graph on a source language monolingual text and employ a graph propagation technique in order to find translations for all the source language phrases. Experimental results show that our graph propagation method significantly improves performance over two strong baselines under intrinsic and extrinsic evaluation metrics.

Chapter 6: Conclusion This chapter summarizes the previous chapters and discusses how those individual models can be combined in future work to create an integrated model suitable for resource-poor languages.

Appendix A: Kriya In this appendix, we describe our in-house implementation of hierarchical phrase-based MT systems, which has been used as the baseline system in many of our experiments throughout this thesis. The ensemble decoder has been built by modifying this Hiero decoder.

1.5 Research Contributions

This dissertation contains several research contributions:

- We introduce a novel system-combination approach that combines multiple translation models in the decoder on-the-fly;

- We use the ensemble decoding approach to handle triangulation and we show it is superior to the conventional pre-processing mixture model approach;

- We propose a novel approach to adopting *stacking* to statistical machine translation and we show how the quality of translation can be boosted almost for free.

- We introduce a novel approach to benefit from source-language-side monolingual text to enhance translation quality measured by BLEU.

Chapter 2

Ensemble Decoding

Statistical machine translation is often faced with the problem of combining training data from many diverse sources into a single translation model which then has to translate sentences in a new domain. We propose a novel approach, ensemble decoding, which combines a number of translation systems dynamically at the decoding step. In this chapter, we evaluate performance on a domain adaptation setting where we translate sentences from the medical domain. Our experimental results show that ensemble decoding outperforms various strong baselines including mixture models, the current state-of-the-art for domain adaptation in machine translation.

2.1 Introduction

Statistical machine translation (SMT) systems require large parallel corpora in order to be able to obtain a reasonable translation quality. In statistical learning theory, it is assumed that the training and test datasets are drawn from the same distribution, or in other words, they are from the same domain. However, bilingual corpora are only available in very limited domains and building bilingual resources in a new domain is usually very expensive. It is an interesting challenge whether a model that is trained on an existing large bilingual corpus in a specific domain can be adapted to another domain for which little parallel data is present. Domain adaptation techniques aim at finding ways to adjust an *out-of-domain* (OUT) model to represent a target domain (*in-domain* or IN).

Common techniques for model adaptation adapt two main components of contemporary state-of-the-art SMT systems: the language model and the translation model. However, language model adaptation is a more straight-forward problem compared to translation model adaptation, because various measures such as perplexity of adapted language models can be easily computed on data in the target domain. As a result, language model adaptation has been well studied in various work (Clarkson and Robinson, 1997; Seymore and Rosenfeld, 1997; Bacchiani and Roark, 2003; Eck et al., 2004) both for speech recognition and for machine translation. It is also easier to obtain monolingual data in the target domain, compared to bilingual data which is required for translation model adaptation. In this work, we focused on adapting only the translation model by fixing a language model for all the experiments. We expect domain adaptation for machine translation can be improved further by combining orthogonal techniques for translation model adaptation combined with language model adaptation.

In this chapter, a new approach for adapting the translation model is proposed. We use a novel system combination approach called ensemble decoding in order to combine two or more translation models with the goal of constructing a system that outperforms all the component models. The strength of this system combination method is that the systems are combined in the decoder. This enables the decoder to pick the best hypotheses for each span of the input. The main applications of ensemble models are domain adaptation, domain mixing and system combination. We have modified *Kriya* (Sankaran et al., 2012), an in-house implementation of hierarchical phrase-based translation system (Chiang, 2005a), to implement ensemble decoding using multiple translation models. Kriya has been explained in Appendix A.

We compare the results of ensemble decoding with a number of baselines for domain adaptation. In addition to the basic approach of concatenation of in-domain and out-of-domain data, we also trained a log-linear mixture model (Foster and Kuhn, 2007) as well as the linear mixture model of (Foster et al., 2010) for conditional phrase-pair probabilities over IN and OUT. Furthermore, within the framework of ensemble decoding, we study and evaluate various methods for combining translation tables.

2.2 Baselines

The natural baseline for model adaptation is to concatenate the IN and OUT data into a single parallel corpus and train a model on it. In addition to this baseline, we have experimented with two more sophisticated baselines which are based on mixture techniques.

2.2.1 Log-Linear Mixture

Log-linear translation model (TM) mixtures are of the form:

$$p(\bar{e}|\bar{f}) \propto \exp\left(\sum_{m}^{M} \lambda_m \log p_m(\bar{e}|\bar{f})\right)$$

where m ranges over IN and OUT, $p_m(\bar{e}|\bar{f})$ is an estimate from a component phrase table, and each λ_m is a weight in the top-level log-linear model, set so as to maximize dev-set BLEU using minimum error rate training (Och, 2003a). We learn separate weights for relative-frequency and lexical estimates for both $p_m(\bar{e}|\bar{f})$ and $p_m(\bar{f}|\bar{e})$. Thus, for 2 component models (from IN and OUT training corpora), there are $4 * 2 = 8$ TM weights to tune. Whenever a phrase pair does not appear in a component phrase table, we set the corresponding $p_m(\bar{e}|\bar{f})$ to a small epsilon value.

2.2.2 Linear Mixture

Linear TM mixtures are of the form:

$$p(\bar{e}|\bar{f}) = \sum_{m}^{M} \lambda_m p_m(\bar{e}|\bar{f})$$

Our technique for setting λ_m is similar to that outlined in Foster et al. (2010). We first extract a joint phrase-pair distribution $\tilde{p}(\bar{e}, \bar{f})$ from the development set using standard techniques (HMM word alignment with grow-diag-and symmetrization (Koehn et al., 2003)). We then find the set of weights $\hat{\lambda}$ that minimize the cross-entropy of the mixture $p(\bar{e}|\bar{f})$ with respect to $\tilde{p}(\bar{e}, \bar{f})$:

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmax}} \sum_{\bar{e},\bar{f}} \tilde{p}(\bar{e}, \bar{f}) \log \sum_{m}^{M} \lambda_m p_m(\bar{e}|\bar{f})$$

For efficiency and stability, we use the EM algorithm to find $\hat{\lambda}$, rather than L-BFGS as in (Foster et al., 2010). Whenever a phrase pair does not appear in a component phrase table, we set the corresponding $p_m(\bar{e}|\bar{f})$ to 0; pairs in $\tilde{p}(\bar{e}, \bar{f})$ that do not appear in at least one component table are discarded. We learn separate linear mixtures for relative-frequency and lexical estimates for both $p(\bar{e}|\bar{f})$ and $p(\bar{f}|\bar{e})$. These four features then appear in the top-level model as usual – there is no runtime cost for the linear mixture.

2.3 Ensemble Decoding

Ensemble decoding is a way to combine the expertise of different models in one single model. The current implementation is able to combine hierarchical phrase-based systems (Chiang, 2005a) as well as phrase-based translation systems (Koehn et al., 2003). However, the method can be easily extended to support combining a number of heterogeneous translation systems e.g. phrase-based, hierarchical phrase-based, and/or syntax-based systems. This section explains how such models can be combined during decoding.

Given a number of SMT systems which are already trained and tuned, the ensemble decoder uses hypotheses constructed from all of the models in order to translate a sentence. We use the bottom-up CKY parsing algorithm for decoding. For each sentence, a CKY chart is constructed. The cells of the CKY chart are populated with appropriate rules from all the phrase tables of different components. As in the Hiero SMT system (Chiang, 2005a), the cells which span up to a certain length (i.e. the maximum span length) are populated from the phrase-tables and the rest of the chart uses *glue rules* as defined in (Chiang, 2005a).

The rules suggested from the component models are combined in a single set. Some of the rules may be unique and others may be common with other component model rule sets, though with different scores. Therefore, we need to combine the scores of such common rules and assign a single score to them. Depending on the mixture operation used for combining the scores, we would get different mixture scores. The choice of mixture operation will be discussed in Section 2.3.1.

Figure 2.1 illustrates how the CKY chart is filled with the rules. Each cell, covering a span, is populated with rules from all component models as well as from cells covering a sub-span of it. This figure shows two systems with different language and translation models and weight vectors, contributing to a decoder. Each system contributes three translations for the French word *il* with different scores (in log scale). The ensemble decoder combines these rules along with their scores and produces a better translation candidate ranking.

In the typical log-linear SMT models, the posterior probability for each phrase pair $(\bar{e}, \bar{f})$ is given by:

$$\begin{aligned} p(\bar{e} \mid \bar{f}) &\propto \exp\Big(\sum_i w_i \phi_i(\bar{e}, \bar{f})\Big) \\ &\propto \exp\Big(\mathbf{w} \cdot \boldsymbol{\phi}\Big) \end{aligned}$$

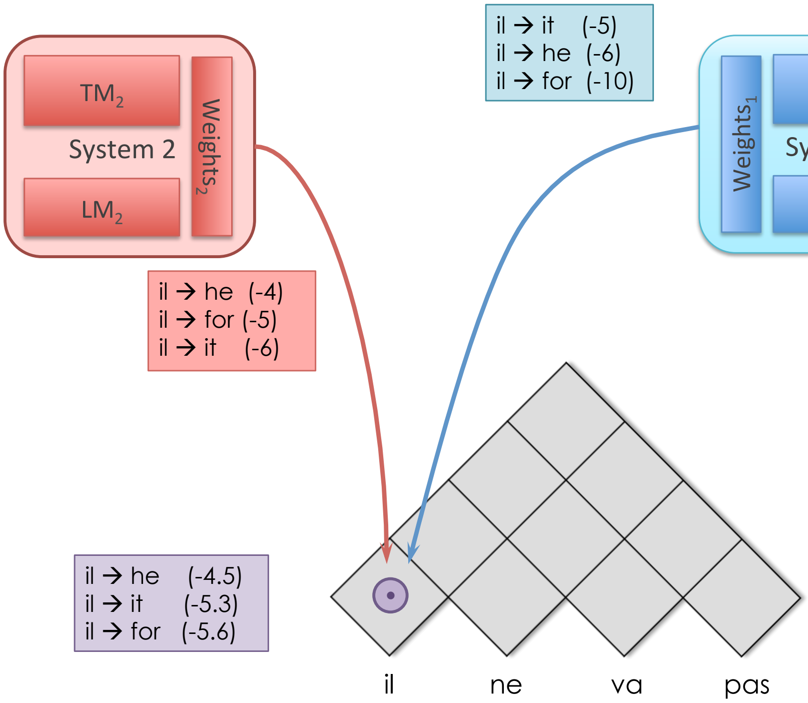


Figure 2.1: The cells in the CKY chart are populated using rules from all component models as well as their sub-spans' cells. This figure shows two systems with different language and translation models and weight vectors that are being combined in a decoder.

Ensemble decoding uses the same framework for each individual system. Therefore, the score of a phrase-pair $(\bar{e}, \bar{f})$ in the ensemble model is:

$$p(\bar{e} \mid \bar{f}) \propto \exp\Big(\underbrace{\mathbf{w_1} \cdot \boldsymbol{\phi_1}}_{1^{st}\text{ model}} \odot \underbrace{\mathbf{w_2} \cdot \boldsymbol{\phi_2}}_{2^{nd}\text{ model}} \odot \cdots \Big)$$

where $\odot$ denotes the mixture operation between two or more model scores.

2.3.1 Mixture Operations

Mixture operations receive two or more scores (probabilities) and return the mixture score (probability). In this section, we explore different options for mixture operation and discuss some of the characteristics of these mixture operations.

- **Weighted Sum (wsum):** in *wsum* the ensemble probability is proportional to the weighted sum of all individual model probabilities (i.e. linear mixture).

  $$p(\bar{e} \mid \bar{f}) \quad \propto \quad \sum_{m}^{M} \lambda_m \exp\left(\mathbf{w}_m \cdot \boldsymbol{\phi}_m\right)$$

  where m denotes the index of component models, M is the total number of them and λ_i is the weight for component i.

- **Weighted Max (wmax):** where the ensemble score is the weighted max of all model scores.

  $$p(\bar{e} \mid \bar{f}) \quad \propto \quad \max_{m} \left(\lambda_m \exp\left(\mathbf{w}_m \cdot \boldsymbol{\phi}_m\right)\right)$$

- **Model Switching (Switch):** in model switching, each cell in the CKY chart gets populated only by rules from one of the models and the other models' rules are discarded. This is based on the hypothesis that each component model is an expert on certain parts of sentence. In this method, we need to define a binary indicator function $\delta(\bar{f}, m)$ for each span and component model to specify rules of which model to retain for each span.

  $$\delta(\bar{f}, m) = \begin{cases} 1, & m = \underset{n \in M}{\operatorname{argmax}} \ \psi(\bar{f}, n) \\ 0, & \text{otherwise} \end{cases}$$

  The criteria for choosing a model for each cell, $\psi(\bar{f}, n)$, could be based on:

  - **Max:** for each cell, the model that has the highest weighted best-rule score wins:

    $$\psi(\bar{f}, n) = \lambda_n \max_{\bar{e}}(\mathbf{w}_n \cdot \boldsymbol{\phi}_n(\bar{\mathbf{e}}, \bar{\mathbf{f}}))$$

  - **Sum:** Instead of comparing only the scores of the best rules, the model with the highest weighted sum of the probabilities of the rules wins. This sum has to take into account the translation table limit (*ttl*), on the number of rules suggested by each model for each cell:

$$\psi(\bar{f}, n) = \lambda_n \sum_{\bar{e}} \exp\left(\mathbf{w}_n \cdot \boldsymbol{\phi}_n(\bar{\mathbf{e}}, \bar{\mathbf{f}})\right)$$

The probability of each phrase-pair $(\bar{e}, \bar{f})$ is then computed as:

$$p(\bar{e} \mid \bar{f}) = \sum_{m}^{M} \delta(\bar{f}, m)\; p_m(\bar{e} \mid \bar{f})$$

- **Product (prod):** in Product models or Product of Experts (Hinton, 1999), the probability of the ensemble model or a rule is computed as the product of the probabilities of all components (or equally the sum of log-probabilities, i.e. log-linear mixture). Product models can also make use of weights to control the contribution of each component. These models are generally known as *Logarithmic Opinion Pools (LOPs)* where:

  $$p(\bar{e} \mid \bar{f}) \;\; \propto \;\; \exp\left(\sum_{m}^{M} \lambda_m \left(\mathbf{w}_m \cdot \boldsymbol{\phi}_m\right)\right)$$

  Product models have been used in combining LMs and TMs in SMT as well as some other NLP tasks such as ensemble parsing (Petrov, 2010).

In Section 2.4.3, we compare the BLEU scores of different mixture operations on a French-English experimental setup.

2.3.2 A Semiring Definition

A *semiring* is an algebraic structure generalizing the arithmetic operations of addition and multiplication. Formally, a semiring is a 5-tuple $R = (A, \oplus, \otimes, \bar{0}, \bar{1})$ such that:

- A is a set (e.g. $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}$);
- $\oplus$ is a binary operation that is both associative and commutative on the set A (e.g. $+$ for natural numbers);

  $$a \oplus b = b \oplus a$$

  $$a \oplus (b \oplus c) = (a \oplus b) \oplus c$$

- $\otimes$ is a binary operation that is associative on the set A (e.g. × for natural numbers);

$$a \otimes (b \otimes c) = (a \otimes b) \otimes c$$

- $\otimes$ distributes over $\oplus$:

$$(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$$

$$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$$

- $\bar{0}$ is the identity element for $\oplus$:

$$a \oplus \bar{0} = \bar{0} \oplus a = a$$

- $\bar{1}$ is the identity element for $\otimes$:

$$a \otimes \bar{1} = \bar{1} \otimes a = a$$

- $\bar{0}$ is an annihilator for $\otimes$:

$$\bar{0} \otimes a = a \otimes \bar{0} = \bar{0}$$

Some examples of semirings are: *natural numbers semiring*: $(\mathbb{N}_0^\infty, +, \times, 0, 1)$ and *boolean semiring*: $(\{0, 1\}, \vee, \wedge, 0, 1)$.

The CKY algorithm assigns scores to derivations by summing the scores of constituent rules in that derivation.

$$\begin{aligned} S(d) &= \sum_{\forall r \in d} s(r) \\ &= \sum_{\forall r \in d} \log p(r) \end{aligned}$$

where r is a context-free rule with score of $s(r) = \log p(r)$ in the derivation d from a set of derivations D and $S(d)$ is the score of that derivation. In the Viterbi decoding, the derivation with highest score is picked by the algorithm and its yield is returned as the best translation.

$$\hat{S} = \max_{\forall d \in D} S(d)$$

This can be formulated using the semiring $(\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$ on the log-space scores. In ensemble decoding, each rule's score $s(r)$ is defined as the mixed scores over all component models' scores:

$$s(r) = \bigodot_{\forall m \in M} s_m(r)$$

where $s_m(r)$ is the score of rule r according to the model m. The ensemble decoding semiring would be:

$$(\mathbb{R} \cup \{-\infty\},\ \max,\ \sum \bigodot,\ -\infty,\ 0)$$

where $\odot$ is a mixture operation over scores of component models on r and

- **wsum:**
$$a \odot b = \log(\lambda_1\, e^a + \lambda_2\, e^b)$$
- **wmax:**
$$\begin{aligned} a \odot b &= \log(\max(\lambda_1\, e^a,\ \lambda_2\, e^b)) \\ &= \max(a + \log \lambda_1,\ b + \log \lambda_2) \end{aligned}$$
- **prod:**
$$a \odot b = \lambda_1 a + \lambda_2 b$$
- **switching:**
$$a \odot b = \begin{cases} a & \lambda_1\, a^* \geq \lambda_2\, b^* \\ \\ b & \textit{otherwise} \end{cases}$$
where a^* and b^* are the scores of top rules of the two component models for *switching:max* and sum of the *exp* of all rules scores for *switching:sum*.

2.3.3 Normalization

Since in log-linear models, the model scores are not normalized to form probability distributions, the scores that different models assign to each phrase-pair may not be in the same scale. Therefore, mixing their scores might wash out the information in one (or some) of the models. We experimented with two different ways to deal with this normalization issue.

A practical but inexact heuristic is to normalize the scores over a shorter list. So the list of rules coming from each model for a cell in CKY chart is normalized before getting mixed with other phrase-table rules. However, experiments showed changing the scores with the normalized scores hurts the BLEU score radically. So we use the normalized scores only for pruning and the actual scores are intact. We could also globally normalize the scores to obtain posterior probabilities using the inside-outside algorithm. However, we did not try it as the BLEU scores we got using the normalization heuristic was not promising and it would impose a cost in decoding as well. More investigation on this issue has been left for future work (Section 2.7.1).

A more principled way is to systematically find the most appropriate model weights that can avoid this problem by scaling the scores properly. We used a publicly available toolkit, Condor (Vanden Berghen and Bersini, 2005), a direct optimizer based on Powell's algorithm, that does not require explicit gradient information for the objective function. Component weights for each mixture operation are optimized on the dev set using Condor to maximize the BLEU score (Papineni et al., 2002b).

2.4 Experiments & Results

We experimented with ensemble decoding in a domain adaptation setting, where a model trained in a domain (OUT) is adapted to another domain (IN). We do the adaptation by training two models on IN and OUT and combining them in the decoder using ensemble decoding.

2.4.1 Baselines

The natural baseline for model adaptation as discussed before is to concatenate the IN and OUT data into a single parallel corpus and train a model on it. In addition to this baseline, we have experimented with two more sophisticated baselines which are based on mixture techniques: linear and log-linear mixtures. These two state-of-the-art baselines are implemented following Foster et al. (2010).

Dataset	**Sents**	**Words**	
		French	**English**
EMEA (IN)	11770	168K	144K
Europarl (OUT)	1.3M	40M	37M
Dev	1533	29K	25K
Test	1522	29K	25K

Table 2.1: Training, dev and test sets for EMEA.

2.4.2 Experimental Setup

We carried out translation experiments using the European Medicines Agency (EMEA) corpus (Tiedemann, 2009) as IN, and the Europarl (EP) corpus[1] as OUT, for French to English translation. The dev and test sets were randomly chosen from the EMEA corpus. The details of datasets used are summarized in Table 2.1.

For the mixture baselines, we used a standard one-pass phrase-based system (Koehn et al., 2003), Portage (Sadat et al., 2005), with the following 7 features: relative-frequency and lexical translation model (TM) probabilities in both directions; word-displacement distortion model; language model (LM) and word count. The corpus was word-aligned using both HMM and IBM2 models, and the phrase table was the union of phrases extracted from these separate alignments, with a length limit of 7. It was filtered to retain the top 20 translations for each source phrase using the TM part of the current log-linear model.

For ensemble decoding, we modified an in-house implementation of hierarchical phrase-based system, *Kriya* (Sankaran et al., 2012), which uses the same features mentioned in (Chiang, 2005a): forward and backward relative-frequency and lexical TM probabilities; LM; word, phrase and glue-rules penalty. GIZA++(Och and Ney, 2000) has been used for word alignment with phrase length limit of 7.

In both systems, feature weights were optimized using MERT (Och, 2003a) and a 5-gram language model and Kneser-Ney smoothing was used in all the experiments. We used SRILM (Stolcke, 2002a) as the language model toolkit. Fixing the language model allows us to compare various translation model combination techniques.

[1] https://www.statmt.org/europarl

Baseline	**PBS**	**Hiero**
IN	31.84	33.69
OUT	24.08	25.32
IN + OUT	31.75	33.76
LOGLIN	32.21	–
LINMIX	33.81	**35.57**

Table 2.2: The results of various baselines implemented in a phrase-based (PBS) and a Hiero SMT on EMEA.

Mixture Operation	**Uniform**	**Tuned**	**Norm.**
Wmax	35.39	35.47 (s=0.03)	35.47
Wsum	35.35	35.53 (s=0.04)	35.45
Switching:Max	35.93	**35.96** (s=0.01)	32.62
Switching:Sum	34.90	34.72 (s=0.23)	34.90
Prod	33.93	35.24 (s=0.05)	35.02

Table 2.3: The results of ensemble decoding on EMEA for *fr* → *en* when using uniform weights, tuned weights and normalization heuristic. The tuned BLEU scores are averaged over three runs with multiple initial points with the standard deviations in brackets.

2.4.3 Results

Table 2.2 shows the results of the baselines. The first group are the baseline results on the phrase-based system discussed in Section 2.4.1 and the second group are those of our hierarchical MT system. Since the Hiero baselines results were substantially better than those of the phrase-based model, we also implemented the best-performing baseline, linear mixture, in our Hiero-style MT system and in fact it achieves the highest BLEU score among all the baselines as shown in Table 2.2. This baseline is run three times and the score is averaged over the BLEU scores with standard deviation of 0.34.

Table 2.3 shows the results of ensemble decoding with different mixture operations and model weight settings. Each mixture operation has been evaluated on the test-set by setting the component weights uniformly (denoted by *uniform*) and by tuning the weights using Condor (denoted by *tuned*) on the dev set. The tuned scores (3rd column in Table 2.3) are averages of three runs with different initial points as in Clark et al. (2011). We also reported the BLEU scores when we applied the span-wise normalization heuristic. All of these mixture operations were able to significantly improve over the concatenation baseline. In particular, *Switching:Max* could gain up to 2.2 BLEU points over the concatenation baseline and 0.39 BLEU points over the best performing baseline (i.e. linear mixture model implemented in Hiero) which is statistically significant based on Clark et al. (2011) ($p = 0.02$).

Prod when using with uniform weights gets the lowest score among the mixture operations, however after tuning, it learns to bias the weights towards one of the models and hence improves by 1.31 BLEU points. Although *Switching:Sum* outperforms the concatenation baseline, it is substantially worse than other mixture operations. One explanation that *Switching:Max* is the best performing operation and *Switching:Sum* is the worst one, despite their similarities, is that *Switching:Max* prefers more peaked distributions while *Switching:Sum* favors a model that has fewer hypotheses for each span.

An interesting observation based on the results in Table 2.3 is that uniform weights are doing reasonably well given that the component weights are not optimized and therefore model scores may not be in the same scope (refer to discussion in §2.3.3). We suspect this is because a single LM is shared between both models. This shared component controls the variance of the weights in the two models when combined with the standard L-1 normalization of each model's weights and hence prohibits models to have too varied scores for the same input. Though, it may not be the case when multiple LMs are used which are not shared.

Two sample sentences from the EMEA test-set along with their translations by the IN, OUT and Ensemble models are shown in Figure 2.2. The boxes show how the Ensemble model is able to use n-grams from the IN and OUT models to construct a better translation than both of them. In the first example, there are two oovs (i.e. out-of-vocabulary), one for each of the IN and OUT models. Our approach is able to resolve the oov issues by taking advantage of the other model's presence. Similarly, the second example shows how ensemble decoding improves lexical choices as well as word re-orderings.

2.4.4 WMT experiments

We also participated in the WMT'12 machine translation shared task in *fr* → *en*. We trained a simplified version of hierarchical phrase-based models where the right-hand side can have at most one non-terminal (denoted as 1NT) instead of the usual two non-terminal (2NT) model. Sankaran et al. (2012) found that the 1NT model performs comparably to the 2NT model for close language pairs such as French-English at the same time resulting in

SOURCE	aménorrhée , menstruations irrégulières
REF	amenorrhoea , irregular menstruation
IN	amenorrhoea , menstruations irrégulières
OUT	aménorrhée , irregular menstruation
ENSEMBLE	amenorrhoea , irregular menstruation

SOURCE	le traitement par naglazyme doit être supervisé par un médecin ayant l' expérience de la prise en charge des patients atteints de mps vi ou d' une autre maladie métabolique héréditaire .
REF	naglazyme treatment should be supervised by a physician experienced in the management of patients with mps vi or other inherited metabolic diseases .
IN	naglazyme treatment should be supervisé by a doctor the with in the management of patients with mps vi or other hereditary metabolic disease .
OUT	naglazyme 's treatment must be supervised by a doctor with the experience of the care of patients with mps vi. or another disease hereditary metabolic .
ENSEMBLE	naglazyme treatment should be supervised by a physician experienced in the management of patients with mps vi or other hereditary metabolic disease .

Figure 2.2: Examples illustrating how this method is able to use expertise of both out-of-domain and in-domain systems.

a smaller model. We used the shared-task training data consisting of Europarl (v7), News commentary and UN documents for training the translation models having a total of 15 M sentence pairs (we did not use the *fr* → *en* Giga parallel corpus for the training). We trained a 5-gram language model for English using the English Gigaword (v4).

In addition to the baseline system, we also trained separate systems for *News* and *Non-News* genres for applying ensemble decoding. The news genre system was trained only using the news-commentary corpus (about $137K$ sentence pairs) and the non-news genre system was trained on the Europarl and UN documents data ($14.8M$ sentence pairs). The idea is to effectively use the small amount of news genre data in order to maximize the performance on the news-based test sets.

Mix. Operation	**Weights**	**Base**	**Norm.**
WMAX	uniform	27.67	27.94
WSUM	uniform	27.72	27.95
SWITCH:MAX	uniform	27.96	26.21
SWITCH:SUM	uniform	27.98	27.98
PROD	uniform	**27.99**	**28.09**
PROD	optimized	**28.25**	28.11

Table 2.4: Applying ensemble decoding with different mixture operations on the Test-11 dataset.

Method	**Devset**	**Test-11**	**Test-12**
Baseline Hiero	26.03	27.63	**28.15**
News data	24.02	26.47	26.27
Non-news data	26.09	27.87	28.15
Ensemble PROD	25.66	**28.25**	28.09

Table 2.5: French-English BLEU scores.

We used 7567 sentence pairs from news-tests 2008 through 2010 for tuning and used news-test 2011 for testing in addition to the 2012 eval set.

Table 2.4 shows the results of applying various mixture operations on the devset and testset, both in normalized (denoted by Norm.) and un-normalized settings (denoted by Base). We present results for these mixture operations using uniform weights (i.e. untuned weights) and for PROD we also present the results using the weights optimized by CONDOR. Most of the mixture operations outperform the Test-11 BLEU of the baseline models (shown in Table 2.5) even with uniform (untuned) weights. We took the best performing operation (i.e. PROD) and tuned its component weights using our optimizer which lead to 0.26 points improvement over its uniform-weight version.

The results for the French-English experiments are reported in Table 2.5. We note that both baseline Hiero model and the model trained from the non-news genre get comparable BLEU scores. The news genre model however gets a lesser BLEU score and this is to be expected due to the very small training data available for this genre.

The last row in Table 2.5 reports the BLEU score for PROD with the tuned weights on the Test-12 dataset and it is marginally less than the baseline model. The reason is that PROD works best when the base systems are very close to one another with a small amount of diversity. In Test-11, the News and Non-news systems are performing closer compared to Test-12. Once the difference between the base models get larger, the PROD fails to improve over the baseline.

Section 4.4.4 discusses the decoding time of this approach and shows that the decoding time complexity is sub-linear in the number of component models participated in the ensemble.

2.5 Related Work

2.5.1 Domain Adaptation

Early approaches to domain adaptation involved information retrieval techniques where sentence pairs related to the target domain were retrieved from the training corpus using IR methods (Eck et al., 2004; Hildebrand et al., 2005). Foster et al. (2010), however, uses a different approach to select related sentences from OUT. They use language model perplexities from IN to select relevant sentences from OUT. These sentences are used to enrich the IN training set.

Other domain adaptation methods involve techniques that distinguish between general and domain-specific examples (Daumé and Marcu, 2006). Jiang and Zhai (2007) introduce a general instance weighting framework for model adaptation. This approach tries to penalize misleading training instances from OUT and assign more weight to IN-like instances than OUT instances. Foster et al. (2010) propose a similar method for machine translation that uses features to capture degrees of generality. Particularly, they include the output from an SVM classifier that uses the intersection between IN and OUT as positive examples. Unlike previous work on instance weighting in machine translation, they use phrase-level instances instead of sentences.

A large body of work uses interpolation techniques to create a single TM/LM from interpolating a number of LMs/TMs. Two famous examples of such methods are linear mixtures and log-linear mixtures (Koehn and Schroeder, 2007; Civera and Juan, 2007; Foster and Kuhn, 2007) which were used as baselines and discussed in Section 2.4.1. Other methods include using self-training techniques to exploit monolingual in-domain data (Ueffing et al., 2007; Bertoldi and Federico, 2009). In this approach, a system is trained on the parallel OUT and IN data and it is used to translate the monolingual IN data set. Iteratively, most confident sentence pairs are selected and added to the training corpus on which a new system is trained.

2.5.2 System Combination

Tackling the model adaptation problem using system combination approaches has been experimented in various work (Koehn and Schroeder, 2007; Hildebrand and Vogel, 2009). Among these approaches are sentence-based, phrase-based and word-based output combination methods. In a similar approach, Koehn and Schroeder (2007) use a feature of the factored translation model framework in Moses SMT system (Koehn and Schroeder, 2007) to use multiple alternative decoding paths. Two decoding paths, one for each translation table (IN and OUT), were used during decoding. The weights are set with minimum error rate training (Och, 2003a).

Our work is closely related to Koehn and Schroeder (2007) but uses a different approach to deal with multiple translation tables. The Moses SMT system implements Koehn and Schroeder (2007)'s approach and can treat multiple translation tables in two different ways: *intersection* and *union*. In *intersection*, for each span only the hypotheses would be used that are present in all phrase tables. For each set of hypothesis with the same source and target phrases, a new hypothesis is created whose feature-set is the union of feature sets of all corresponding hypotheses. *Union*, on the other hand, uses hypotheses from all the phrase tables. The feature set of these hypotheses are expanded to include one feature set for each table. However, for the corresponding feature values of those phrase-tables that did not have a particular phrase-pair, a default log probability value of 0 is assumed (Bertoldi and Federico, 2009) which is counter-intuitive as it boosts the score of hypotheses with phrase-pairs that do not belong to all of the translation tables.

Our approach is different from Koehn and Schroeder (2007) in a number of ways. Firstly, unlike the multi-table support of Moses which only supports phrase-based translation table combination, our approach supports ensembles of both hierarchical and phrase-based systems. With little modification, it can also support ensemble of syntax-based systems with the other two state-of-the-art SMT systems. Secondly, our combining method uses the *union* option, but instead of preserving the features of all phrase-tables, it only combines their scores using various mixture operations. This enables us to experiment with a number of different operations as opposed to sticking to only one combination method. Finally, by avoiding increasing the number of features we can add as many translation models as we need without serious performance drop. In addition, MERT would not be an appropriate optimizer when the number of features increases a certain amount (Chiang et al., 2008).

Our approach differs from the model combination approach of DeNero et al. (2010), a generalization of consensus or minimum Bayes risk decoding where the search space consists of those of multiple systems, in that model combination uses forest of derivations of all component models to do the combination. In other words, it requires all component models to fully decode each sentence, compute n-gram expectations from each component model and calculate posterior probabilities over translation derivations. While, in our approach we only use partial hypotheses from component models and the derivation forest is constructed by the ensemble model. A major difference is that in the model combination approach the component search spaces are conjoined and they are not intermingled as opposed to our approach where these search spaces are intermixed on spans. This enables us to generate new sentences that cannot be generated by component models. Furthermore, various combination methods can be explored in our approach. Finally, main techniques used in that work are orthogonal to our approach such as *minimum Bayes risk* decoding, n-gram features and tuning using MERT.

Finally, our work is most similar to that of Liu et al. (2009) where max-derivation and max-translation decoding have been used. Max-derivation finds a derivation with highest score and max-translation finds the highest scoring translation by summing the score of all derivations with the same yield. The combination can be done in two levels: translation-level and derivation-level. Their derivation-level max-translation decoding is similar to our ensemble decoding with *wsum* as the mixture operation. We did not restrict ourselves to this particular mixture operation and experimented with a number of different mixing techniques and as Table 2.3 shows we could improve over *wsum* in our experimental setup. Liu et al. (2009) used a modified version of MERT to tune max-translation decoding weights, while we use a two-step approach using MERT for tuning each component model separately and then using Condor to tune component weights on top of them.

2.6 Conclusion

In this chapter, we presented a new approach for model combination using ensemble decoding. In this approach a number of MT systems are combined at decoding time in order to form an ensemble model. The model combination can be done using various mixture operations. We showed that this approach can gain up to 2.2 BLEU points over the concatenation baseline and 0.39 BLEU points over a powerful mixture model in a domain adaptation scenario.

2.7 Future Directions

In this section, we investigate possible extensions, applications and further experiments based on ensemble decoding.

In our experiments, we fixed the language model in order to study the effect of mixing translation models. A natural extension would be to allow each translation model to couple with a separate (or multi) language model(s). These enhancements will allow us to mix multiple systems in the decoder instead of multiple translation models. Normalization (discussed in Section 2.7.1) is a prerequisite step for this expansion.

2.7.1 Global Normalization

As mentioned in Section 2.3.3, the scores in the log-linear models are not normalized since we only use them to rank hypothesis.

$$p(\bar{e} \mid \bar{f}) \;=\; \frac{\exp\left(\sum_i w_i \phi_i(\bar{e}, \bar{f})\right)}{\sum\limits_{\bar{e'}} \exp\left(\sum_i w_i \phi_i(\bar{e'}, \bar{f})\right)}$$

$$p(\bar{e} \mid \bar{f}) \;\propto\; \exp\left(\sum_i w_i \phi_i(\bar{e}, \bar{f})\right)$$

However, this would cause a problem when using multiple language models in multiple systems. Though, in the experiments reported in Section 2.4 we did not suffer from this problem as a result of using a shared language model in combination with L1 normalized weights. A more principled approach is to exactly compute the normalized scores using the inside-outside algorithm.

In this approach, each system separately parses each sentence without consulting other systems' translation models. All the hypotheses scores are normalized using the inside and outside scores. Next, an ensemble CKY chart is populated from partial hypotheses located in all corresponding CKY chart cells. The rest of the approach remains unchanged.

2.7.2 Domain Mixing Scenario

In this setting, the training, dev and test sets consist of sentences from a variety of domains. However, the sentences are not labeled with the domain they are belonging to. This use case is similar to what translation web services such as Google Translate and Bing Translator face with on daily basis. Eidelman et al. (2012) suggests discovering latent topics (i.e. finer-grained domains) using an unsupervised approach (LDA) and they used these topic distributions to compute topic-dependent lexical weighting probabilities. These probabilities are added to translation models as features. This approach can gain up to 1 BLEU point over a strong baseline.

In this setting, we can take advantage of unsupervised topic modeling toolkits to cluster the corpus into N subcorpora. Then a separate translation model can be learned on each subcorpora and the ensemble decoding approach can be applied on these models. One potential problem with this approach would be sparsity as the translation model probabilities would be estimated on smaller data. One remedy to this problem is to learn a general translation model on the whole corpus and do an ensemble model on this general model and all sub-corpora-based models. Furthermore, learning a separate language model on each subcorpora can also be beneficial when using in conjunction with a general language model.

2.7.3 Mixture Operation Characteristics

In Section 2.3.1, we defined five mixture operations and we reported the BLEU scores when using them in ensemble decoding (Section 2.4). Each of these mixture operations has specific properties that make it work in specific domain adaptation or system combination scenarios. For instance, prod, or in general LOPs, may not be optimal for domain adaptation in the setting where there are two or more models trained on heterogeneous corpora. As discussed in Smith et al. (2005), LOPs work best when all the models accuracies are high and close to each other with some degree of diversity. LOPs give veto power to any of the component models and this perfectly works for settings such as the one in Petrov (2010) where a number of parsers are trained by changing the randomization seeds but having the same base parser and using the same training set. They noticed that parsers trained using different randomization seeds have high accuracies but there are some diversities among them and they used product models for their advantage to build a better parser by combining the base models. We assume that each of the models is expert in some parts and so they do not necessarily agree on correct hypotheses. In other words, product models (or LOPs) tend to have intersection-style effects while we are more interested in union-style effects.

We would like to study the characteristics of other mixture operations and figure out what operations would best work in what settings. The results can be used to potentially come up with better mixture operations.

2.7.4 Consensus Ensemble Decoding

Current SMT systems suffer from spurious ambiguity which is resulted from having many distinct derivations (i.e. trees in hierarchical phrase-based or syntax-based systems or segmentations in phrase-based systems) with same yield. To get the exact posterior probabilities, the partial probabilities need to be summed up:

$$p(e|f) = \sum_{d:yield(d)=e} p(e,d|f)$$

$$e^* = \underset{e}{\text{argmax}}\ \ p(e|f) = \underset{e}{\text{argmax}} \sum_{d:yield(d)=e} p(e,d|f)$$

This is known as *Maximum A Posteriori (MAP)*. However, computing this *argmax* is computationally intractable at decoding. Therefore, most people resort to using Viterbi approximation that only takes into account the most probable derivation:

$$e^* = yield(\underset{d}{\text{argmax}}\ \ p(e,d|f))$$

To alleviate this problem, researchers proposed and applied approaches that consider all the derivations, yet allow tractable decoding, namely *variational decoding*, *minimum Bayes risk* and *consensus decoding*. These methods consistently outperform the Viterbi approximation and they use the same idea for tackling this problem which is taking advantage of n-gram features.

Variational Decoding

Since the exact inference is intractable, in *variational decoding* the posterior probability $p(e|f)$ is approximated by a tractable model $q(e|f)$ (or simply $q(e)$). $q \in Q$ is chosen to minimize some information loss such as the KL divergence $KL(p||q)$ (Li et al., 2009b). If we choose a q that is factorizable, we can use efficient dynamic programming algorithms for tractable decoding. A natural choice for $q(e)$ which depends on the target sentence and is factorizable is the n-gram model families.

The decoder scores each string using the n-gram features collected using the inside-outside algorithm.

$$e^* = \operatorname*{argmax}_{e \in T(f)} \sum_n \theta_n . \log q_n^*(e)$$

where $T(f)$ denotes all translation candidates for input string f and θ is the n-gram weight vector that controls the power of each n-gram type. $q_n^*(e)$ is defined as:

$$q_n^*(e) = \prod_{g \in \text{ngram(e)}} \sum_{e,d} c_g(e) * p(e, d|f)$$

where $c_g(e)$ is the count of n-gram g in the translation e and $p(e, d|f)$ is the posterior probability of the derivation d.

This approach scores translations that have more n-gram overlaps with other translations higher. In this sense, it is very similar to the minimum Bayes risk decoding which will be discussed in the next section.

Minimum Bayes Risk

The *minimum Bayes risk (MBR)* objective aims at minimizing risk based on a loss function (Kumar and Byrne, 2004):

$$\begin{aligned}
e^* &= \operatorname*{argmin}_e R(e) \\
&= \operatorname*{argmin}_e \mathbb{E}_{p(e'|f)}[l(e, e')] \\
&= \operatorname*{argmin}_e \sum_{e'} p(e'|f) l(e, e')
\end{aligned}$$

Bayes risk is defined as the expectation of a loss function l which returns the loss of a translation e with regard to a reference e' (or other translations). Equivalently, we can define it as:

$$\begin{aligned} e^* &= \operatorname*{argmin}_{e} 1 - \sum_{e'} p(e'|f) S(e, e') \\ &= \operatorname*{argmax}_{e} \sum_{e'} p(e'|f) BLEU(e, e') \end{aligned}$$

In other words, this objective tries to find a translation that is most similar, on expectation, to any possible reference translations. The similarity is evaluated based on a function $S(e, e')$. In practice, MBR chooses a translation that maximizes expected similarity to other candidate translations under $p(e|f)$. Since the similarity metric (e.g. BLEU) needs to be computed for all candidate translations and the number of translations is already exponential, in practice, MBR is computed over a k-best list.

Consensus Decoding

DeNero et al. (2009) introduces a variant of MBR, *consensus decoding*, that applies efficiently to translation forests rather than k-best lists. Instead of maximizing expected similarity (i.e. BLEU score), similarity is expressed in terms of n-gram features and translations are scored with respect to similarity to expected feature values:

$$\begin{aligned} e^* &= \operatorname*{argmax}_{e} \mathbb{E}_{p(e'|f)}[BLEU(e, e')] \\ &\approx \operatorname*{argmax}_{e} BLEU(e, \mathbb{E}_{p(e'|f)}[\phi(e')]) \end{aligned}$$

We propose to apply the techniques of consensus decoding in our ensemble method. More specifically, once the normalization step (see Section 2.7.1) is done, ensemble decoding combines hypotheses from all the models. Meanwhile, the n-gram expected counts are collected in the ensemble decoder. Once the input sentence is fully parsed, all the candidate translations are scored based on the new n-gram-based objective function and the translation with highest score is chosen as the system output.

The idea of applying consensus decoding on multiple systems has been successfully used in the *model combination* approach of DeNero et al. (2010). This approach assumes that each system provides expectations of n-gram features, though, it does not care about the latent structure of component systems. The objective function used in this approach is:

$$s_w(d) = \sum_{i=1}^{I} \left(\sum_{n=1}^{4} w_i^n v_i^n(d) + w_i^\alpha \alpha_i(d) \right) + w^b.b(d) + w^l.l(d)$$

This objective function scores a derivation d using n-gram scores from I different systems with weights $\mathbf{w}$. $\alpha_i(d)$ is a system indicator feature which is 1 if the derivation d came from the system i and 0 otherwise. $b(d)$ is the model score of the derivation d under the model it is from and l is the target side length. v_i^n is combination feature function on n-grams for system i, that is:

$$\begin{aligned} v_i^n(d) &= \sum_{g \in \text{ngram}(d)} v_i^n(g) \\ &= \sum_{g \in \text{ngram(d)}} \mathbb{E}_{p_i(d'|f)}[c(g, d')] \\ &= \sum_{g \in \text{ngram(d)}} \sum_{d'} p_i(d'|f) c(g, d') \end{aligned}$$

However, DeNero et al. (2010) do not intermix search spaces from multiple systems while our ensemble decoding method is able to generate new sentences that are not in any of the component systems' search spaces. Another advantage of using consensus decoding on top of ensemble decoding is that we can benefit from the hypergraph-based minimum error-rate training algorithm of Kumar et al. (2009) and have a more systematic tuning procedure, replacing Condor.

Chapter 3

Ensemble Triangulation

State-of-the-art statistical machine translation systems rely heavily on training data and insufficient training data usually results in poor translation quality. One solution to alleviate this problem is *triangulation*. Triangulation uses a third language as a pivot through which another source-target translation system can be built. In this approach, we dynamically create multiple such triangulated systems and combine them using *ensemble decoding* (Chapter 2). Experimental results of this approach show significant improvements in the BLEU score over the direct source-target system. Our approach also outperforms a strong linear mixture baseline.

3.1 Introduction

The objective of current statistical machine translation (SMT) systems is to build cheap and rapid corpus-based SMT systems without involving human translation expertise. Such SMT systems rely heavily on their training data. State-of-the-art statistical machine translation systems automatically extract translation rules (e.g. phrase pairs), learn segmentation models, re-ordering models, etc. and find tuning weights solely from data and hence they rely heavily on high quality training data. There are many language pairs for which there is no parallel data or the available data is not sufficiently large to build a reliable SMT system. For example, there is no Chinese-Farsi parallel text, although there exists sufficient parallel data between these two languages and English. For SMT, an important research direction is to improve the quality of translation when there is no, insufficient or poor-quality parallel data between a pair of languages.

One approach that has been recently proposed is *triangulation*. Triangulation is the process of translating from a source language to a target language via an intermediate language (aka pivot, or bridge). This is very useful specifically for low-resource languages as SMT systems built using small parallel corpora perform poorly due to data sparsity. In addition, ambiguities in translating from one language into another may disappear if a translation into some other language is available.

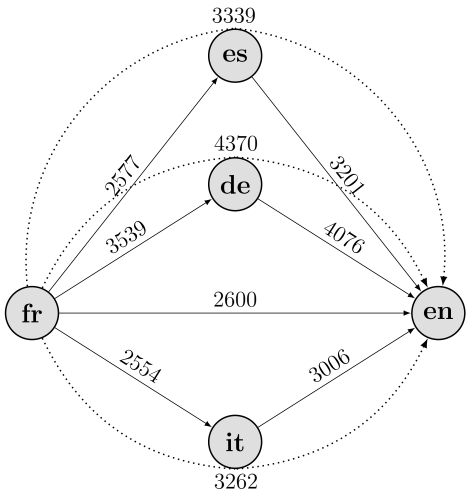
One obvious benefit of triangulation is to increase the coverage of the model on the input text. In other words, we can reduce the number of out-of-vocabulary words (oovs), which are a major cause of poor quality translations, using other paths to the target language. This can be especially helpful when the model is built using a small amount of parallel data.

Figure 3.1 shows how triangulation can be useful in reducing the number of oovs when translating from French to English through three pivot languages: Spanish (*es*), German (*de*) and Italian (*it*). The solid lines show the number of oovs for a direct MT system with regard to a multi-language parallel test set (Section 3.5.2 contains the details about the data sets) and the dotted lines show the number of oovs in the triangulated (*src* → *pvt* → *tgt*) systems. The number of oovs on triangulated paths can never be less that the first edge (i.e. *src* → *pvt*) and it is usually higher than the second edge (i.e. *pvt* → *tgt*) as well. Thus, the choice of intermediate language is very important in triangulation.

Figure 3.1 also shows how combining multiple triangulated systems can reduce this number from 2600 (16%) oovs to 1536 (9%) oovs. Thus, combining triangulated systems with the original *src* → *tgt* system is a good idea. When combining multiple systems, the upper bound on the number of oovs is the minimum among all oovs in the different triangulations. These oov rates provide useful hints, among other clues, as to which pivot languages will be more useful. In Figure 3.1, we can expect Italian (*it*) to help more than Spanish (*es*) and both to help more than German (*de*) in translation from French (*fr*) to English (*en*), which we confirmed in our experimental results (Table 3.2).

In addition to providing translations for otherwise untranslatable phrases, triangulation can find new translations for current phrases. The conditional distributions used for the translation model have been estimated on small amounts of data and hence are not robust due to data sparseness. Using triangulation, these distributions are smoothed and become more reliable as a result.

For each pivot language for which there exists parallel data with the source and the target language, we can create a *src* → *tgt* system by bridging through the pivot language.



direct (*fr*-*en*)	2600 (16%)
triangulated (*fr*-{*es*, *de*, *it*}-*en*)	2066 (12%)
direct + triangulated	1536 (9%)

Figure 3.1: Number of oovs when translating directly from *fr* to *en* (solid lines), triangulating through *es*, *de* or *it* individually (dotted lines), and when combining multiple triangulation systems with the direct system. These oov numbers are based on a multi-language parallel test set and the models are built on small corpora (10k sentence pairs), which are not multi-parallel.

If there are a number of such pivot languages with corresponding data, we can use mixture approaches to combine them in order to build a stronger model. We propose to apply the ensemble decoding approach in this triangulation scenario. We experimented with 12 different language pairs and 3 pivot languages for each source-target language pair. Our experimental results show significant improvements in the BLEU score over the direct source-target system in all the 12 language pairs. We also compare to a strong linear mixture baseline.

3.2 Related Work

Use of pivot languages in machine translation dates back to the early days of machine translation. Boitet (1988) discusses the choice of pivot languages, natural or artificial (e.g. interlingua), in machine translation. Schubert (1988) argues that a proper choice for an intermediate language for high-quality machine translation is a natural language due to the inherent lack of expressiveness in artificial languages.

Previous work in applying pivot languages in machine translation can be categorized into these divisions:

3.2.1 System Cascades

In this approach, a *src* → *pvt* translation system translates the source input into the pivot language and a second *pvt* → *tgt* system takes the output of the previous system and translates it into the target language. Utiyama and Isahara (2007) use this approach to triangulate between Spanish, German and French through English. However, instead of using only the best translation, they took the n-best translations and translated them into the target language. MERT (Och, 2003a) has been used to tune the weights for the new feature set which consists of *src* → *pvt* and *pvt* → *tgt* feature functions. The highest scoring sentence from the target language is used as the final translation. They showed that using 15 hypotheses in the *pvt* side is generally superior to using only one best hypothesis.

3.2.2 Corpus Synthesis

Given a *pvt* → *tgt* MT system, one can translate the pivot side of a *src-pvt* parallel corpus into the target language and create a noisy *src-tgt* parallel corpus. This can also be exploited in the other direction, meaning that a *pvt* → *src* MT system can be used to translate the pivot side of a *pvt-tgt* bitext. de Gispert and Marino (2006), for example, translated the Spanish side of an English-Spanish bitext into Catalan using an available Spanish-Catalan SMT system. Then, they built an English-Catalan MT system by training on this new parallel corpus.

3.2.3 Phrase-Table Triangulation

In this approach, instead of translating the input sentences from a source language to a pivot language and from that to a target language, triangulation is done on the phrase level by triangulating two phrase-tables: *src* → *pvt* and *pvt* → *tgt*:

$$(\bar{f}, \bar{e}) \in T_{\mathcal{F}\rightarrow\mathcal{E}} \iff \exists \bar{i}: \;\; (\bar{f}, \bar{i}) \in T_{\mathcal{F}\rightarrow\mathcal{I}} \;\; \land \;\; (\bar{i}, \bar{e}) \in T_{\mathcal{I}\rightarrow\mathcal{E}}$$

where $\bar{f}, \bar{i}$ and $\bar{e}$ are phrases in the source $\mathcal{F}$, pivot $\mathcal{I}$ and target $\mathcal{E}$ languages respectively and T is a set representing a phrase table.

Utiyama and Isahara (2007) also experimented with phrase-table triangulation. They compared both triangulation approaches when using Spanish, French and German as the source and target languages and English as the only pivot language. They showed that phrase-table triangulation is superior to the MT system cascades but both of them did not outperform the direct *src* → *tgt* system.

The phrase-table triangulation approach with multiple pivot languages has been also investigated in several work (Cohn and Lapata, 2007; Wu and Wang, 2007). These triangulated phrase-tables are combined together using linear and log-linear mixture models. They also successfully combined the mixed phrase-table with a *src-tgt* phrase-table to achieve a higher BLEU score.

Bertoldi et al. (2008) formulated phrase triangulation in the decoder where they also consider the phrase-segmentation model between *src-pvt* and the reordering model between *src-tgt*.

Beside machine translation, the use of pivot languages has found applications in other NLP areas. Gollins and Sanderson (2001) used a similar idea in cross-lingual information retrieval where query terms were translated through multiple pivot languages to the target language and the translations are combined to reduce the error. Pivot languages have also been successfully used in inducing translation lexicons (Mann and Yarowsky, 2001) as well as word alignments for resource-poor languages (Kumar et al., 2007; Wang et al., 2006). Callison-Burch et al. (2006) used pivot languages to extract paraphrases for unknown words.

3.3 Baselines

We compare our approach with two baselines. A simple baseline is the direct system between source and target languages which is trained on the same amount of parallel data as triangulated ones. In addition, we implemented a phrase-table triangulation method (Cohn and Lapata, 2007; Wu and Wang, 2007; Utiyama and Isahara, 2007). This approach presents a probabilistic formulation for triangulation by marginalizing out the pivot phrases, and factorizing using the chain rule:

$$\begin{aligned} p(\bar{e} \,|\, \bar{f}) &= \sum_{\bar{i}} p(\bar{e}, \bar{i} \,|\, \bar{f}) \\ &= \sum_{\bar{i}} p(\bar{e} \,|\, \bar{i}, \bar{f})\, p(\bar{i} \,|\, \bar{f}) \\ &\approx \sum_{\bar{i}} p(\bar{e} \,|\, \bar{i})\, p(\bar{i} \,|\, \bar{f}) \end{aligned}$$

where $\bar{f}, \bar{e}$ and $\bar{i}$ are phrases in the source, target and intermediate language respectively. In this equation, a conditional independence assumption has been made that source $\bar{f}$ and target phrases $\bar{e}$ are independent given their corresponding pivot phrase(s) $\bar{i}$. The equation requires that all phrases in the *src* → *pvt* direction must also appear in *pvt* → *tgt*. All missing phrases are simply dropped from the final phrase-table.

Using this approach, a triangulated source-target phrase-table is generated for each pivot language. Then, linear and log-linear mixture methods are used to combine these phrase-tables into a single phrase-table in order to be used in the decoder. We implemented the linear mixture approach, since linear mixtures often outperform log-linear ones (Cohn and Lapata, 2007). We then compare the results of these baselines with our approach over multiple language pairs (Section 3.5.2). In linear mixture models, each feature in the mixture phrase-table is computed as a linear interpolation of corresponding features in the component phrase-tables using a weight vector $\vec{\lambda}$.

$$\begin{aligned} p(\bar{e} \,|\, \bar{f}) &= \sum_{i} \lambda_i\, p_i(\bar{e} \,|\, \bar{f}) \\ p(\bar{f} \,|\, \bar{e}) &= \sum_{i} \lambda_i\, p_i(\bar{f} \,|\, \bar{e}) \\ \forall\ \lambda_i > 1 & \qquad \sum_{i} \lambda_i = 1 \end{aligned}$$

Following Cohn and Lapata (2007), we combined triangulated phrase-tables with uniform weights into a single phrase table and then interpolated it with the phrase-table of the direct system.

3.4 Our Approach

3.4.1 Dynamic Triangulation

Given a *src* → *pvt* and a *pvt* → *tgt* system which are independently trained and tuned on their corresponding parallel data, these two systems can be triangulated dynamically in the decoder.

For each source phrase $\bar{f}$, the decoder consults the *src* → *pvt* system to get its translations on the pivot side $\bar{i}$ with their scores. Consequently, each of these pivot-side translation phrases is queried from the *pvt* → *tgt* system to obtain their translations on the target side with their corresponding scores. Finally a $(\bar{f},\, \bar{e})$ pair is constructed from each $(\bar{f},\, \bar{i})$ and $(\bar{i},\, \bar{e})$ pair, whose score is computed as:

$$p_{\mathcal{I}}(\bar{f}\,|\,\bar{e}) \;\propto\; \max_{\bar{i}}\; \exp\Big(\underbrace{w_1\,.\,\phi_1(\bar{f},\,\bar{i})}_{\mathcal{F}\to\mathcal{I}} \;+\; \underbrace{w_2\,.\,\phi_2(\bar{i}\,,\,\bar{e})}_{\mathcal{I}\to\mathcal{E}}\Big)$$

This method requires the language model score of the *src* → *pvt* system. However for simplicity we do not use the pivot-side language models and hence the score of the *src* → *pvt* system does not include the language model and word penalty scores. In this formulation for a given source and target phrase pair $(\bar{f},\, \bar{e})$, if there are multiple bridging pivot phrases $\bar{i}$, we only use the one that yields the highest score. This is in contrast with previous work where they take the sum over all such pivot phrases (Cohn and Lapata, 2007; Utiyama and Isahara, 2007). We use *max* as it outperforms *sum* in our preliminary experiments.

It is noteworthy that in computing the score for $p_{\mathcal{I}}(\bar{f}\,|\,\bar{e})$, the scores from *src* → *pvt* and *pvt* → *tgt* are added uniformly. However, there is no reason why this should be the case. Two different weights can be assigned to these two scores to highlight the importance of one against the other one.

A naive implementation of phrase-triangulation in the decoder would require $O(n^2)$ steps for each source sub-span, where n is the average number of translation fan-out (i.e. possible translations) for each phrase. However, since the phrase candidates from both *src* → *pvt* and *pvt* → *tgt* are already sorted, we use a lazy algorithm that reduces the computational complexity to $O(n)$.

3.4.2 Combining Triangulated Systems

If we can make use of multiple pivot languages, a system can be created on-the-fly for each pivot language by triangulation and these systems can then be combined together in the decoder using the *ensemble decoding* approach discussed in Chapter 2. Following previous work, these triangulated phrase-tables can also be combined with the direct system to produce a yet stronger model. However, we do not combine them in two steps. Instead, all triangulated systems and the direct one are combined together in a single step.

Ensemble decoding is aware of full model scores when it compares, ranks and prunes hypotheses. This includes the language model, word, phrase and glue rule penalty scores as well as standard phrase-table probabilities.

Since ensemble decoding combines the scores of common hypotheses across multiple systems rather than combining their feature values as in mixture models, it can be used to triangulate heterogeneous systems such as phrase-based, hierarchical phrase-based, and syntax-based with completely different feature types. Considering that ensemble decoding can be used in these diverse scenarios, it offers an attractive alternative to current phrase-table triangulation systems.

3.4.3 Tuning Component Weights

Component weights control the contribution of each model in the ensemble. A tuning procedure should assign higher weights to the models that produce higher quality translations and lower weights to weak models in order to control their noise propagation in the ensemble. In the ensemble decoder, since we do not have explicit gradient information for the objective function, we use a direct optimizer for tuning. We used *Condor* (Vanden Berghen and Bersini, 2005) which is a publicly available toolkit based on Powell's algorithm.

The ensemble between three triangulated models and a direct one requires tuning in a 4-dimensional space, one for each system. If, on average, the tuner evaluates the decoder n times in each direction in the optimization space, there needs to be n^4 ensemble decoder evaluations, which is very time consuming. Instead, we resorted to a simpler approach for tuning: each triangulated model is separately tuned against the direct model with a fixed weights (we used a weight of 1). In other words, three ensemble models are created, each on a single triangulated model plus the direct one. These ensembles are separately tuned and once completed, these weights comprise the final tuned weights. Thus, the total number of

$\mathbf{L}_1$ - $\mathbf{L}_2$	$\mathbf{L}_1$ tokens (K)	$\mathbf{L}_2$ tokens (K)
de - en	232	249
de - es	232	263
de - fr	231	259
de - it	245	253
en - es	250	264
en - fr	251	262
en - it	260	251
es - fr	262	261
es - it	274	252
fr - it	272	251

Table 3.1: Number of tokens in each language pair in the training data.

ensemble evaluations reduces from $O(n^4)$ to $O(3n)$.

In addition to this significant complexity reduction, this method enables parallelism in tuning, since the three individual tuning branches can now be run independently. The final tuned weights are not necessarily a local optima and one can run further optimization steps around this point to get to even better solutions which should lead to higher BLEU scores.

3.5 Experiments & Results

3.5.1 Experimental Setup

For our experiments, we used the Europarl corpus (v7) (Koehn, 2005) for training sets and ACL/WMT 2005[1] data for dev/test sets (2k sentence pairs) following Cohn and Lapata (2007). Our goal here was to understand how multiple languages can help in triangulation, the improvement in coverage of the unseen data due to triangulation, and the importance of choosing the right languages as pivot languages. Thus, we needed to run experiments on a large number of language pairs, and for each language pair we wanted to work with many pivot languages. To this end, we created small sub-corpora from Europarl by sampling 10,000 sentence pairs and conducted our experiments on them. As we will show, using larger data than this would result in prohibitively large triangulated phrase tables. Table 3.1 shows the number of words on both sides of used language pairs in our corpora.

[1] http://www.statmt.org/wpt05/mt-shared-task/

The ensemble decoder uses the following standard features: forward and backward relative-frequency and lexical TM probabilities; LM; word, phrase and glue-rules penalty. GIZA++ (Och and Ney, 2000) has been used for word alignment with phrase length limit of 10. In both systems, feature weights were optimized using MERT (Och, 2003a). We used the target sides of the Europarl corpus to build 5-gram language models and smooth them using Kneser-Ney algorithm. We used SRILM (Stolcke, 2002a) as the language model toolkit.

3.5.2 Results

Table 3.2 shows the BLEU scores when using two languages from {*fr*, *en*, *es*, *de*} as source and target, and the other two languages plus *it* as intermediate languages. The first group of numbers are BLEU scores for triangulated systems through the specified pivot language. For example, translating from *de* to *es* through *en* (i.e. *de* → *en* → *es*) gets 15.94% BLEU score. The second group shows the BLEU scores of the baseline systems including the direct system between the source and target languages as well as a linear mixture of the three triangulated systems. The BLEU scores of ensemble decoding using different mixture operations are illustrated at the bottom.

As the table shows, our approach outperforms the direct systems in all the 12 language pairs. It also outperforms the mixture models in most cases. Overall, ensemble decoding with *wmax* as mixture operation performs the best among the different systems and baselines. Figure 3.2 shows the average of the BLEU score of the direct systems, mixture models and ensemble decoding with *wmax* as the mixture operation on all 12 systems. On average, the *wmax* method obtains 0.33 BLEU points higher than the mixture models.

We also computed the Meteor scores (Denkowski and Lavie, 2011) for all systems and the results are summarized in Figure 3.3. As the figure illustrates, our ensemble decoding approach with *wmax* outperforms the mixture models in 11 of 12 language pairs based on Meteor scores.

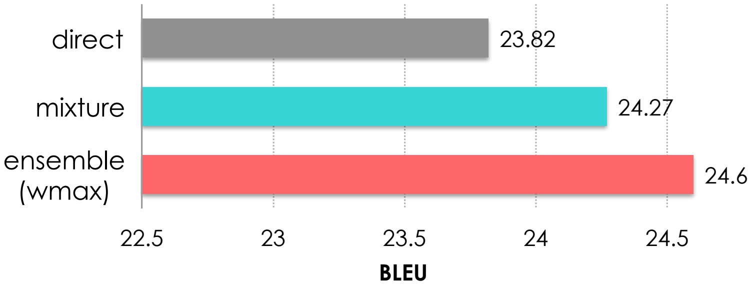


Figure 3.2: The average BLEU scores of the direct system, mixture models and *wmax* ensemble triangulation approach over all 12 language pairs.

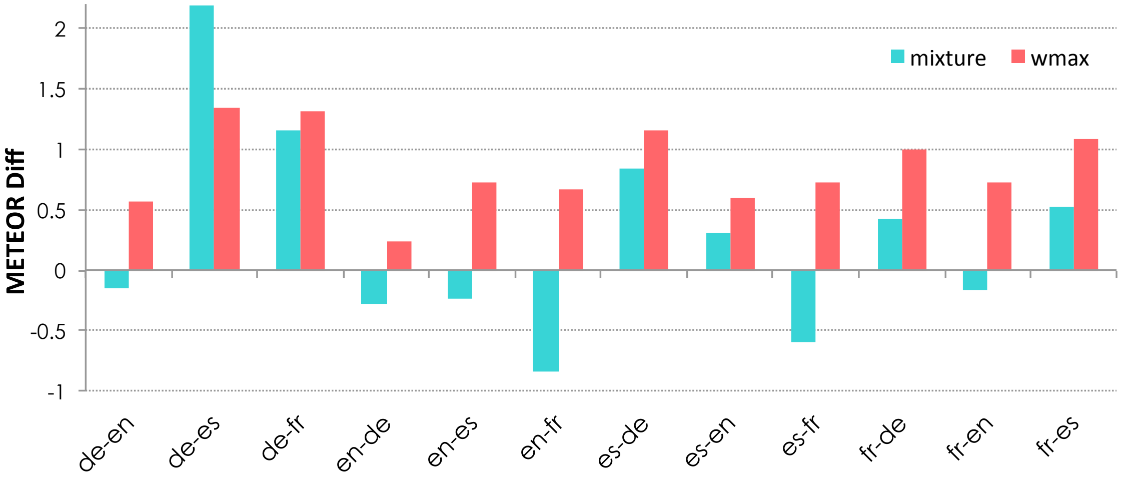
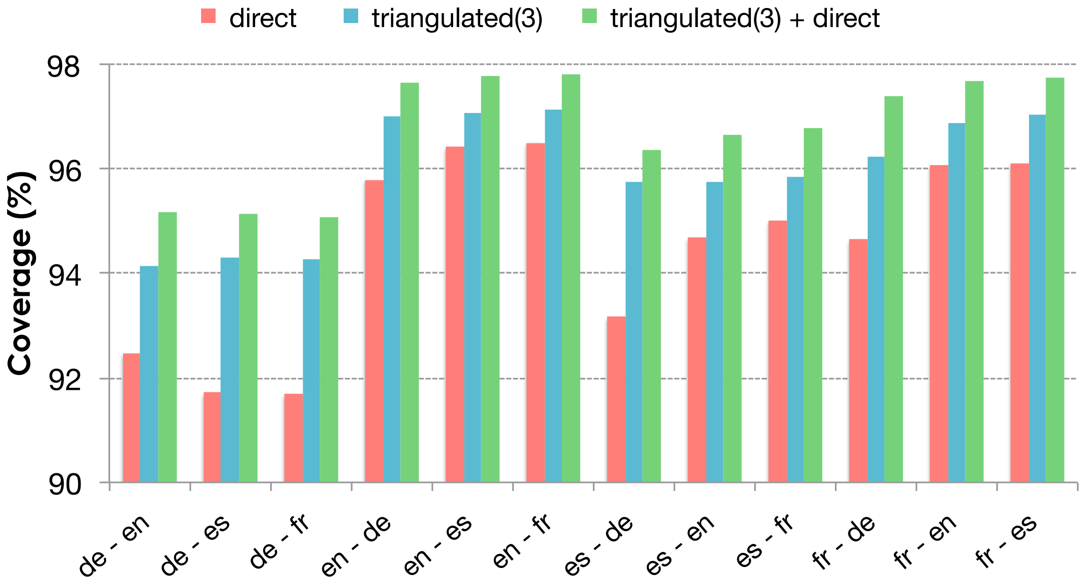


Figure 3.3: Meteor score difference between mixture models and direct systems as well as the difference between ensemble decoding approach with *wmax* and the direct system.

3.5.3 Phrase table coverage

Figure 3.4 shows the phrase-table coverage of the test set for different language pairs. The coverage is defined as the percentage of unigrams in the source side of the test set for which the corresponding phrase-table has translations for. The first set of bars shows the coverage of the direct systems and the second one shows that of the combined triangulated systems for three pivot languages. Finally, the last set of bars indicate the coverage when the direct phrase-table is combined with the triangulated ones. In all language pairs, the combined triangulated phrase-tables have a higher coverage compared to the direct phrase-tables. As expected, the coverage increases when these two phrase-tables are aggregated. The table



direct	478K	393K	403K	665K	1,084K	1,155K	479K	927K	1,319K	394K	743K	976K
tri+direct	83M	102M	132M	113M	103M	133M	129M	101M	152M	141M	109M	129M

Figure 3.4: Coverage for i) direct system; ii) combined triangulated system with three 3 languages; and iii) the combination of the triangulated phrase-tables and the direct one. The table shows the number of rules for each system and language pair after filtering based on the source side of the test set.

below the figure shows the number of rules for each system and language pair after filtering out based on the source side of the test set. This illustrates why running experiments on larger sizes of parallel data is prohibitive for hierarchical phrase-based models.

Choice of Pivot Language

Cohn and Lapata (2007) showed that the pivot language should be close to the source or the target language in order to be effective. For example, when translating between Romance languages (Italian, Spanish, etc.), the pivot language should also be a Romance language. In addition to those findings, based on the results presented in Table 3.2 here are some observations for these five European languages:

- When translating from or to *de*, *en* is the best pivot language;

- Generally *de* is not a suitable pivot language for any translation pair;
- When translating from *en* to any other language, *fr* is the best pivot;
- *it* is the best intermediate language when translating from *fr* or *es* to other languages; except when translating to *de* for which *en* is the best pivot language (c.f. first finding);

Figure 3.5 shows an example of how *ensemble decoding* is able to benefit from its component systems in translating different parts of a sentence.

SOURCE	les tâches de l ’ agence seront alors les suivantes : recherche scientifique indépendante , consultation , inspections et contrôles indépendants , ainsi qu ’ un système d ’ alerte rapide , mais sûrement aussi une tâche de coordination afin de restaurer la confiance .
REFERENCE	the tasks will then involve independent scientific research , advice , independent inspections and a rapid alert system , but definitely coordination too , in order to restore confidence .
DIRECT	the tasks of the agency , the following : independent scientific research , consultation , independent inspections and controls , and that a system of rapid alert , but also sûrement a task of coordination in order to restore confidence .
MIXTURE	the functions of the agency are the following : independent scientific research , consultation , independent inspections and controls , and that a system of early warning , but certainly also for the coordination in order to restoring confidence .
ENSEMBLE	the tasks of the agency will then be the following : independent scientific research , consultation , independent inspections and controls , and a rapid alert system , but can also a task of coordination in order to restore confidence .

Figure 3.5: Example translation output of the two baselines: direct and mixture and our approach: ensemble on *fr* → *en*.

3.6 Conclusion and Future Work

In the chapter, we introduced a novel approach for triangulation which does phrase-table triangulation and model combination on-the-fly in the decoder. Ensemble decoder uses the full hypothesis score for triangulation and combination and hence is able to mix hypotheses from heterogeneous systems.

Another advantage of this method to the phrase-table triangulation approach is that our method is applicable even when there exists no parallel data between source and target languages for tuning because we only use the *src-tgt* tuning set to optimize hyper-parameters, though phrase-table triangulation methods use it to learn MT log-linear feature weights for which having a tuning set is much more essential. Empirical results also showed that this method with *wmax* outperforms the baselines.

Future work includes imposing restrictions on the generated triangulated rules in order to keep only ones that have a strong support from the word alignments. By exploiting such constraints, we can experiment with larger sizes of parallel data. Specifically, a more natural experimental setup for triangulation which we would like to try is to use a small direct system with big *src* → *pvt* and *pvt* → *tgt* systems. This resembles the actual situation for resource-poor language pairs. We will also experiment with higher number of pivot languages.

Currently, most research in this area focuses on triangulation on paths containing only one pivot language. We can also analyze our method when using more languages in the triangulation chain and see whether there would any gain in doing such.

Finally, in current methods all $(\bar{f}, \bar{i})$ phrase pairs of the *src* → *pvt* systems, for which there does not exist any $(\bar{i}, \bar{e})$ pair in *pvt* → *tgt* are simply discarded. However in most cases, such $\bar{i}$ phrases can be segmented into smaller phrases (or rules for Hiero systems) to be triangulated via them. This segmentation is a decoding problem which requires an efficient algorithm to be practical.

src↓		**tgt →**	**en**	**es**	**fr**
de	**pivots**	**en**	–	15.94	13.62
		es	14.47	–	13.43
		fr	14.39	13.45	–
		it	14.14	14.90	11.67
	direct		21.94	20.70	17.37
	mixture		21.86	**22.30**	**18.28**
	wmax		22.49	21.32	18.22
	wsum		22.22	21.42	17.98
	switch		**22.59**	21.80	17.70

src↓		**tgt →**	**de**	**es**	**fr**
en	**pivots**	**de**	–	20.47	17.38
		es	12.95	–	20.78
		fr	14.09	23.25	–
		it	13.00	23.18	19.02
	direct		17.57	28.81	24.58
	mixture		**17.91**	28.89	24.30
	wmax		17.77	29.17	**25.39**
	wsum		17.68	**29.33**	24.70
	switch		17.77	29.32	24.98

src↓		**tgt →**	**de**	**en**	**fr**
es	**pivots**	**de**	–	18.84	23.28
		en	14.50	–	18.55
		fr	12.48	22.81	–
		it	13.69	23.14	23.44
	direct		16.30	28.11	29.83
	mixture		**17.75**	28.99	29.47
	wmax		17.34	**29.23**	**30.54**
	wsum		16.79	28.79	30.12
	switch		16.53	29.16	29.68

src↓		**tgt →**	**de**	**en**	**es**
fr	**pivots**	**de**	–	20.15	22.96
		en	14.84	–	27.84
		es	14.35	23.59	–
		it	14.08	24.08	30.38
	direct		16.56	28.79	35.27
	mixture		17.39	28.83	35.27
	wmax		**17.67**	**29.95**	**36.07**
	wsum		17.41	28.62	35.98
	switch		17.78	28.79	36.33

Table 3.2: Results of i) single-pivot triangulation; ii) baseline systems including direct systems and linear mixture of triangulated phrase-tables; iii) ensemble triangulation results based on different mixture operations. The mixture and ensemble methods are based on multi-pivot triangulation. These methods are built on 10k sentence-pair corpora.

Chapter 4

Stacking

We propose the use of *stacking*, an ensemble learning technique, to the statistical machine translation (SMT) models. A diverse ensemble of weak learners is created using the same SMT engine (a hierarchical phrase-based system) by manipulating the training data and a strong model is created by combining the weak models on-the-fly. Experimental results on two language pairs and three different sizes of training data show significant improvements of up to 4 BLEU points over a conventionally trained SMT model.

4.1 Introduction

Ensemble-based methods have been widely used in machine learning with the aim of reducing the instability of classifiers and regressors and/or their bias. The idea behind ensemble learning is to combine multiple models, *weak learners*, in an attempt to produce a *strong model* with less error. It has also been successfully applied to a wide variety of tasks in NLP (Tomeh et al., 2010; Surdeanu and Manning, 2010; F. T. Martins et al., 2008; Sang, 2002) and recently has attracted attention in the statistical machine translation community in various work (Xiao et al., 2013; Song et al., 2011; Xiao et al., 2010; Lagarda and Casacuberta, 2008).

In this chapter, we propose a method to adopt *stacking* (Wolpert, 1992), an ensemble learning technique, to SMT. We manipulate the full set of training data, creating k disjoint sets of *held-out* and *held-in* data sets as in k-fold cross-validation and build a model on each partition. This creates a diverse ensemble of statistical machine translation models where each member of the ensemble has different feature function values for the SMT log-linear model (Koehn, 2010). The weights of these models are then tuned using minimum error rate training (Och, 2003a) on the *held-out* fold to provide k weak models. We then create a strong model by stacking another meta-learner on top of weak models to combine them into a single model. The particular second-tier model we use is the ensemble decoding approach (Chapter 2) which combines hypotheses from the weak models on-the-fly in the decoder.

Using this approach, we take advantage of the diversity created by manipulating the training data and obtain a significant and consistent improvement over a conventionally trained SMT model with a fixed training and tuning set.

4.2 Ensemble Learning Methods

In the machine learning literature, ensemble learning methods combine the predictive power of multiple models to achieve a better performance compared to any of the constituent models.

Two well-known instances of general framework of ensemble learning are *bagging* and *boosting*. Bagging (Breiman, 1996a) (i.e. bootstrap aggregating) takes a number of samples with replacement from a training set. The generated sample set may have 0, 1 or more instances of each original training instance. This procedure is repeated a number of times and the base learner is applied to each sample to produce a weak learner. These models are aggregated by doing a uniform voting for classification or averaging the predictions for regression. Bagging reduces the variance of the base model while leaving the bias relatively unchanged and is most useful when a small change in the training data affects the prediction of the model (i.e. the model is unstable) (Breiman, 1996a). Bagging has been recently applied to SMT (Xiao et al., 2013; Song et al., 2011)

Boosting (Schapire, 1990) constructs a strong learner by repeatedly choosing a weak learner and applying it on a re-weighted training set. In each iteration, a weak model is learned on the training data, whose instance weights are modified from the previous iteration to concentrate on examples on which the model predictions were poor. By putting more weight on the wrongly predicted examples, a diverse ensemble of weak learners is created. Boosting has also been used in SMT (Xiao et al., 2013; Xiao et al., 2010; Lagarda and Casacuberta, 2008).

Stacking (or stacked generalization) (Wolpert, 1992) is another ensemble learning algorithm that uses a second-level learning algorithm on top of the base learners to reduce the bias. The first level consists of predictors $g_1, \ldots, g_k$ where $g_i : \mathbb{R}^d \to \mathbb{R}$, receiving input $x \in \mathbb{R}^d$ and producing a prediction $g_i(x)$. The next level consists of a single function $h : \mathbb{R}^{d+k} \to \mathbb{R}$ that takes $\langle x, g_1(x), \ldots, g_k(x) \rangle$ as input and produces an ensemble prediction $\hat{y} = h(x, g_1(x), \ldots, g_k(x))$.

Two categories of ensemble learning are *homogeneous learning* and *heterogeneous learning*. In homogeneous learning, a single base learner is used, and diversity is generated by data sampling, feature sampling, randomization and parameter settings, among other strategies. In heterogeneous learning different learning algorithms are applied to the same training data to create a pool of diverse models. In this chapter, we focus on homogeneous ensemble learning by manipulating the training data.

In the primary form of stacking (Wolpert, 1992), the training data is split into multiple disjoint sets of *held-out* and *held-in* data sets using k-fold cross-validation and k models are trained on the held-in partitions and run on held-out partitions. Then a meta-learner uses the predictions of all models on their held-out sets and the actual labels to learn a final model. The details of the first-layer and second-layer predictors are considered to be a “black art” (Wolpert, 1992).

Breiman (1996b) linearly combines the weak learners in the stacking framework. The weights of the base learners are learned using ridge regression: $s(x) = \sum_k \alpha_k m_k(x)$, where m_k is a base model trained on the k-th partition of the data and s is the resulting strong model created by linearly interpolating the weak learners.

Stacking (aka blending) has been used in the system that won the Netflix Prize[1], which used a multi-level stacking algorithm.

Stacking has been actively used in statistical parsing: Nivre and McDonald (2008) integrated two models for dependency parsing by letting one model learn from features generated by the other; F. T. Martins et al. (2008) further formalized the stacking algorithm and improved on Nivre and McDonald (2008); Surdeanu and Manning (2010) includes a detailed analysis of ensemble models for statistical parsing: *i)* the diversity of base parsers is more important than the complexity of the models; *ii)* unweighted voting performs as well as weighted voting; and *iii)* ensemble models that combine at decoding time significantly outperform models that combine multiple models at training time.

[1] http://www.netflixprize.com/

Algorithm 1: Stacking for SMT

```
Input: $\mathcal{D} = \{\langle f_j, e_j \rangle\}_{j=1}^{N}$        ▷ A parallel corpus
Input: $k$        ▷ # of folds (i.e. weak learners)
Output: StrongModel $s$
$\mathcal{D}^1, \ldots, \mathcal{D}^k \leftarrow$ Split($\mathcal{D}$, $k$)
for $i = 1 \rightarrow k$ do
    $\mathcal{T}^i \leftarrow \mathcal{D} - \mathcal{D}^i$        ▷ Use all but current partition as training set.
    $\boldsymbol{\phi}_i \leftarrow$ Train($\mathcal{T}^i$)        ▷ Train feature functions.
    $\mathcal{M}_i \leftarrow$ Tune($\boldsymbol{\phi}_i$, $\mathcal{D}^i$)        ▷ Tune the model on the current partition.
end for
s $\leftarrow$ CombineModels($\mathcal{M}_1$ , …, $\mathcal{M}_k$)        ▷ Combine all the base models to produce a strong stacked model.
```

4.3 Our Approach

In this chapter, we propose a method to apply stacking to statistical machine translation (SMT) and our method is the first to successfully exploit stacking for statistical machine translation. We use a standard statistical machine translation engine and produce multiple diverse models by partitioning the training set using the k-fold cross-validation technique. A diverse ensemble of weak systems is created by learning a model on each $k-1$ fold and tuning the statistical machine translation log-linear weights on the remaining fold. However, instead of learning a model on the output of base models as in (Wolpert, 1992), we combine hypotheses from the base models in the decoder with uniform weights (Algorithm 1).

For the base learner, we use Kriya (Sankaran et al., 2012), an in-house hierarchical phrase-based machine translation system, to produce multiple weak models. These models are combined together using the ensemble decoding approach (discussed in Chapter 2) to produce a strong model in the decoder.

4.4 Experiments & Results

We experimented with two language pairs: French to English and Spanish to English on the *Europarl* corpus (v7) (Koehn, 2005) and used ACL/WMT 2005 [2] data for dev and test

[2] http://www.statmt.org/wpt05/mt-shared-task/

	Train size	**Src tokens**	**Tgt tokens**
Fr - En	0+dev	67K	58K
	10k+dev	365K	327K
	100k+dev	3M	2.8M
Es - En	0+dev	60K	58K
	10k+dev	341K	326K
	100k+dev	2.9M	2.8M

Table 4.1: Statistics of the training set for different systems and different language pairs.

Direction	**k-fold**	**Resub**	**Mean**	**wsum**	**wmax**	**prod**	**sw:max**	**sw:sum**
Fr - En	2	18.08	19.67	22.32	**22.48**	22.06	21.70	21.81
	4	18.08	21.80	23.14	23.48	**23.55**	22.83	22.95
	8	18.08	22.47	23.76	23.75	**23.78**	23.02	23.47
Es - En	2	18.61	19.23	**21.62**	21.33	21.49	21.48	21.51
	4	18.61	21.52	23.42	22.81	**22.91**	22.81	**22.92**
	8	18.61	22.20	23.69	**23.89**	23.51	22.92	23.26

Table 4.2: Test set BLEU scores when applying stacking on the dev set only (using no specific training set).

sets.

The base models are built using Kriya (Sankaran et al., 2012), which uses the same features mentioned in (Chiang, 2005a): forward and backward relative-frequency and lexical TM probabilities; LM; word, phrase and glue-rules penalty. GIZA++ (Och and Ney, 2003) has been used for word alignment with phrase length limit of 10. Feature weights were optimized using MERT (Och, 2003a). We built a 5-gram language model on the English side of Europarl and used the Kneser-Ney smoothing method and SRILM (Stolcke, 2002a) as the language model toolkit.

4.4.1 Training on dev set

We first consider the scenario in which there is no parallel data between a language pair except a small bi-text used as a dev set (2k sentence pairs, see Table 4.1). We use no specific training data and construct a SMT system completely on the dev set by using our approach and compare to two different baselines. A natural baseline when having a limited parallel text is to do re-substitution validation where the model is trained on the whole dev set and

Corpus	k-fold	Baseline	bma	wsum	wmax	prod	sw:max	sw:sum
10k+dev	6	28.75	29.49	**29.87**	29.78	29.21	29.69	29.59
100k+dev	11 / 51	29.53	29.75	34.00	**34.07**	33.11	34.05	33.96

Table 4.3: Test set BLEU scores for *fr* → *en* when using 10k and 100k sentence training sets along with the dev set.

Corpus	k-fold	Baseline	bma	wsum	wmax	prod	sw:max	sw:sum
10k+dev	6	28.21	28.76	**29.59**	29.51	29.15	29.10	29.21
100k+dev	11 / 51	33.25	33.44	**34.21**	34.00	33.17	34.19	**34.22**

Table 4.4: Test set BLEU scores for *es* → *en* when using 10k and 100k sentence training sets along with the dev set.

is tuned on the same set. This validation process suffers seriously from over-fitting. The second baseline is the mean of BLEU scores of all base models.

Table 4.2 summarizes the BLEU scores on the test set when using stacking only on the dev set on two different language pairs. As the table shows, increasing the number of folds results in higher BLEU scores. However, doing such will generally lead to higher variance among base learners.

Figure 4.1 shows the BLEU score of each of the base models resulted from a 20-fold partitioning of the dev set along with the strong models' BLEU scores. As the figure shows, the strong models are generally superior to the base models whose mean is represented as a horizontal line.

4.4.2 Training on train+dev

When we have some training data, we can use the cross-validation-style partitioning to create k splits. We then train a system on $k-1$ folds and tune on the dev set. However, each system eventually wastes a fold of the training data. In order to take advantage of that remaining fold, we concatenate the dev set to the training set and partition the whole union. In this way, we use all data available to us. We experimented with two sizes of training data: 10k sentence pairs and 100k, that with the addition of the dev set, we have 12k and 102k sentence-pair corpora.

Table 4.1 summarizes statistics of the data sets used in all scenarios. Tables 4.3 and

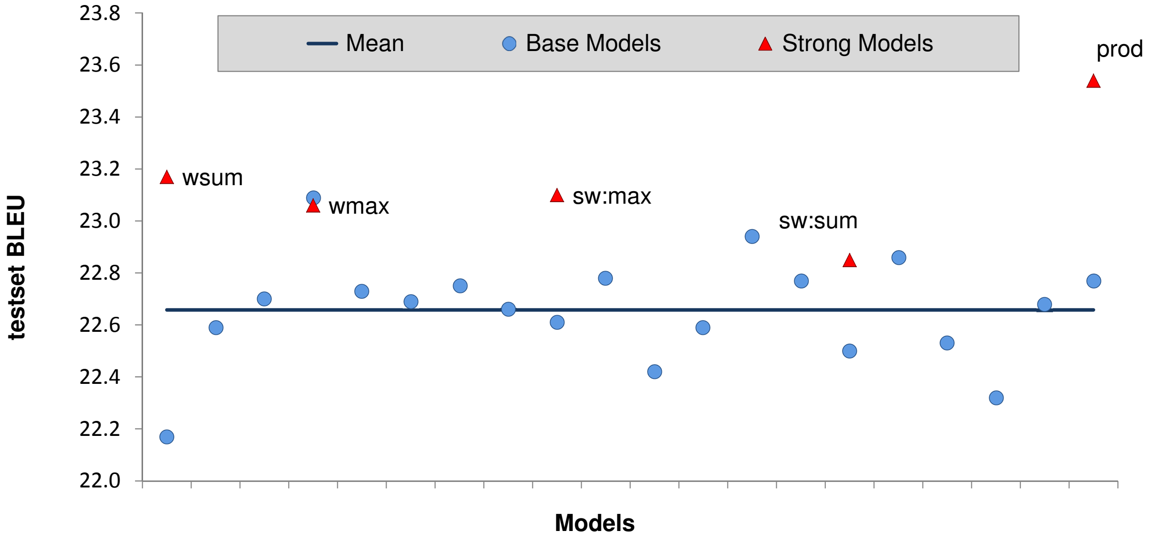


Figure 4.1: BLEU scores for all the base models and stacked models on the *fr* → *en* dev set with 20-fold cross validation. The horizontal line shows the mean of base models' scores.

4.4 reports the BLEU scores when using stacking on these two corpus sizes for *fr* → *en* and *es* → *en* respectively. The baselines are the conventional systems which are built on the training-set only and tuned on the dev set as well as *Bayesian Model Averaging* (BMA, see §4.5). For the 100k+dev corpus, we sampled 11 partitions from all 51 possible partitions by taking every fifth partition as training data. The results in Table 4.3 show that stacking can improve over the baseline BLEU scores by up to 4 points.

Examining the performance of the different mixture operations, we can see that WSUM and WMAX typically outperform other mixture operations. Different mixture operations can be dominant in different language pairs and different sizes of training sets.

4.4.3 Partitioning Methods

The improvements shown in the previous sections are due to three factors: i) using stacking that reduces the bias of the models and has been shown useful in many different applications; ii) taking advantage of the dev set in the training of $k-1$ base models; and iii) having different tuning sets for each base model.

Method	Mean	wsum	wmax	prod	sw:max	sw:sum
sampling with replacement	27.08	28.77	28.23	27.73	28.23	28.08
sampling without replacement	28.03	29.20	28.88	28.08	27.97	28.20
cross validation (fixed tuning set)	28.10	29.17	28.81	28.46	28.10	28.10
cross validation (this work)	28.92	29.87	29.78	29.21	29.69	29.59

Figure 4.2: Results when using sampling without or without replacement and cross validation when the tuning set is fixed or changing (our approach) on the *fr* → *en* 10k sentence-pair data. *Mean* refers to the mean of the BLEU scores of the base models in each partitioning method.

Table 4.2 compares a number of ways for partitioning the training set. The first two rows refer to sampling with and without replacement. As the results indicate, sampling without replacement has an obvious advantage over sampling with replacement. The third row shows the results when a cross validation method has been used without using the dev set in the partitioning. In other words, in this method the dev set has been set aside for tuning the model parameters only. In order to introduce some diversity to the base models, each model gives up 20% of the training data. Cross validation with fixed tuning set gets similar results to sampling without substitution. Finally, the last row shows the result when we take advantage of the dev set and base models are tuned on different sets. Since the base models in this method use the whole training data, the base models' mean score (i.e. 28.92) is close to that of the conventional systems (i.e. 28.75) as opposed to the three other methods and combining these base models can yield an improvement of up to 1.12 BLEU points over the baseline.

4.4.4 Decoding Time Overhead

In this chapter, we showed how we can take advantage of diversities of different models to create a stronger model. This section discusses the time overhead of stacking compared to the conventional systems. Clearly, the training time grows linearly with the number of models since the models are trained and tuned separately. Fortunately, the training and tuning can be done completely simultaneously. It is also possible to modify the SMT word-alignment, phrase-extraction and scoring modules to avoid the computation redundancy in this process. In other words, given that each training sentence contributes to $k - 1$

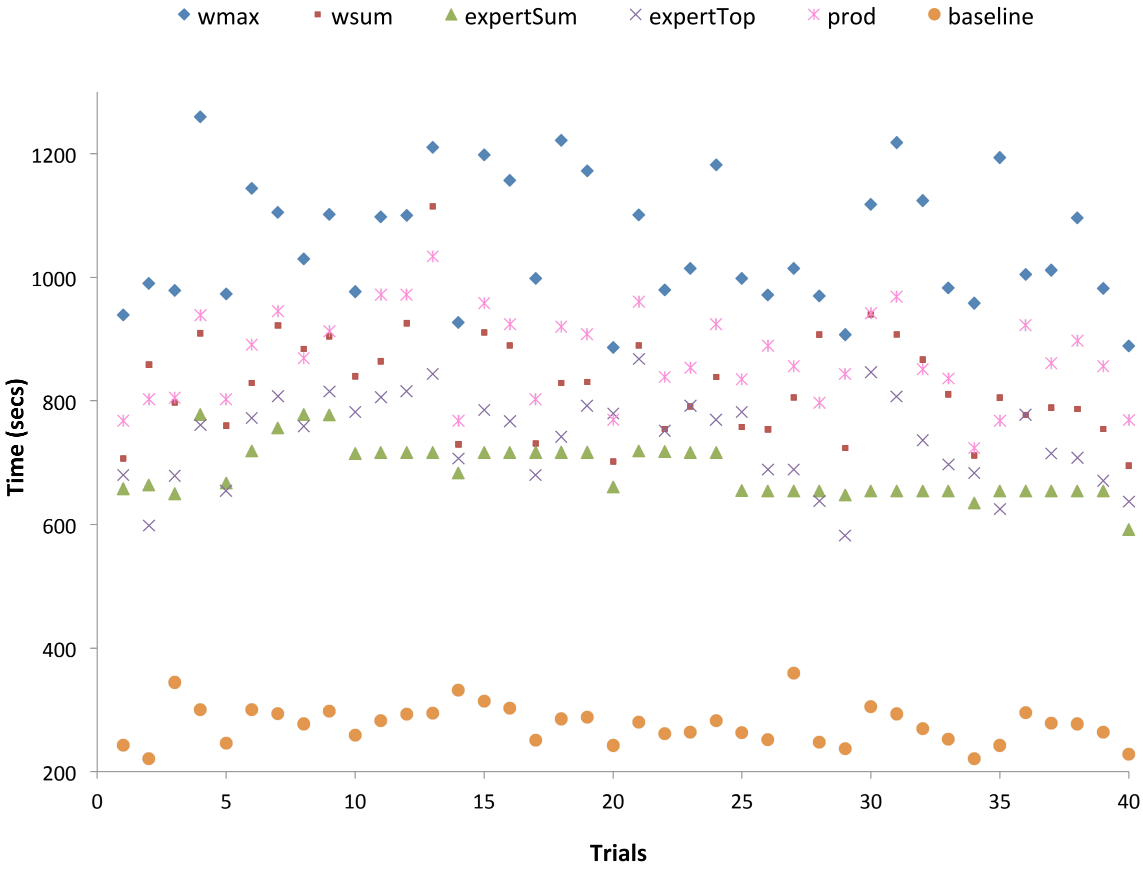


Figure 4.3: Time comparison between different mixture operations of ensemble decoding where 11 base models are combined.

base models in a *k*-fold cross validation, more intelligent methods can be used to do the computations for each sentence only once, but to use them as many times as required. However, this has been left for future work and in this work, we simply parallelize the training and tuning steps having access to cluster machines.

Figure 4.3 compares the decoding time for 40 equal-size subsets of the original dev set when mixing 11 base models using different mixture operations. As the figure shows, the decoding time does not grow linearly with the number of base models.

Table 4.5 summarizes the decoding time for each mixture operation when 11 models are combined and its ratio to that of the baseline model.

operation	time(s)	time ratio
baseline	275	1
sw:sum	687	2.49
sw:max	737	2.67
wsum	825	2.99
prod	884	3.20
wmax	1054	3.82

Table 4.5: Time overhead of stacking 11 base models over the baseline.

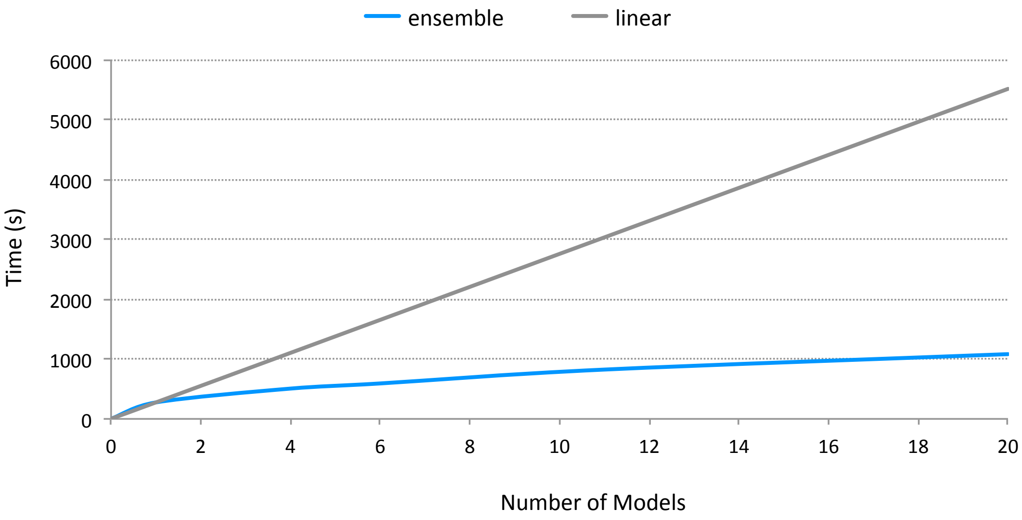


Figure 4.4: Stacking time ratio with regard to the number of base models.

Figure 4.4 illustrates how decoding time increases as the number of base models changes. The time values are based on the *wsum* mixture operation and the base models are trained on the same amount of training data. As the figure shows, the ensemble decoding time complexity is sub-linear in the number of base models. The decoding time can still be drastically improved by caching the combined hypotheses for later uses, however at the expense of more memory usage.

4.5 Related Work

Xiao et al. (2013) have applied both boosting and bagging on three different statistical machine translation engines: phrase-based (Koehn et al., 2003), hierarchical phrase-based (Chiang, 2005a) and syntax-based (Galley et al., 2006) and showed SMT can benefit from these methods as well.

Duan et al. (2009) creates an ensemble of models by using feature subspace method in the machine learning literature (Ho, 1998). Each member of the ensemble is built by removing one non-LM feature in the log-linear framework or varying the order of language model. Finally they use a sentence-level system combination on the outputs of the base models to pick the best system for each sentence. Though, they do not combine the hypotheses search spaces of individual base models.

Our work is most similar to that of Duan et al. (2010) which uses *Bayesian model averaging* (BMA) (Hoeting et al., 1999) for SMT. They used sampling without replacement to create a number of base models whose phrase-tables are combined with that of the baseline (trained on the full training-set) using linear mixture models (Foster and Kuhn, 2007).

Our approach differs from this approach in a number of ways: *i)* we use cross-validation-style partitioning for creating training subsets while they do sampling without replacement (80% of the training set); *ii)* in our approach a number of base models are trained and tuned and they are combined on-the-fly in the decoder using *ensemble decoding* which has been shown to be more effective than offline combination of phrase-table-only features; *iii)* in Duan et al. (2010)'s method, each system gives up 20% of the training data in exchange for more diversity, but in contrast, our method not only uses all available data for training, but promotes diversity through allowing each model to tune on a different data set; *iv)* our approach takes advantage of held out data (the tuning set) in the training of base models which is beneficial especially when little parallel data is available or tuning/test sets and training sets are from different domains.

Empirical results (Tables 4.3 and 4.4) also show that our approach outperforms the Bayesian model averaging approach (BMA).

4.6 Conclusion

In this chapter, we proposed a novel method of applying stacking to the statistical machine translation task. The results when using no, 10k and 100k sentence-pair training sets (along with a development set for tuning) show that stacking can yield an improvement of up to 4 BLEU points over conventionally trained SMT models which use a fixed training and tuning set.

Future work includes experimenting with larger training sets to investigate how useful this approach can be when having larger sizes of training data and how each mixture operation behaves when the training data enlarges. In addition, implementing efficient algorithms for reducing the training time by keeping track of each sentence's computations for different phases of training, e.g. word alignment, phrase extraction and phrase scoring and reusing them is also left as future work.

Chapter 5

Graph Propagation for Paraphrasing Out-of-Vocabulary Words

Out-of-vocabulary (oov) words or phrases still remain a challenge in statistical machine translation especially when a limited amount of parallel text is available for training or when there is a domain shift from training data to test data. In this chapter, we propose a novel approach to finding translations for oov words. We induce a lexicon by constructing a graph on source language monolingual text and employ a graph propagation technique in order to find translations for all the source language phrases. Our method differs from previous approaches by adopting a graph propagation approach that takes into account not only one-step (from oov directly to a source language phrase that has a translation) but multi-step paraphrases from oov source language words to other source language phrases and eventually to target language translations. Experimental results show that our graph propagation method significantly improves performance over two strong baselines under intrinsic and extrinsic evaluation metrics.

5.1 Introduction

Out-of-vocabulary (oov) words or phrases still remain a challenge in statistical machine translation. SMT systems usually copy unknown words verbatim to the target language output. Although this is helpful for languages with same writing systems in translating a small fraction of oovs such as named entities, it harms the translation in other types of oovs and distant language pairs. In general, copied-over oovs are a hindrance to fluent, high quality translation, and we can see evidence of this in automatic measures such as BLEU (Papineni et al., 2002b) and also in human evaluation scores such as HTER. The problem becomes more severe when only a limited amount of parallel text is available for training or when the training and test data are from different domains. Even noisy translation of oovs can aid the language model to better re-order the words in the target language (Zhang et al., 2012).

Increasing the size of the parallel data can reduce the number of oovs. However, there will always be some words or phrases that are new to the system and finding ways to translate such words or phrases will be beneficial to the system. Researchers have applied a number of approaches to tackle this problem. Some approaches use pivot languages (Callison-Burch et al., 2006) while others use lexicon-induction-based approaches from source language mono-lingual corpora (Koehn and Knight, 2002; Garera et al., 2009; Marton et al., 2009).

Pivot language techniques tackle this problem by taking advantage of available parallel data between the source language and a third language (see Chapter 3). Using a pivot language, oovs are translated into a third language and back into the source language and thereby paraphrases to those oov words are extracted (Callison-Burch et al., 2006). For each oov, the system can be augmented by aggregating the translations of all its paraphrases and assign them to the oov. However, these methods require parallel corpora between the source language and one or multiple pivot languages.

Another line of work exploits spelling and morphological variants of oov words. Habash (2008) presents techniques for online handling of oov words for Arabic to English such as spelling expansion and morphological expansion. Huang et al. (2011) proposes a method to combine sublexical/constituent translations of an oov word or phrase to generate its translations.

Several researchers have applied lexicon-induction methods to create a bilingual lexicon for those oovs. Marton et al. (2009) use a mono-lingual text on the source side to find paraphrases to oov words for which the translations are available. The translations for these paraphrases are then used as the translations of the oov word. These methods are based on the *distributional hypothesis* which states that words appearing in the same contexts tend to have similar meaning (Harris, 1954). Marton et al. (2009) showed that this method improves over the baseline system where oovs are untranslated.

We propose a graph propagation-based extension to the approach of Marton et al. (2009) in which a graph is constructed from source language monolingual text[1] and the source-side of the available parallel data. Nodes that have related meanings are connected together and nodes for which we have translations in the phrase-table are annotated with target-side translations and their feature values. A graph propagation algorithm is then used to propagate translations from labeled nodes to unlabeled nodes (phrases appearing only in the monolingual text and oovs). This provides a general purpose approach to handling several types of oovs, including morphological variants, spelling variants and synonyms[2].

Constructing such a huge graph and propagating messages through it pose severe computational challenges. Throughout the chapter, we will see how these challenges are dealt with using scalable algorithms.

5.2 Collocational Lexicon Induction

Rapp (1995) introduced the notion of a distributional profile in bilingual lexicon induction from monolingual data. A *distributional profile* (DP) of a word or phrase type is a co-occurrence vector created by combining all co-occurrence vectors of the tokens of that phrase type. Each distributional profile can be seen as a point in a $|V|$-dimensional space where V is the vocabulary where each word type represents a unique axis. Points (i.e. phrase types) that are close to one another in this high-dimensional space can represent paraphrases. This approach has also been used in machine translation to find in-vocabulary paraphrases for oov words on the source side and find a way to translate them.

5.2.1 Baseline System

Marton et al. (2009) was the first to successfully integrate a collocational approach to finding translations for oov words into an end-to-end SMT system. We explain their method in detail as we will compare against this approach. The method relies on monolingual distributional profiles (DPs) which are numerical vectors representing the context around each word. The goal is to find words or phrases that appear in similar contexts as the oovs. For each oov

[1] Here on by monolingual data we always mean monolingual data on the source language

[2] Named entity oovs may be handled properly by copying or transliteration.

a distributional profile is created by collecting all words appearing in a fixed distance from all occurrences of the oov word in the monolingual text. These co-occurrence counts are converted to an association measure (Section 5.2.2) that encodes the relatedness of each pair of words or phrases.

Then, the most similar phrases to each oov are found by measuring the similarity of their DPs to that of the oov word. Marton et al. (2009) uses a heuristic to prune the search space for finding candidate paraphrases by keeping the surrounding context (e.g. $L\_\_R$) of each occurrences of the oov word. All phrases that appear in any of such contexts are collected as candidate paraphrases. For each of these paraphrases, a DP is constructed and compared to that of the oov word using a similarity measure (Section 5.2.2).

The top-k paraphrases that have translations in the phrase-table are used to assign translations and scores to each oov word by marginalizing translations over paraphrases:

$$p(t|o) = \sum_{s} p(t|s)p(s|o)$$

where t is a phrase on the target side, o is the oov word or phrase, and s is a paraphrase of o. $p(s|o)$ is estimated using a similarity measure over DPs and $p(t|s)$ is coming from the phrase-table.

We reimplemented this collocational approach for finding translations for oovs and used it as a baseline system.

Alternative ways of modeling and comparing distributional profiles have been proposed (Rapp, 1999; Fung and Yee, 1998; Terra and Clarke, 2003; Garera et al., 2009; Marton et al., 2009). We review some of them here and compare their performance in Section 5.4.4.

5.2.2 Association Measures

Given a word u, its distributional profile $DP(u)$ is constructed by counting surrounding words (in a fixed window size) in a monolingual corpus.

$$DP(u) = \{\langle A(u, w_i)\rangle \mid w_i \in V\}$$

The counts can be collected in positional[3] (Rapp, 1999) or non-positional way (count all the word occurrences within the sliding window). $A(\cdot, \cdot)$ is an association measure and

[3] e.g., position 1 is the word immediately after, position -1 is the word immediately before, etc.

can simply be defined as co-occurrence counts within sliding windows. Stronger association measures can also be used such as:

Conditional probability: the probability for the occurrence of each word in DP given the occurrence of u (Schütze and Pedersen, 1997):

$$\text{CP}(u, w_i) = P(w_i|u)$$

Pointwise Mutual Information: this measure is a transformation of the independence assumption into a ratio. Positive values indicate that words co-occur more than what we expect under the independence assumption (Lin, 1998):

$$\text{PMI}(u, w_i) = log_2 \frac{P(u, w_i)}{P(u)P(w_i)}$$

Likelihood ratio: Dunning (1993) uses the likelihood ratio for word similarity:

$$\lambda(u, w_i) = \frac{L(P(w_i|u); p) * L(P(w_i|\neg u); p)}{L(P(w_i|u); p_1) * L(P(w_i|\neg u); p_2)}$$

where L is likelihood function under the assumption that word counts in text have binomial distributions. The numerator represents the likelihood of the hypothesis that u and w_i are independent ($P(w_i|u) = P(w_i|\neg u) = p$) and the denominator represents the likelihood of the hypothesis that u and w_i are dependent ($P(w_i|u) \neq P(w_i|\neg u)$, $P(w_i|u) = p_1$, $P(w_i|\neg u) = p_2$)[4].

Chi-square test is a statistical hypothesis testing method to evaluate independence of two categorical random variables, e.g. whether the *occurrence* of u and w_i (denoted by x and y respectively) are independent. The test statistics $\chi^2(u, w_i)$ is the deviation of the observed counts $f_{x,y}$ from their expected values $E_{x,y}$:

$$\chi^2(u, w_i) := \sum_{x \in \{w_i, \neg w_i\}} \sum_{y \in \{u, \neg u\}} \frac{(f_{x,y} - E_{x,y})^2}{E_{x,y}}$$

[4] Binomial distribution $B(k; n, \theta)$ gives the probability of observing k heads in n tosses of a coin where the coin parameter is θ. In our context, p, p_1 and p_2 are parameters of Binomial distributions estimated using maximum likelihood.

5.2.3 Similarity Measures

Various functions have been used to estimate the similarity between distributional profiles. Given two distributional profiles $DP(u)$ and $DP(v)$, some similarity functions can be defined as follows. Note that $A(\cdot,\cdot)$ stands for the various association measures defined in Sec. 5.2.2.

Cosine coefficient is the cosine of the angle between two vectors $DP(u)$ and $DP(v)$:

$$cos(DP(u), DP(v)) = \frac{\sum_{w_i \in V} A(u, w_i)A(v, w_i)}{\sqrt{\sum_{w_i \in V} A(u, w_i)^2}\sqrt{\sum_{w_i \in V} A(v, w_i)^2}}$$

L_1-Norm computes the accumulated distance between entries of two distributional profiles ($L_1(\cdot,\cdot)$). It has been used as word similarity measure in language modeling (Dagan et al., 1999).

$$L_1(DP(u), DP(v)) = \sum_{w_i \in V} |A(u, w_i) - A(v, w_i)|$$

Jensen-Shannon Divergence is a symmetric version of *contextual average mutual information* (KL) which is used by Dagan et al. (1999) as word similarity measure.

$$JSD(DP(u), DP(v)) = KL(DP(u), AVG_{DP}(u, v)) + KL(DP(v), AVG_{DP}(u, v))$$

$$AVG_{DP}(u, v) = \left\{\frac{A(u, w_i) + A(v, w_i)}{2} \mid w_i \in V\right\}$$

$$KL(DP(u), DP(v)) = \sum_{w_i \in V} A(u, w_i) log \frac{A(u, w_i)}{A(v, w_i)}$$

5.3 Graph-based Lexicon Induction

We propose a novel approach to alleviate the oov problem. Given a (possibly small amount of) parallel data between the source and target languages, and a large monolingual data in the source language, we construct a graph over all phrase types in the monolingual text and the source side of the parallel corpus and connect phrases that have similar meanings (i.e. appear in similar context) to one another. To do so, the distributional profiles of all source phrase types are created. Each phrase type represents a vertex in the graph and is connected to other vertices with a weight defined by a similarity measure between the two profiles (Section 5.2.3). There are three types of vertices in the graph:

1. **labeled nodes** which appear in the parallel corpus and for which we have the target-side translations[5];
2. **oov nodes** from the *dev/test set* for which we seek labels (translations); and
3. **unlabeled nodes** words or phrases from the *monolingual data* which appear usually between oov nodes and labeled nodes. When a relatively small parallel data is used, unlabeled nodes outnumber labeled ones and many of them lie on the paths between an oov node to labeled ones.

Marton et al. (2009)'s approach ignores these bridging nodes and connects each oov node to the k-nearest *labeled* nodes. One may argue that these unlabeled nodes do not play a major role in the graph and the labels will eventually get to the oov nodes from the labeled nodes by directly connecting them. However based on the definition of the similarity measures using context, it is quite possible that an oov node and a labeled node which are connected to the same unlabeled node do not share any context words and hence are not directly connected. For instance, consider three nodes, u (unlabeled), o (oov) and l (labeled) where u has the same left context words with o but share the right context with l. o and l are not connected since they do not share any context word.

Once a graph is constructed based on similarities of phrases, graph propagation is used to propagate the labels from labeled nodes to unlabeled and oov nodes. The approach is based on the *smoothness assumption* (Chapelle et al., 2006) which states if two nodes are similar according to the graph, then their output labels should also be similar.

The baseline approach (Marton et al., 2009) can be formulated as a *bipartite graph* with two types of nodes: labeled nodes (L) and oov nodes (O). Each oov node is connected to a number of labeled nodes, and vice versa and there is no edge between nodes of the same type. In such a graph, the similarity of each pair of nodes is computed using one of the similarity measures discussed above. The labels are translations and their probabilities (more specifically $p(e|f)$) from the phrase-table extracted from the parallel corpus. Translations

[5] It is possible that a word/phrase appears in the parallel corpus, but not in the phrase-table. This happens when the word-alignment module is not able to align it to a target-side word.

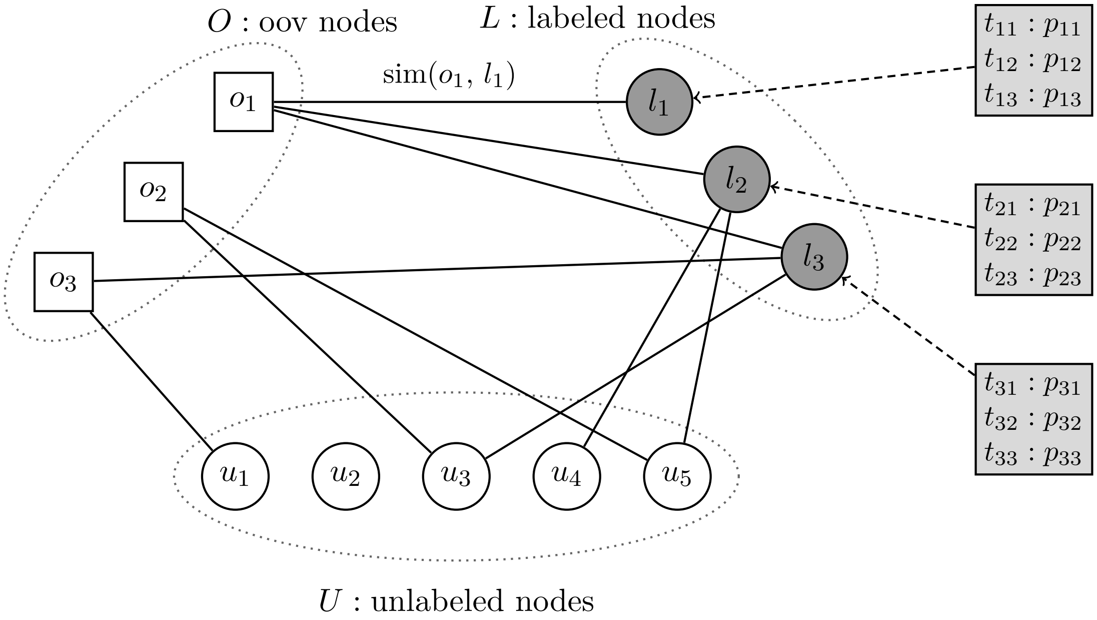


Figure 5.1: A tripartite graph between oov, labeled and unlabeled nodes. Translations propagate either directly from labeled nodes to oov nodes or indirectly via unlabeled nodes.

get propagated to oov nodes using a label propagation technique. However beside the difference in the oov label assignment, there is a major difference between our bipartite graph and the baseline (Marton et al., 2009): we do not use a heuristic to reduce the number of neighbor candidates and we consider all possible candidates that share at least one context word. This makes a significant difference in practice as shown in Section 5.4.4.

We also take advantage of unlabeled nodes to help connect oov nodes to labeled ones. The discussed bipartite graph can easily be expanded to a *tripartite graph* by adding unlabeled nodes. Figure 5.1 illustrate a tripartite graph in which unlabeled nodes are connected to both labeled and oov nodes. Again, there is no edge between nodes of the same type. We also created the *full graph* where all nodes can be freely connected to nodes of any type including the same type. However, constructing such graph and doing graph propagation on it is computationally very expensive for large n-grams.

5.3.1 Label Propagation

Let $G = (V, E, W)$ be a graph where V is the set of vertices, E is the set of edges, and W is the edge weight matrix. The vertex set V consists of labeled V_L and unlabeled V_U nodes, and the goal of the labeling propagation algorithm is to compute soft labels for unlabeled vertices from the labeled vertices. Intuitively, the edge weight $W(u, v)$ encodes the degree of our belief about the similarity of the soft labeling for nodes u and v. A soft label $\hat{Y}_v \in \Delta^{m+1}$ is a probability vector in $(m + 1)$-dimensional simplex, where m is the number of possible labels and the additional dimension accounts for the *undefined* $\perp$ label[6].

In this method, we make use of the *modified Adsorption* (MAD) algorithm (Talukdar and Crammer, 2009) which finds soft label vectors $\hat{Y}_v$ to solve the following unconstrained optimization problem:

$$\begin{aligned} \min_{\hat{Y}} \quad & \mu_1 \sum_{v \in V_L} p_{1,v} ||Y_v - \hat{Y}_v||_2^2 + && (5.1) \\ & \mu_2 \sum_{v,u} p_{2,v} W_{v,u} ||\hat{Y}_v - \hat{Y}_u||_2^2 + && (5.2) \\ & \mu_3 \sum_{v} ||\hat{Y}_v - p_{3,v}\, R_v||_2^2 && (5.3) \end{aligned}$$

where μ_i and $p_{i,v}$ are hyper-parameters. $p_{1,v}, p_{2,v}$ and $p_{3,v}$ are *inject, continue* and *abondon* probabilities respectively for node v and $(\forall v : \sum_i p_{i,v} = 1)$[7], and $R_v \in \Delta^{m+1}$ encodes our prior belief about the labeling of a node v. The first term (5.1) enforces the labeling of the algorithm to match the seed labeling Y_v with different extent for different labeled nodes. The second term (5.2) enforces the *smoothness* of the labeling according to the graph structure and edge weights. The last term (5.3) regularizes the soft labeling for a vertex v to match a priori label R_v, e.g. for high-degree unlabeled nodes (hubs in the graph) we may believe that the neighbors are not going to produce reliable label and hence the probability of undefined label $\perp$ should be higher. The optimization problem can be solved with an efficient iterative algorithm which is parallelized in a MapReduce framework (Talukdar et al., 2008; Rao and Yarowsky, 2009). We used the *Junto label propagation* toolkit (Talukdar and Crammer, 2009) for label propagation.

5.3.2 Efficient Graph Construction

Graph-based approaches can easily become computationally very expensive as the number of nodes grow. In our case, we use phrases in the monolingual text as graph vertices. These

[6] Capturing those cases where the given data is not enough to reliably compute a soft labeling using the initial *m real* labels.

[7] The values of these hyper-parameters are set to their defaults in the *Junto* toolkit (Talukdar and Crammer, 2009).

phrases are n-grams up to a certain number, which can result in millions of nodes. For each node a distributional profile (DP) needs to be created. The number of possible edges can easily explode in size as there can be as many as $O(n^2)$ edges where n is the number of nodes. A common practice to control the number of edges is to connect each node to at most k other nodes (k-nearest neighbor). However, finding the top-k nearest nodes to each node requires considering its similarity to all the other nodes which requires $O(n^2)$ computations and since n is usually very large, doing such is practically intractable. Therefore, researchers usually resort to an approximate k-NN algorithms such as *locality-sensitive hashing* (Ravichandran et al., 2005; Goyal et al., 2012).

Fortunately, since we use context words as cues for relating their meaning and since the similarity measures are defined based on these cues, the number of neighbors we need to consider for each node is reduced by several orders of magnitude. We incorporate an inverted-index-style data structure which indicates what nodes are neighbors based on each context word. Therefore, the set of neighbors of a node consists of union of all the neighbors bridged by each context word in the DP of the node. However, the number of neighbors to be considered for each node even after this drastic reduction is still large (in order of a few thousands).

Figure 5.2 shows a portion of a graph constructed on unigram nodes of the French monolingual. The graph highlights the paraphrases of “spécialement” and their respective paraphrases.

5.4 Experiments & Results

5.4.1 Experimental Setup

We experimented with two different domains for the bilingual data: the *Europarl* corpus (v7) (Koehn, 2005), and *European Medicines Agency* documents (EMEA) (Tiedemann, 2009) from French to English. For the monolingual data, we used French side of the Europarl corpus and we used ACL/WMT 2005[8] data for dev/test sets. Table 5.1 summarizes statistics of the datasets used.

From the dev and test sets, we extract all source words that do not appear in the

[8] http://www.statmt.org/wpt05/mt-shared-task/

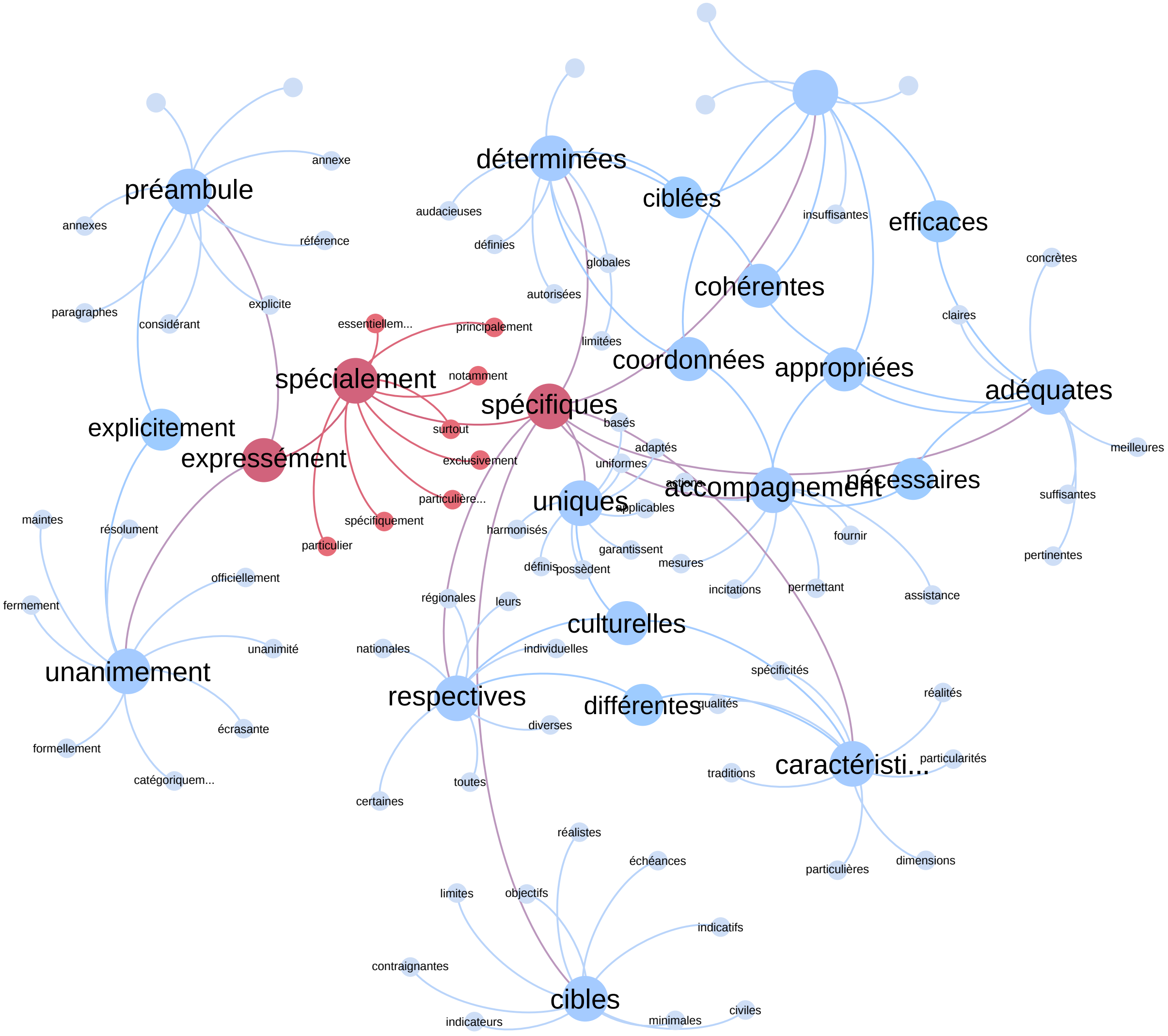


Figure 5.2: A portion of unigram graph constructed on the French side of Europarl.

Dataset	Domain	Sents	Tokens	
			Fr	En
Bitext	Europarl	10K	298K	268K
	EMEA	1M	16M	14M
Monotext	Europarl	2M	60M	–
Dev-set	WMT05	2K	67K	58K
Test-set	WMT05	2K	66K	58K

Table 5.1: Statistics of training sets in different domains.

Dataset	Dev		Test	
	types	tokens	types	tokens
Europarl	1893	2229	1830	2163
EMEA	2325	4317	2294	4190

Table 5.2: number of oovs in dev and test sets for Europarl and EMEA systems.

phrase-table constructed from the parallel data. From the oovs, we exclude numbers as well as named entities. We apply a simple heuristic to detect named entities: basically words that are capitalized in the original dev/test set that do not appear at the beginning of a sentence are named entities. Table 5.2 shows the number of oov types and tokens for Europarl and EMEA systems in both dev and test sets.

For the end-to-end MT pipeline, we used Moses (Koehn et al., 2007a) with these standard features: relative-frequency and lexical translation model (TM) probabilities in both directions; distortion model; language model (LM) and word count. Word alignment is done using GIZA++ (Och and Ney, 2003). We used distortion limit of 6 and max-phrase-length of 10 in all the experiments. For the language model, we used the KenLM toolkit (Heafield, 2011a) to create a 5-gram language model on the target side of the Europarl corpus (v7) with approximately $54M$ tokens with Kneser-Ney smoothing.

5.4.2 Phrase-table Integration

Once the translations and their probabilities for each oov are extracted, they are added to the phrase-table that is induced from the parallel text. The probability for new entries are added as a new feature in the log-linear framework to be tuned along with other features. The value of this newly introduced feature for original entries in the phrase-table is set to

1. Similarly, the value of original four probability features in the phrase-table for the new entries are set to 1. The entire training pipeline is as follows:

1. A phrase table is constructed using the parallel data as usual;
2. Oovs for dev and test sets are extracted and numbers and named-entities are removed;
3. Oovs are translated using the graph propagation approach;
4. Oovs and translations are added to the phrase table by introducing a new feature type;
5. The new phrase table is tuned (with a LM) using MERT (Och, 2003a) on the dev set.

5.4.3 Evaluation

If we have a list of possible translations for oovs with their probabilities, we become able to evaluate different methods we discussed. We word-aligned the dev/test sets by concatenating them to a large parallel corpus and running GIZA++ on the whole set. The resulting word alignments are used to extract the translations for each oov, making the gold standard. The correctness of this gold standard is limited to the size of the parallel data used as well as the quality of the word alignment software toolkit, and is not 100% precise. However, it gives a good estimate of how each oov should be translated without the need for human judgements.

For evaluating our baseline as well as graph-based approaches, we use both intrinsic and extrinsic evaluations. Two intrinsic evaluation metrics that we use to evaluate the possible translations for oovs are *Mean Reciprocal Rank* (MRR) (Voorhees, 1999) and *Recall*. Intrinsic evaluation metrics are faster to apply and are used to optimize different hyper-parameters of the approach (e.g. window size, phrase length, etc.). Once we come up with the optimized values for the hyper-parameters, we extrinsically evaluate different approaches by adding the new translations to the phrase-table and run it through the MT pipeline.

MRR

MRR is an Information Retrieval metric used to evaluate any process that produces a ranked list of possible candidates. The reciprocal rank of a list is the inverse of the rank of the correct answer in the list. Such score is averaged over a set, oov set in our case, to get the mean-reciprocal-rank score.

$$\text{MRR} = \frac{1}{|O|}\sum_{i=1}^{|O|}\frac{1}{rank_i} \qquad O = \{oov\}$$

In a few cases, there are multiple translations for an oov word (i.e. appearing more than once in the parallel corpus and being assigned to multiple different phrases), we take the average of reciprocal ranks for each of them.

Recall

MRR takes the probabilities of oov translations into account in sorting the list of candidate translations. However, in an MT pipeline, the language model is supposed to rerank the hypotheses and move more appropriate translations (in terms of fluency) to the top of the list. Hence, we also evaluate our candidate translation regardless of the ranks. Since Moses uses a certain number of translations per source phrase (called the translation table limit or *ttl* which we set to 20 in our experiments) , we use the *recall* measure to evaluate the top *ttl* translations in the list. Recall is another Information Retrieval measure that is the fraction of correct answers that are retrieved. For example, it assigns score of 1 if the correct translation of the oov word is in the top-k list and 0 otherwise. The scores are averaged over all oovs to compute recall.

$$\text{Recall} = \frac{|\{\text{gold standard}\} \cap \{\text{candidate list}\}|}{|\{\text{gold standard}\}|}$$

5.4.4 Intrinsic Results

In Section 5.2.2 and 5.2.3, different types of association measures and similarity measures have been explained to build and compare distributional profiles. Table 5.3 shows the results on Europarl when using different similarity combinations. The measures are evaluated by fixing the window size to 4 and maximum candidate paraphrase length to 2 (e.g. bigram). First column shows the association measures used to build DPs. As the results show, the combination of PMI as association measure and cosine as DP similarity measure outperforms the other possible combinations. We use these two measures throughout the rest of the experiments.

Assoc	cosine(%)		L_1norm(%)		JSD(%)	
	MRR	RCL	MRR	RCL	MRR	RCL
CP	1.66	4.16	2.18	5.55	2.33	6.32
LLR	1.79	4.26	0.13	0.37	0.5	1.00
PMI	**3.91**	**7.75**	0.50	1.17	0.59	1.21
Chi	1.66	4.16	0.26	0.55	0.03	0.05

Table 5.3: Results of intrinsic evaluations (MRR and Recall) on Europarl, window size 4 and paraphrase length 2

Figure 5.3 illustrates the effects of different window sizes[9] and paraphrase lengths on MRR. As the figure shows, the best MRR is reached when using window size of 4 and trigram nodes. Going from trigram to 4-gram results in a drop in MRR. One reason would be that distributional profiles for 4-grams are very sparse and that negatively affects the stability of similarity measures.

Figure 5.4 illustrates the effect of increasing the size of monolingual text on both MRR and recall. 1× refers to the case of using $125k$ sentences for the monolingual text and the 16× indicates using the whole Europarl text on the source side ($\approx 2M$ sentences). As shown, there is a linear correlation between the logarithm of the data size and the MRR and recall ratios. Interestingly, MRR is growing faster than recall by increasing the monolingual text size, which means that the scoring function gets better when more data is available. The figure also indicates that a much bigger monolingual text data can be used to further improve the quality of the translations, however, at the expense of more computational resources.

Graph-based Results

Table 5.4 shows the intrinsic results on the Europarl corpus when using unigram nodes in each of the graphs. The results are evaluated on the dev-set based on the gold alignment created using GIZA++. Each node is connected to at most 20 other nodes (same as the max-paraphrase-limit in the baseline). For the tripartite graph, each node is connected to 15 labeled nodes and 5 unlabeled ones. The tripartite graph gets a slight improvement over the bipartite one, however, the full graph failed to have the same increase. One reason

[9] Here "window size" refers to the number of context words on either side of each phrase.

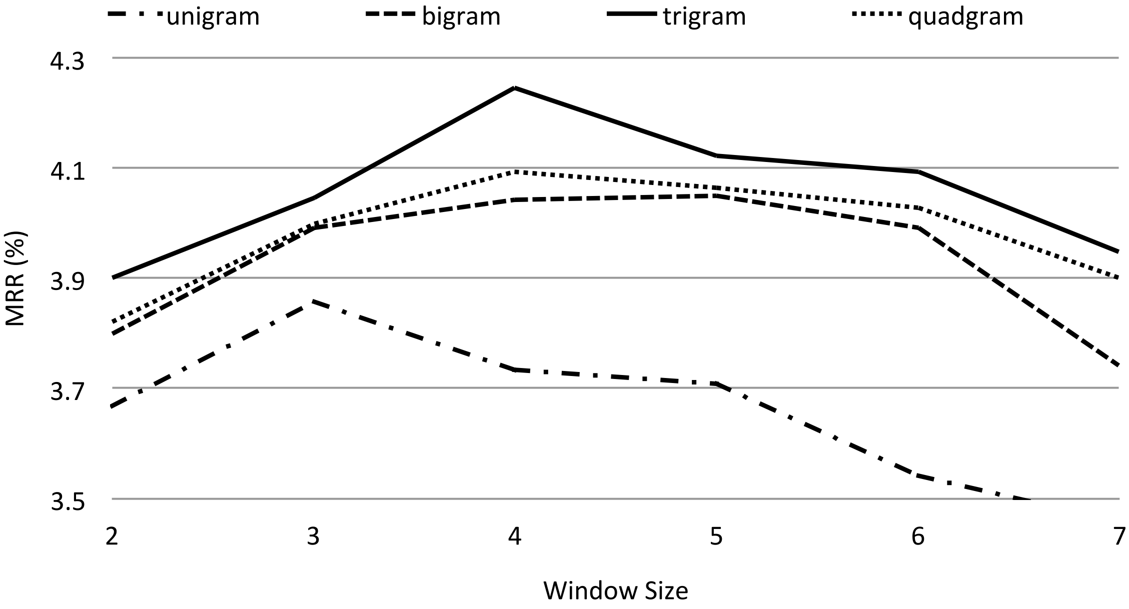


Figure 5.3: Effects of different window sizes and paraphrase length on the MRR of the dev set.

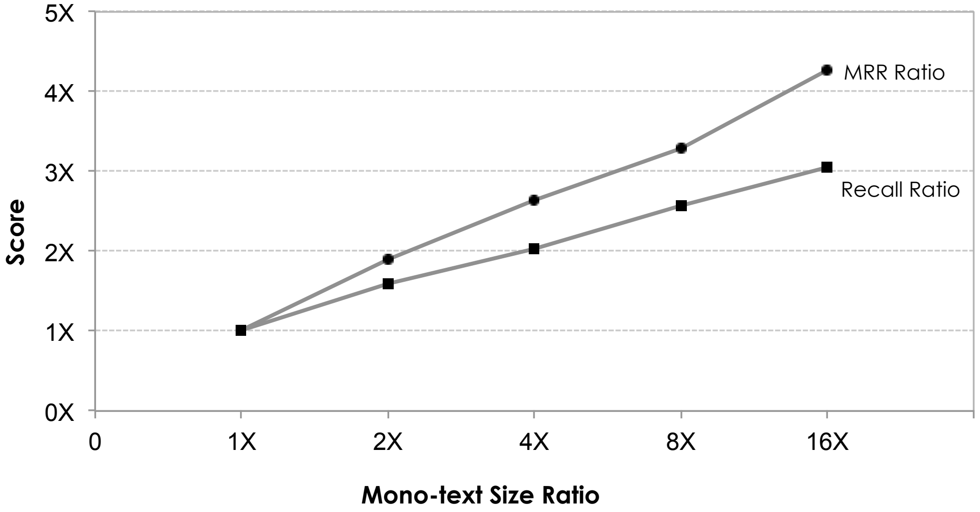


Figure 5.4: Effect of increasing the monolingual text size on MRR and Recall.

Graph	Neighbor	MRR %	RCL %
Bipartite	20	5.2	12.5
Tripartite	15+5	5.9	12.6
Full	20	5.1	10.9
Baseline	20	3.7	7.2

Table 5.4: Intrinsic results of different types of graphs when using unigram nodes on Europarl.

is that allowing paths longer than 2 between oov and labeled nodes causes more noise to propagate into the graph. In other words, a paraphrase of a paraphrase of a paraphrase is not necessarily a useful paraphrase for an oov as the translation may no longer be a valid one.

Table 5.5 also shows the effect of using bigrams in addition to unigrams as graph nodes. There is an improvement by going from unigrams to bigrams in both bipartite and tripartite graphs. We did not use trigrams or larger n-grams in our experiments due to computational limitations.

Type	Node	MRR %	RCL %
Bipartite	unigram	5.2	12.5
	bigram	6.8	15.7
Tripartite	unigram	5.9	12.6
	bigram	6.9	15.9
Baseline	bigram	3.9	7.7

Table 5.5: Results on using unigram or bigram nodes.

Corpus	System	MRR	Recall	Dev BLEU	Test BLEU
Europarl	Baseline	–	–	28.53	28.97
	Our approach	5.9	12.6	28.76	29.40*
EMEA	Baseline	–	–	20.05	20.34
	Our approach	3.6	7.4	20.54	20.80*

* Statistically significant with $p < 0.02$ using the bootstrap resampling significance test (in Moses).

Table 5.6: BLEU scores for different domains with or without using oov translations.

5.4.5 Extrinsic Results

The generated candidate translations for the oovs can be added to the phrase-table created using the parallel corpus to increase the coverage of the phrase-table. This aggregated phrase-table is to be tuned along with the language model on the dev set, and run on the test set. BLEU (Papineni et al., 2002b) is still the de facto evaluation metric for machine translation and we use that to measure the quality of our proposed approaches for MT. In these experiments, we do not use alignment information on dev or test sets unlike the previous section.

Table 5.6 reports the BLEU scores for different domains when the oov translations from the graph propagation is added to the phrase-table and compares them with the baseline system (i.e. Moses). Results for our approach is based on unigram tripartite graphs and show that we improve over the baseline in both the same-domain (Europarl) and domain adaptation (EMEA) settings. Table 5.7 shows some translations found by our system for oov words.

oov	gold standard	candidate list
assentiment	approval	support agreement approval accession will approve endorses
spécialement	undone particularly especially special particular	particularly specific only particular should and especially

Table 5.7: Two examples of oov translations found by our method.

5.5 Related work

There has been a long line of research on learning translation pairs from non-parallel corpora (Rapp, 1995; Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009; Marton et al., 2009; Laws et al., 2010). Most have focused on extracting a translation lexicon by mining monolingual resources of data to find clues, using probabilistic methods to map words, or by exploiting the cross-language evidence of closely related languages. Most of them evaluated only high-frequency words of specific types (nouns or content words) (Rapp, 1995; Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009; Laws et al., 2010). In contrast, we do not consider any constraint on our test data and our data includes many low frequency words. It has been shown that translation of high-frequency words is easier than low frequency words (Tamura et al., 2012).

Some methods have used a third language(s) as pivot or bridge to find translation pairs (Mann and Yarowsky, 2001; Schafer and Yarowsky, 2002; Callison-Burch et al., 2006).

Context similarity has been used effectively in bilingual lexicon induction (Rapp, 1995; Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009; Marton et al., 2009; Laws et al., 2010). It has been modeled in different ways: in terms of adjacent words (Rapp, 1999; Fung and Yee, 1998), or dependency relations (Garera et al., 2009). Recently linguistic analysis and dependency information between words have been shown to be useful in improving translation accuracy. Garera et al. (2009) modeled distributional profiles using dependency relations rather than adjacency. Instead of applying the fixed-size window on the sequence of words, the window is applied on the dependency tree of the sentence. This approach helps to capture long distance dependencies between words in the distributional profiles, even if they fall outside of the sliding window. Laws et al. (2010) used linguistic analysis in the form of graph-based models instead of a vector space. But all of these researches used an available seed lexicon as the basic source of similarity between source and target languages unlike our method which just needs a monolingual corpus of source language which is freely available for many languages and a small bilingual corpora.

Some methods tried to alleviate the lack of seed lexicon by using orthographic similarity to extract a seed lexicon (Koehn and Knight, 2002; Fišer and Ljubešić, 2011). They used identical words between source and target to create a seed lexicon, then used clues like context and orthographic similarities to extend this lexicon. However, this method is less practical in case of unrelated languages.

Haghighi et al. (2008) and Daumé and Jagarlamudi (2011) proposed generative models based on canonical correlation analysis to extract translation lexicons for non-parallel corpora by learning a matching between source and target lexicons. Using monolingual features to represent words, feature vectors are projected from source and target words into a canonical space to find the appropriate matching between them. Their method relies on context features which need a seed lexicon and orthographic features which only works for phylogenetically related languages.

Graph-based semi-supervised methods have been shown to be useful for domain adaptation in MT as well. Alexandrescu and Kirchhoff (2009) applied a graph-based method to determine similarities between sentences and use these similarities to promote similar translations for similar sentences. They used a graph-based semi-supervised model to re-rank the n-best translation hypothesis. Liu et al. (2012) extended Alexandrescu and Kirchhoff (2009)'s model to use translation consensus among similar sentences in bilingual training data by developing a new structured label propagation method. They derived some features to use during decoding process that has been shown useful in improving translation quality. Our graph propagation method connects monolingual source phrases with oovs to obtain translation and so is a very different use of graph propagation from these previous works.

Recently label propagation has been used for lexicon induction (Tamura et al., 2012). They used a graph based on context similarity as well as co-occurrence graph in propagation process. Similar to our approach they used unlabeled nodes in label propagation process.

However, they use a seed lexicon to define labels and comparable corpora to construct graphs unlike our approach.

5.6 Conclusion

We presented a novel approach for inducing oov translations from a monolingual corpus on the source side and a parallel data using graph propagation. Our results showed improvement over the baselines both in intrinsic evaluations and BLEU.

Future work includes studying the effect of size of parallel corpus on the induced oov translations. Increasing the size of parallel corpus on one hand reduces the number of oovs. But, on the other hand, there will be more labeled paraphrases that increases the chance of finding the correct translation for oovs in the test set.

Currently, we find paraphrases for oov words. However, oovs can be considered as n-grams (phrases) instead of unigrams. In this scenario, we can also look for paraphrases and translations for phrases containing oovs and add them to the phrase-table as new translations along with the translations for unigram oovs.

We also plan to explore different graph propagation objective functions. Regularizing these objective functions appropriately might let us scale to much larger data sets with an order of magnitude more nodes in the graph. We are also considering using locality-sensitive hashing (Ravichandran et al., 2005; Goyal et al., 2012) for constructing the graph.

Chapter 6

Conclusion

In this chapter, we briefly review the approaches presented for four different scenarios to enhancing statistical machine translation systems using additional resources. In Section 6.2, we briefly restate future directions for all four approaches. Section 6.3 discusses how all these methods can be integrated into a single system that is able to take advantage of all possible sources for a language pair.

6.1 Summary

In this thesis, we proposed approaches that incorporate diverse sources to improve the quality of current machine translation systems.

1. We introduced a novel method, ensemble decoding, that combines multiple translation models in the decoder on-the-fly. This combination method can be used when an additional parallel corpus between the source and the target language is available;

2. We proposed an approach based on ensemble decoding, ensemble triangulation, to alleviate the problem of scarce parallel corpora for resource-poor languages. This approach takes advantage of available parallel corpora between the source language and a third language and from that to the target language to build a pivot system and then combines multiple such systems together and to the direct system to make a stronger system;

3. We proposed the use of *stacking* to the statistical machine translation (SMT) models. We create a diverse ensemble of weak learners using a hierarchical phrase-based system

by manipulating the training data. We then create a strong model by combining the weak models on-the-fly. This method can gain higher translation quality, measured by BLEU, without using any additional resources.

4. Finally we proposed an approach to make use of a source-language monolingual corpus to improve translation quality. It mines translations for out-of-vocabulary (oov) words by constructing a graph on the source language monolingual text and employ a graph propagation technique to propagate translations from phrases to their paraphrases.

6.2 Future directions

6.2.1 Future Directions for Ensemble Decoding

Global Normalization

As mentioned in Section 2.3.3, the scores in the log-linear models are not normalized since we only use them to rank hypothesis.

$$p(\bar{e} \mid \bar{f}) \;\propto\; \exp\left(\sum_i w_i \phi_i(\bar{e}, \bar{f})\right)$$

However, this would cause a problem when using multiple language models in multiple systems. Though, in the experiments reported in Section 2.4 we did not suffer from this problem as a result of using a shared language model in combination with L1 normalized weights. A more principled approach is to exactly compute the normalized scores using the inside-outside algorithm.

In this approach, each system separately parses each sentence without consulting other systems' translation models. All the hypotheses scores are normalized using the inside and outside scores. Next, an ensemble CKY chart is populated from partial hypotheses located in all corresponding CKY chart cells. The rest of the approach remains unchanged.

Domain Mixing Scenario

In this setting, the training, dev and test sets consist of sentences from a variety of domains. However, the sentences are not labeled with the domain they are belonging to. This use case is similar to what translation web services such as Google Translate and Bing Translator face with on daily basis. Eidelman et al. (2012) suggests discovering latent topics (i.e.

finer-grained domains) using an unsupervised approach (LDA) and they used these topic distributions to compute topic-dependent lexical weighting probabilities. These probabilities are added to translation models as features. This approach can gain up to 1 BLEU point over a strong baseline.

In this setting, we can take advantage of unsupervised topic modeling toolkits to cluster the corpus into N subcorpora. Then a separate translation model can be learned on each subcorpora and the ensemble decoding approach can be applied on these models. One potential problem with this approach would be sparsity as the translation model probabilities would be estimated on smaller data. One remedy to this problem is to learn a general translation model on the whole corpus and do an ensemble model on this general model and all sub-corpora-based models. Furthermore, learning a separate language model on each subcorpora can also be beneficial when using in conjunction with a general language model.

Mixture Operation Characteristics

In Section 2.3.1, we defined five mixture operations and we reported the BLEU scores when using them in ensemble decoding (Section 2.4). Each of these mixture operations has specific properties that make it work in specific domain adaptation or system combination scenarios. For instance, prod, or in general LOPs, may not be optimal for domain adaptation in the setting where there are two or more models trained on heterogeneous corpora. As discussed in Smith et al. (2005), LOPs work best when all the models accuracies are high and close to each other with some degree of diversity. LOPs give veto power to any of the component models and this perfectly works for settings such as the one in Petrov (2010) where a number of parsers are trained by changing the randomization seeds but having the same base parser and using the same training set. They noticed that parsers trained using different randomization seeds have high accuracies but there are some diversities among them and they used product models for their advantage to build a better parser by combining the base models. We assume that each of the models is expert in some parts and so they do not necessarily agree on correct hypotheses. In other words, product models (or LOPs) tend to have intersection-style effects while we are more interested in union-style effects.

We would like to study the characteristics of other mixture operations and figure out what operations would best work in what settings. The results can be used to potentially come up with better mixture operations.

Consensus Ensemble Decoding

DeNero et al. (2009) introduces a variant of MBR, *consensus decoding*, that applies efficiently to translation forests rather than k-best lists. Instead of maximizing expected similarity (i.e. BLEU score), similarity is expressed in terms of n-gram features and translations are scored with respect to similarity to expected feature values:

$$\begin{aligned} e^* &= \operatorname*{argmax}_e \mathbb{E}_{p(e'|f)}[BLEU(e,e')] \\ &\approx \operatorname*{argmax}_e BLEU(e, \mathbb{E}_{p(e'|f)}[\phi(e')]) \end{aligned}$$

We propose to apply the techniques of consensus decoding in our ensemble method. More specifically, once the normalization step (see Section 2.7.1) is done, ensemble decoding combines hypotheses from all the models. Meanwhile, the n-gram expected counts are collected in the ensemble decoder. Once the input sentence is fully parsed, all the candidate translations are scored based on the new n-gram-based objective function and the translation with highest score is chosen as the system output.

The idea of applying consensus decoding on multiple systems has been successfully used in the *model combination* approach of DeNero et al. (2010). This approach assumes that each system provides expectations of n-gram features, though, it does not care about the latent structure of component systems. The objective function used in this approach is:

$$s_w(d) = \sum_{i=1}^{I} \left(\sum_{n=1}^{4} w_i^n v_i^n(d) + w_i^\alpha \alpha_i(d) \right) + w^b.b(d) + w^l.l(d)$$

This objective function scores a derivation d using n-gram scores from I different systems with weights $\mathbf{w}$. $\alpha_i(d)$ is a system indicator feature which is 1 if the derivation d came from the system i and 0 otherwise. $b(d)$ is the model score of the derivation d under the model it is from and l is the target side length. v_i^n is combination feature function on n-grams for system i, that is:

$$\begin{aligned} v_i^n(d) &= \sum_{g \in \text{ngram}(d)} v_i^n(g) \\ &= \sum_{g \in \text{ngram(d)}} \mathbb{E}_{p_i(d'|f)}[c(g,d')] \\ &= \sum_{g \in \text{ngram(d)}} \sum_{d'} p_i(d'|f) c(g,d') \end{aligned}$$

However, DeNero et al. (2010) do not intermix search spaces from multiple systems while our ensemble decoding method is able to generate new sentences that are not in any of the component systems' search spaces. Another advantage of using consensus decoding on top of ensemble decoding is that we can benefit from the hypergraph-based minimum error-rate training algorithm of Kumar et al. (2009) and have a more systematic tuning procedure, replacing Condor.

6.2.2 Future Directions for Ensemble Triangulation

Future work for Ensemble Triangulation includes imposing restrictions on the generated triangulated rules in order to keep only ones that have a strong support from the word alignments. By exploiting such constraints, we can experiment with larger sizes of parallel data. Specifically, a more natural experimental setup for triangulation which we would like to try is to use a small direct system with big *src* → *pvt* and *pvt* → *tgt* systems. This resembles the actual situation for resource-poor language pairs. We will also experiments with higher number of pivot languages.

Currently, most research in this area focuses on triangulation on paths containing only one pivot language. We can also analyze our method when using more languages in the triangulation chain and see whether there would any gain in doing such.

Finally, in current methods all $(\bar{f}, \bar{i})$ phrase pairs of the *src* → *pvt* systems, for which there does not exist any $(\bar{i}, \bar{e})$ pair in *pvt* → *tgt* are simply discarded. However in most cases, such $\bar{i}$ phrases can be segmented into smaller phrases (or rules for Hiero systems) to be triangulated via them. This segmentation is a decoding problem which requires an efficient algorithm to be practical.

6.2.3 Future Directions for Stacking

Future work includes experimenting with larger training sets to investigate how useful this approach can be when having larger sizes of training data and how each mixture operation behaves when the training data enlarges. In addition, implementing efficient algorithms for reducing the training time by keeping track of each sentence's computations for different phases of training, e.g. word alignment, phrase extraction and phrase scoring and reusing them is among future work.

6.2.4 Future Directions for Graph Propagation for Paraphrasing OOVs

Future work includes studying the effect of size of parallel corpus on the induced oov translations. Increasing the size of parallel corpus on one hand reduces the number of oovs. But, on the other hand, there will be more labeled paraphrases that increases the chance of finding the correct translation for oovs in the test set.

Currently, we find paraphrases for oov words. However, oovs can be considered as n-grams (phrases) instead of unigrams. In this scenario, we can also look for paraphrases and translations for phrases containing oovs and add them to the phrase-table as new translations along with the translations for unigram oovs.

We also plan to explore different graph propagation objective functions. Regularizing these objective functions appropriately might let us scale to much larger data sets with an order of magnitude more nodes in the graph. We are also considering using locality-sensitive hashing (Ravichandran et al., 2005; Goyal et al., 2012) for constructing the graph.

6.3 Integration into a Single Model

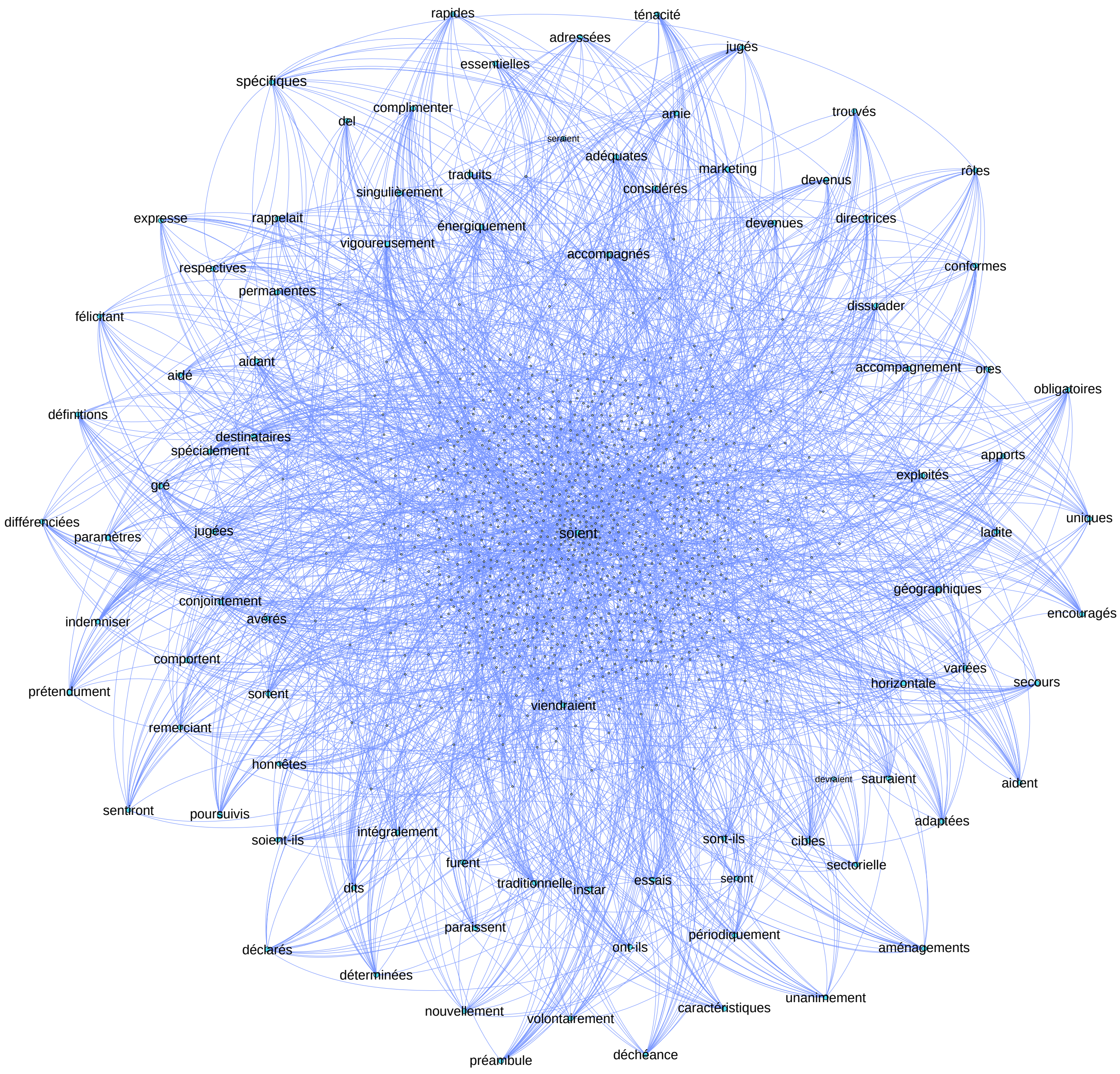
All four approaches introduced in this thesis are orthogonal to one another and can be integrated into a single system to boost the translation quality. Here we explain how they can be combined in practice:

- All *src* → *tgt* parallel corpora can be partitioned using k-fold cross validation technique and a system is built on each partition of each corpus comprising $k \times c$ systems (c is the number of such corpora). These systems can be integrated using the stacking approach to increase the bias of the integrated system;

- Oovs can be passed to a graph-based translation mining method to find translations for. This step can be done offline and translations can be added to the phrase-tables of the previous step. Note that the translation candidates for each oov can be different based on the phrase-table that is used to assign labels to labeled nodes of the graph.

- For each pair of *src* → *pvt* and *pvt* → *tgt* corpora, a triangulated system can be built dynamically and combined with the systems built in the first step using the ensemble triangulated approach. This way we can i) reduce the number of oovs using the pivot language(s) and ii) smooth the translation model probabilities;

All in all, this thesis presented a number of approaches for resource-poor languages, on how to exploit other sources of information in the current SMT models. We showed that in the absence of sufficient parallel data, one can use other sources to improve the quality of SMT systems. These types of sources can further be extended to linguistic resources (e.g. dictionaries), monolingual text on the target side as well as comparable data, which are left for future work.


rapides
ténacité
adressées
jugés
essentielles
spécifiques
complimenter
del
amie
trouvés
seraient
adéquates
traduits
marketing
rôles
singulièrement
considérés
devenus
expresse
rappelait
énergiquement
devenues
directrices
vigoureusement
accompagnés
respectives
conformes
permanentes
dissuader
félicitant
aidant
accompagnement
ores
aidé
obligatoires
définitions
destinataires
spécialement
apports
exploités
gré
uniques
différenciées
paramètres
jugées
soient
ladite
géographiques
conjointement
encouragés
indemniser
avérés
comportent
variées
horizontale
secours
prétendument
sortent
viendraient
remerciant
honnêtes
devraient
sauraient
aident
sentiront
poursuivis
adaptées
intégralement
soient-ils
sont-ils
cibles
furent
sectorielle
dits
traditionnelle
instar
essais
seront
paraissent
périodiquement
déclarés
ont-ils
aménagements
déterminées
nouvellement
volontairement
caractéristiques
unanimement
préambule
déchéance

Appendix A

Kriya - A Hierarchical Phrase-based MT System

In this chapter *Kriya*, our in-house statistical machine translation (SMT) system is explained. We have modified the decoder of Kriya to support ensemble decoding. Kriya is a hierarchical phrase-based system (Chiang, 2007b), for which Baskaran Sankaran has been the lead developer. The text in this section is taken from Sankaran et al. (2012).

Kriya supports both a grammar extraction module for synchronous context-free grammars (SCFGs) and a CKY-based decoder. There are several re-implementations of Hiero in the machine translation community, but Kriya offers the following novel contributions: (a) Grammar extraction in Kriya supports extraction of the full set of Hiero-style SCFG rules but also supports the extraction of several types of compact rule sets which leads to faster decoding for different language pairs without compromising the BLEU scores. Kriya currently supports extraction of compact SCFGs such as grammars with one non-terminal and grammar pruning based on certain rule patterns. (b) The Kriya decoder can take advantage of parallelization using a networked cluster. (c) The Kriya decoder offers some unique improvements in the implementation of cube pruning, such as increasing diversity in the target language *n*-best output and novel methods for language model (LM) integration. Kriya supports KENLM and SRILM for language model queries and exploits advanced features such as *n*-gram history states in KENLM. This chapter also provides several experimental results which demonstrate that the translation quality of Kriya compares favourably to the Moses (Koehn et al., 2007b) phrase-based system in several language pairs while showing a substantial improvement for Chinese-English similar to (Chiang, 2007b). We also quantify the model sizes for phrase-based and Hiero-style systems apart from presenting experiments comparing variants of Hiero models.

A.1 Introduction

Hierarchical Phrase-based Machine Translation (Chiang, 2005b; Chiang, 2007b) has been one of the recent approaches in Statistical Machine Translation (SMT) gaining prominence. This has been proven to be comparable to or better than Phrase-based systems for several language pairs.

In this chapter, we present Kriya which implements a hierarchical phrase-based machine translation system which includes a grammar extraction module and decoder. The name Kriya is the Sanskrit word for *verb* to signify that syntactic parsing techniques can be useful for machine translation.

Kriya is similar to some of the existing hierarchical phrase-based systems, but has some distinguishing features. For example, Kriya has a unique approach for computing the language model (LM) heuristic in Cube pruning (Chiang, 2007b) which also improves diversity in the cube-pruning step and both ideas lead to small but consistent improvements in BLEU. Kriya supports extraction of different types of more compact grammars as an alternative to full grammars typically extracted using the synchronous CFG (SCFG) extraction heuristics described in the original Hiero paper. The full grammar is typically associated with issues such as over-generation and search errors (de Gispert et al., 2010) and the use of compact grammars can achieve BLEU scores comparable to full grammar. Kriya also supports shallow-n decoding that leads to faster decoding while maintaining same BLEU scores as the full decoding.

The rest of the chapter is structured as follows. First we review some of the existing Machine Translation systems focusing on Hiero-style systems (Section A.2) highlighting specific features. We then give a brief definition of synchronous context-free grammar (SCFG) in Section A.3 to set the stage. In Section A.4 we describe both grammar extractor and decoder modules interspersed with the features in Kriya. We finally present some experiments (Section A.5) comparing Kriya with the well-known phrase-based system Moses which is used to benchmark Kriya's performance for several language pairs.

A.2 Related Works

`Moses`[1] (Koehn et al., 2007b) is an open source toolkit that supports three types of state-of-the-art statistical machine translation systems: phrase-based, hierarchical phrase-based and syntax-based SMT. The toolkit is written in C++ and supports SRILM (Stolcke, 2002b), KenLM (Heafield, 2011b), randLM (Talbot and Osborne, 2007) and irstLM (Federico et al., 2008) for language model queries. To speed up training, tuning and test steps, Moses supports Oracle Grid Engine[2] (formerly Sun Grid Engine) and Amazon EC2 cloud and implements several memory/speed optimization algorithms. Chart decoding is done by the CKY+ algorithm which enables it to process arbitrary context free grammars with no limitations on the number of terminals or non-terminals in a rule. It also implements Chiang (2007b)'s cube pruning algorithm. Advanced methods such as Factored Models (Koehn and Hoang, 2007), Minimum Bayes Risk (MBR) decoding, Lattice MBR, Consensus Decoding and multiple translation table decoding (to name a few) have been implemented in Moses.

`Joshua`[3] (Li et al., 2009a; Li et al., 2010; Weese et al., 2011) developed at the Center for Language and Speech Processing at the Johns Hopkins University, is an open source machine translation toolkit written in Java that implements most critical algorithms required for hierarchical decoding such as chart-parsing, n-gram language model integration, beam and cube-pruning and k-best extraction. An advantage of this toolkit is that each component in the machine translation pipeline can be run with other components or separately such as Z-MERT (Zaidan, 2009) which is a stand-alone implementation of Och (2002)'s algorithm written in Java. The toolkit implements training corpus sub-sampling by which the most representative subset of the training corpus is used to extract rules from resulting in a faster training phase. In addition, Minimum Bayes Risk, Deterministic Annealing and Variational Decoding algorithms are implemented in this toolkit.

`cdec`[4] (Dyer et al., 2010) is another translation toolkit written in C++ which allows training and decoding a number of statistical machine translation models, including word-based models, phrase-based models and hierarchical phrase-based models. cdec provides

[1] `http://www.statmt.org/moses`

[2] `http://www.oracle.com/us/sun`

[3] `http://www.sourceforge.net/projects/joshua`

[4] `https://github.com/redpony/cdec`

support for Hadoop (an implementation of a distributed filesystem and MapReduce) for parallelization. Input to this system can be a sentence, lattice or context-free forest, which is then transformed to a unified translation forest. Secondly, language model re-scoring, pruning, inference algorithms and k-best derivation extraction are uniformly applied to the generated translation forest. cdec supports a number of optimization algorithms, including Minimum Error Rate Training (MERT) (Och, 2003b), LBFGS (Liu and Nocedal, 1989), RPROP (Riedmiller and Braun, 1993) and Stochastic Gradient Descent. Compared to Joshua, cdec uses a smaller memory footprint with the same running time (Dyer et al., 2010).

`Jane`[5] (Vilar et al., 2010; Stein et al., 2011), RWTH's hierarchical phrase-based translation system, is a more recent open source toolkit which offers similar features. It is written in C++ and includes tools for phrase extraction and translation. Most of the operations can be parallelized by supporting grid engine clusters. The implementation of Jane allows for augmenting the feature set with arbitrary number of additional features as described in (Stein et al., 2011). It also offers two ways to support additional models: combination in log-linear fashion and a mechanism to get the model to score a derivation to be incorporated in the main model's score. For tuning, Jane supports three different optimization methods: Minimum Error Rate Training (MERT) (Och, 2003b), Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006) and the Downhill Simplex method (Nelder and Mead, 1965). Stein (2011) shows that Jane is 50% faster than Joshua on identical settings.

A.3 Synchronous Context-Free Grammar

This section provides a formal definition of a synchronous context-free grammar (SCFG) as a precursor to the discussion of the implementation in Kriya.

Formally a grammar G in hierarchical phrase-based model is a special case of probabilistic synchronous context-free grammar (PSCFG) that is defined as a 4-tuple: $G = (T, NT, R, R_g)$, where, T and NT are the set of terminals and non-terminals in G. Hiero grammars typically use two non-terminals X and S with the latter being a special start symbol as well. R is a set of production rules of the form:

$$X \rightarrow <\gamma,\, \alpha,\, \sim>,\ \gamma, \alpha \in \{X \cup T^{+}\} \tag{A.1}$$

[5] `http://www.hltpr.rwth-aachen.de/jane`

The $\sim$ in the hierarchical rule indicates the alignment indices for the non-terminals in the production rule such that the co-indexed non-terminal pair are rewritten synchronously. These production rules are combined in the top by the *glue* rules R_g leading to the start symbol S:

$$S \rightarrow <X_1,\ X_1> \tag{A.2}$$

$$S \rightarrow <S_1X_2,\ S_1X_2> \tag{A.3}$$

where the non-terminal indices indicate synchronous rewriting of the source and target non-terminals having the same index.

A.4 Kriya

Our implementation of Kriya closely follows the original exposition in (Chiang, 2007b) with extensions that provide several additional features. Broadly, Kriya consists of two independent modules: a *grammar extractor* and a *CKY-based decoder*. Traditionally, grammar extraction has been a bottleneck in Hiero-style translation, due to the massive size of Hiero SCFG grammars and also due to the increasing availability of parallel data. The grammar extractor in Kriya has been designed to efficiently learn translation model even for a very large data set and this is achieved by way of parallelization and optimization. Thus our approach does not resort to *sub-sampling* to choose a smaller representative training set. Alternately Kriya also supports extraction of several variants of more compact grammars, for example extracting a 1 non-terminal grammar or filtering the full grammar based on certain greedily selected rule patterns (Iglesias et al., 2009).

Kriya decoder currently supports SCFG models for *string* inputs and features Cube Pruning (Chiang, 2007b) for integrating the language model scores with the decoder. We introduce a novel approach for improving the heuristic language model scores for the left and right contexts in CKY-based decoder by taking into account the potential position of the target hypothesis fragment in the final candidate. Kriya also supports shallow-n decoding (de Gispert et al., 2010) for fast decoding without impacting the translation quality for certain close language pairs.

Kriya has been written primarily in Python (versions 2.6 and 2.7). This allows us to test new ideas by quickly implementing them in short duration at the same time keeping the code-base small, manageable and easy to read. On the negative side, Kriya is bit slower mainly due to the well-known speed issues in Python, which we alleviate using several engineering optimizations. These optimizations have resulted in practically acceptable training and decoding speeds in Kriya as we later quantify in Section A.5.

A.4.1 Kriya Grammar Extractor

The Hiero grammar extraction algorithm (Chiang, 2007b) starts from the set of *initial phrases* that are identified by growing the word alignments into longer phrases. Given the initial phrases corresponding to a sentence pair, the heuristic algorithm first designates the smaller initial phrases (such as phrase pairs having non-decomposable alignments) as terminal rules, expanded from a non-terminal X. The algorithm then extracts hierarchical rules by substituting the smaller spans within the larger phrases by the non-terminal X if the phrase pair corresponding to smaller span has already been identified as a rule. It extracts all possible rules from the initial phrases subject to a maximum of two X non-terminals in a rule such that they do not rewrite adjacent spans in the source side. The Hiero extraction assumes unit count for each initial phrase and distributes this uniformly to the rules extracted it. The parameter estimation then proceeds by relative frequency estimation.

Chiang proposed the total number of source side (terminals and non-terminals) terms and a maximum rule length to be 5 and 10 respectively. We found improvements with longer source side rules. In comparison, phrase-based models typically use a maximum phrase length of at least 7 and often some even longer phrases (between 10 and 20). The source side length and maximum rule length are customizable parameters in the Kriya rule extraction, to facilitate experiments with different lengths.

A major issue in the extraction of Hiero grammar is the exponential size of the resulting grammar such that the full grammar can not be held in memory for parameter estimation. Some of the existing Hiero systems use *sub-sampling* (Weese et al., 2011) to reduce the size of the training corpus and run the grammar extraction on the sub-sampled corpus, resulting in approximate probability estimates. In contrast Kriya uses the entire training data and we use memory optimizations and parallelization to achieve this.

The grammar extractor in Kriya is modularized to run in three phases in order to efficiently extract grammars even for large training corpora. In the first phase the extractor splits the training corpora into smaller chunks and extracts the rules for each chunk by trivial parallelization over the cluster. The second step scans the rules from the individual chunks and filters them based on the source side texts of a tuning or test set; at the same time collecting the accurate counts for the target phrases in the filtered rules. The final step estimates the forward and reverse probabilities using relative frequency estimation.

The grammar extractor has been customized to the cluster environment (Kriya will soon support the Hadoop framework for extraction and decoding) and thus the extraction can be massively parallelized to efficiently extract Hiero grammar for large corpora. For a smaller data set, it is however possible to estimate the parameters for the full grammar by way of changing the configuration file.

Extracting Compact Grammars

Apart from the original Hiero-style model, Kriya grammar extractor supports the extraction of some variants that are smaller than the full grammar using different pruning[6] strategies. The main motivation for such pruned grammars is two fold, i) to reduce the grammar size and ii) to speed up the decoding enabling faster experiments. In some cases, the resulting compact grammars are suggested to improve the translation by way of reducing search errors (Iglesias et al., 2009; He et al., 2009), which has been contradicted elsewhere (Zollmann et al., 2008; Sankaran et al., 2011).

Kriya supports two different approaches that has been proposed earlier to prune the hierarchical phrase-based grammars. First, the Hiero grammar can be simplified to have just 1 non-terminal, instead of 2 as proposed by Chiang. This grammar eliminates large number of rules, many of which turn out to be composed rules (He et al., 2009) that can be constructed by combining two or more smaller rules leading to spurious ambiguity (Chiang, 2007b) during translation. Such 1 non-terminal grammar have been shown to have BLEU scores similar to the full Hiero grammar (Sankaran et al., 2011) for closer languages such as English-Spanish, but suffer a reduction of 1 BLEU point for Chinese-English and Urdu-English (Zollmann et al., 2008).

In another dimension, pruning based on rule patterns (Iglesias et al., 2009) has been attempted as a means to reduce the size of the grammar. Kriya supports pattern-based filtering, and this could be triggered by using a separate configuration file specifying the

[6] Some earlier works use the word *filtering* for this. We prefer *pruning* (or simplification) to indicate the case where some rules are removed that are otherwise applicable in decoding a given tuning/test set, while reserving the word *filtering* to the process of removing rules that will never be applicable for the tuning/test set. The latter is thus risk-free while the former is lossy.

patterns in the training process.

A.4.2 Kriya Decoder

Kriya currently supports decoding with Hierarchical phrase-based models employing CKY-style chart parsing. Given a source sentence f, the decoder finds the target side yield $\mathscr{Y}_e$ of the best scoring derivation obtained by applying rules in the synchronous context-free grammar.

$$\hat{e} = \mathscr{Y}_e\left(\operatorname*{arg\,max}_{d \in D(f)} P(d)\right) \tag{A.4}$$

where, $D(f)$ is the set of derivations attainable from the learned grammar for the source sentence f. The model over derivations $P(d)$ is formulated as a log-linear model (Och and Ney, 2002) employing a set of features $\{\phi_1, \ldots, \phi_M\}$ apart from a language model feature that scores the target yield as $P_{lm}(e)$. The model can written by factorizing derivation d into its component rules R_d as below.

$$P(d) \propto \left(\prod_{i=1}^{M} \prod_{r \in R_d} \phi_i(r)^{\lambda_i}\right) P_{lm}(e)^{\lambda_{lm}} \tag{A.5}$$

where, λ_i is the corresponding weight of the feature ϕ_i. The feature weights λ_i are optimized against some evaluation metric (Och, 2003b), typically BLEU (Papineni et al., 2002a) score.

The decoder parses the source sentence with a modified version of CKY parser with the target side of corresponding derivations simultaneously yielding the candidate translations. The rule parameters and other features are used to score the derivations along with the language model score of the target translation as in Equation A.5.

The derivation starts from the leaf cells of the CKY chart corresponding to the source side tokens and proceeds bottom-up. For each cell in the CKY chart, the decoder identifies the applicable rules and analogous to monolingual parsing, the non-terminals in these rules should have corresponding entries in the respective antecedent cells. The target side of the production rules yield the translation for the source span and the translations in the top-most cell correspond to the entire sentence. We encourage readers to refer to (Chiang, 2007b) for more details.

Similar to Chiang, we use cube pruning, specifically its lazier version (Huang and Chiang, 2007) to integrate the language model scoring in the decoding process. We introduce a novel approach in improving the heuristic language model score by taking into account the likely position of the target hypothesis fragment in the final translation.

Novel Enhancements in Cube Pruning

The traditional phrase-based decoders using beam search generate the target hypotheses in the left-to-right order. In contrast, CKY decoders in Hiero-style systems can freely combine target hypotheses generated in intermediate cells with hierarchical rules in the higher cells. Thus the generation of the target hypotheses are fragmented and out of order in Hiero, compared to the left to right order preferred by n-gram language models.

This leads to challenges in the estimation of language model scores for partial target hypothesis, which is being addressed in different ways in the existing Hiero-style systems. Some systems add a sentence initial marker (`<s>`) to the beginning of each path and some other systems have this implicitly in the derivation through the translation models. Thus the language model scores for the hypothesis in the intermediate cell are approximated, with the true language model score (taking into account sentence boundaries) being computed in the last cell that spans the entire source sentence.

We introduce a novel improvement in computing the language model scores: for each of the target hypothesis fragment, our approach finds the best position for the fragment in the final sentence and uses the corresponding score. We compute three different scores corresponding to the three positions where the fragment can end up in the final sentence, viz. sentence initial, middle and final: and choose the best score. As an example for fragment t_f consisting of a sequence of target tokens, we compute LM scores for i) `<s>` t_f, ii) t_f and iii) t_f `</s>` and use the best score for pruning alone[7].

This improvement significantly reduces the search errors while performing *cube pruning* (Chiang, 2007b) at the cost of additional language model queries. For example, a partial candidate covering a non-final source span might be reordered to the final position in the target translation. If we just compute the LM score for the target fragment as is done normally, this might get pruned early on before being reordered by a production rule. Our approach instead computes the three LM scores and it would correctly use the last LM score which is likely to be the best, for pruning. Our experiments indicated a small but consistent improvement in the BLEU scores due to this improvement in the LM scores. Additionally, this also produced candidate translations having higher model scores than the naive LM integration (for 57-69% of candidates in corresponding n-best lists) clearly showing that our

[7] This ensures the the LM score estimates are never underestimated for pruning. We retain the LM score for fragment (case ii) for estimating the score for the full candidate sentence later.

approach is able to avoid search errors.

However, additional queries to the language model result in a slight reduction in the decoding speed. This could be partly addressed by saving the three LM scores for both left and right edges with the hypothesis and reusing them appropriately when either or both edges remain unchanged. Secondly following the general strategy, we exploit the *state* information in KenLM (Heafield, 2011b) to query the language model for incremental target fragment following a stored state.

As a second enhancement, Kriya optionally supports improved diversity in the Cube pruning by allowing a fixed number of candidates for each cube that are not represented in the cell. These hypotheses are included in the cell in addition to the hypotheses pushed into the stack through cube pruning. We found the cube-pruning diversity to be useful in our experiments in Arabic-English and Chinese-English, which resulted in statistically significant improvement of 0.25 BLEU points in different settings involving full decoding and shallow-n decoding.

Shallow-n Decoding

Shallow-n grammars (de Gispert et al., 2010) are a class of grammars that restrict the number of successive hierarchical rules in a derivation in order to reduce the over-generation caused by large Hiero grammars. While this has restricted reordering capability compared to full Hiero, the degree of reordering can be customized to the requirements of specific language pairs by way of changing n. For example Shallow-1 grammar might be sufficient for language pairs such as English-French and Arabic-English, whereas higher order shallow grammars are required for Chinese-English because of their large syntactic divergence. As an direct consequence of the reduction in the search space, shallow-n decoding results in substantially faster decoding.

Formally, a Shallow-n grammar G is defined as a 5-tuple: $G = (N, T, R, R_g, S)$, such that T is a set of finite terminals and N a set of finite non-terminals $\{X^0, \ldots, X^N\}$. As earlier R_g refers to the glue rules that rewrite the start symbol S:

$$S \rightarrow <X,\, X> \tag{A.6}$$

$$S \rightarrow <SX,\, SX> \tag{A.7}$$

R is the set of finite production rules in G and has two types, viz. hierarchical (A.8) and terminal (A.9). The hierarchical rules at each level n are additionally conditioned to have *at least* one X^{n-1} non-terminal in them. The $\sim$ in the hierarchical rule serves as the index for aligning the non-terminals such that the co-indexed non-terminal pair can be rewritten synchronously.

$$X^n \rightarrow <\gamma,\, \alpha,\, \sim>,\ \gamma, \alpha \in \{\{X^{n-1}\} \cup T^+\} \tag{A.8}$$

$$X^0 \rightarrow <\gamma,\, \alpha>, \qquad \gamma, \alpha \in T^+ \tag{A.9}$$

Kriya supports Shallow-n decoding, without requiring additional non-terminals to be explicitly created in the Hiero grammar (this is similar to other implementations of this idea). We simply keep track of the number of hierarchical nestings in the partial hypotheses stored in the decoder as part of the hypothesis state. We find the shallow grammars to be comparable to closer language pairs such as English-French and English-Spanish, but the translation performance suffers for Arabic-English and Chinese-English without additional hacks.

A.5 Experiments

In this section we present experiments to evaluate Kriya on several language pairs. We use five different language pairs in our experiments - representing a wide range of diversities, such as close languages (English-French), translating into a slightly more inflected language than English (English-Spanish) and languages with high syntactic divergence (Chinese-English). Table A.1 shows some statistics about the corpora used for our experiments.

Language pair	**Corpus**	**Train/ Tune/ Test**	**Language Model**
English-Spanish	WMT10 (Europarl + News commentary)	1.7 M/ 5078/ 2489	WMT10 *train* + UN
English-French		1.7 M/ 5078/ 2489	WMT10 *train*
French-English		1.7 M/ 5078/ 2489	Gigaword
Chinese-English	*Train*: HK + GALE Phase-1 *Tune*: MTC parts 1 & 3; *Test*: MTC part 4	2.3 M/ 1928/ 919	Gigaword
Arabic-English	ISI automatically extracted Parallel text	1.1 M/ 1982/ 987	Gigaword

Table A.1: Corpus Statistics - English-French use a 4-gram LM and other pairs use 5-gram LM. Chinese-English experiments use four references for tuning and testing.

Language Pair	**Moses**		**Kriya**	
	Model size	*BLEU*	*Model size*	*BLEU*
English-Spanish	154.6	28.12	632.8	**28.19**
English-French	81.8	23.48	519.9	**23.54**
French-English	81.9	26.15	439.9	***26.63***
Arabic-English	68.0	37.31	331.5	***37.74***
Chinese-English	83.6	24.48	286.1	***25.96***

Table A.2: Model sizes and BLEU Scores - Model sizes are in millions of rules. Bold face indicates best BLEU score for each language pair and italicized figures point to statistically significant improvements assuming significance level $\alpha = 0.1$.

First we present experimental results to benchmark Kriya's performance in all these language pairs by comparing it with the well-known Moses phrase-based system. We used standard settings for Moses in all these experiments except for the maximum phrase length, which we set to 7. For Kriya models, we set the total source side terms to be 7 for Chinese-English and Arabic-English and 5 for others. For both Moses and Kriya, we trained lower-cased models for Chinese-English and Arabic-English , while training true-cased models for the rest. We used MERT (Och, 2003b) for optimizing the weights of the features. The BLEU (Papineni et al., 2002a) scores are computed using the official NIST evaluation script[8].

Table A.2 lays out the BLEU scores as well as the model sizes of the Moses and Kriya phrase tables. As shown, the Hierarchical phrase-based system always has larger models (unfiltered phrase table size), ranging between 342.2% and 635.5% of their phrase-based counterparts. In terms of BLEU scores, Kriya results in higher BLEU scores in all the language pairs with the best improvement coming for Chinese-English, confirming the results of (Chiang, 2007b). Further, Kriya achieves statistically significant improvements for Arabic-English and English-French experiments.

As mentioned earlier, the huge model size of the Hiero systems slow down decoding and earlier research has proposed two different approaches for this: Shallow-n decoding as opposed to the full decoding restricts the depth of non-glue hierarchical rules in the derivation. Orthogonal to this, more compact models that are substantially smaller than

[8] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v12.pl

Language Pair	**Original (2 NT)/ Sh-1**	**Compact (1 NT)/ Full**	
	BLEU	*BLEU*	*Model size*
English-Spanish	27.70	**28.15**	351.3 (55.5%)
English-French	23.22	**23.48**	290.3 (55.8%)
French-English	**26.67**	**26.66**	248.2 (56.4%)
Arabic-English	37.15	**37.71**	161.4 (49.0%)
Chinese-English	24.04	25.25	154.2 (53.9%)

Table A.3: Shallow-1 decoding vs. Compact (1 NT) model - Bold face indicates BLEU scores comparable to the original Hiero model in Table A.2. Size of the compact 1 NT model as a % of original Hiero model is given within the brackets.

the full Hiero models can be used with full decoding. In this experiment we compare the basic variants of these two approaches in terms of BLEU scores and model size.

In Shallow decoding setting, we use shallow-1 thus restricting the hierarchical rules in the grammar to directly rewrite into terminal rules with the glue rules freely combining the hierarchical rules. We compare this with a simpler Hiero grammar consisting of one non-terminal and this generally results in a compact model compared to original Hiero model. Note that these two ideas are orthogonal and hence can be combined; however we generally find them to result in poor performance and so we ignore the combination experiments here.

The experimental results are summarised in Tables A.3 and A.4. We find the simpler model consisting of one non-terminal employing full decoding to be competitive to the full model for closer language pairs such as French-English and Arabic-English at the same time clocking higher decoding speed. However, we see a reduction in the BLEU score for Chinese-English as has also been found by (Zollmann et al., 2008). We thus hypothesize that 1 NT models have the same expressive power as the regular Hiero models (with 2 non-terminals), at least for languages with little syntactic divergence. They also reduce the model size almost by half achieving a highest reduction of 51% for Arabic-English.

Shallow-1 decoding achieves highest decoding speed among the three but suffers a small reduction in the BLEU score except for French-English and incurs a larger reduction of 1.9 BLEU points for Chinese-English. It is three times faster than full decoding and twice faster than the 1 NT model. Higher order shallow decoding (not shown here), for example shallow-2 for Arabic-English and shallow-3 for Chinese-English achieve competitive performance but

Model	**Decoding level**	**Decoding time**
Original (2 NT)	Full	0.71
Original (2 NT)	Shallow-1	0.24
Compact (1 NT)	Full	0.50

Table A.4: Kriya Decoding time (in secs/word) for Chinese-English translation

shallow-3 case suffers substatial reduction in decoding speed and is only marginally faster than full decoding.

A.6 Future Directions

Kriya continues to be in active development and we are planning to add several new features. Currently, it supports TORQUE cluster for parallelizing training and optimization processes. We are currently working on to support MapReduce framework, specifically Hadoop cluster. We will also be adding PRO (Hopkins and May, 2011) in addition to the well-known MERT algorithm for optimizing feature weights. In terms of Kriya decoder, we are exploring a new left-to-right decoder similar to (Watanabe et al., 2006) in order to take advantage of its straight-forward language model integration and achieve a faster decoding time. Furthermore, we are adding an ensemble framework for decoding with multiple translation and/or language models, which can particularly be useful in scenarios such as domain adaptation and multi-source translation. Finally, efficient alternatives for heuristic rule extraction using Bayesian models is also in the developmental pipeline.

A.7 Conclusion

In this chapter, we briefly described *Kriya* - a new implementation of Hierarchical phrase-based systems having novel features and achieving competitive performances in several language pairs. Kriya's grammar extractor can efficiently extract Hiero grammars even from large training sets and additionally supports extraction of several compact Hiero grammar variants. The decoder currently uses CKY-based decoding, while we plan to extend this to do left-to-right decoding for achieving speedup. Kriya is under active development [9] and several new features are being planned with specific focus on Bayesian models for extracting compact grammars, ensemble decoding, support for MapReduce framework and so on. We also presented experimental results comparing the variants of Hiero decoding and different Hiero grammars, specifically in the context of Kriya for five language pairs.

[9] Kriya can be downloaded from `https://github.com/sfu-natlang/Kriya`

Bibliography

[Alexandrescu and Kirchhoff, 2009] Andrei Alexandrescu and Katrin Kirchhoff. 2009. Graph-based learning 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*, NAACL '09, pages 119–127, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Bacchiani and Roark, 2003] M. Bacchiani and B. Roark. 2003. Unsupervised language model adaptation. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, volume 1, pages I–224 – I–227 vol.1, april.

[Bangalore et al., 2001] Srinivas Bangalore, German Bordel, and Giuseppe Riccardi. 2001. Computing Consensus Translation from Multiple Machine Translation Systems. In *ASRU*.

[Bertoldi and Federico, 2009] Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 182–189, Stroudsburg, PA, USA. ACL.

[Bertoldi et al., 2008] N. Bertoldi, M. Barbaiani, M. Federico, and R. Cattoni. 2008. Phrase-based statistical machine translation with pivot languages. *Proceeding of IWSLT*, pages 143–149.

[Boitet, 1988] C. Boitet. 1988. Pros and cons of the pivot and transfer approaches in multilingual machine translation. *Maxwell et al.(1988)*, pages 93–106.

[Breiman, 1996a] Leo Breiman. 1996a. Bagging predictors. *Machine Learning*, 24(2):123–140, August.

[Breiman, 1996b] Leo Breiman. 1996b. Stacked regressions. *Machine Learning*, 24(1):49–64, July.

[Brown et al., 1990] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Comput. Linguist.*, 16:79–85, June.

[Brown et al., 1993] Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

[Callison-Burch et al., 2006] C. Callison-Burch, P. Koehn, and M. Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24. Association for Computational Linguistics.

[Chapelle et al., 2006] O. Chapelle, B. Schölkopf, and A. Zien, editors. 2006. *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

[Chiang et al., 2008] David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ACL.

[Chiang, 2005a] David Chiang. 2005a. A hierarchical phrase-based model for statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, Morristown, NJ, USA. ACL.

[Chiang, 2005b] David Chiang. 2005b. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of Annual Meeting of Association of Computational Linguistics*, pages 263–270.

[Chiang, 2007a] David Chiang. 2007a. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

[Chiang, 2007b] David Chiang. 2007b. Hierarchical phrase-based translation. *Computational Linguistics*, 33.

[Civera and Juan, 2007] Jorge Civera and Alfons Juan. 2007. Domain adaptation in statistical machine translation with mixture modelling. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 177–180, Stroudsburg, PA, USA. ACL.

[Clark et al., 2011] Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 176–181. ACL.

[Clarkson and Robinson, 1997] P. Clarkson and A. Robinson. 1997. Language model adaptation using mixtures and an exponentially decaying cache. In *Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)-Volume 2 - Volume 2*, ICASSP '97, pages 799–, Washington, DC, USA. IEEE Computer Society.

[Cohn and Lapata, 2007] Trevor Cohn and Mirella Lapata. 2007. Machine translation by triangulation: Making effective use of multi-parallel corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 728–735, Prague, Czech Republic, June. Association for Computational Linguistics.

[Crammer et al., 2006] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

[Dagan et al., 1999] Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Mach. Learn.*, 34(1-3):43–69, February.

[Daumé and Jagarlamudi, 2011] Hal Daumé, III and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 407–412, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Daumé and Marcu, 2006] Hal Daumé, III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *J. Artif. Int. Res.*, 26:101–126, May.

[de Gispert and Marino, 2006] A. de Gispert and J.B. Marino. 2006. Catalan-english statistical machine translation without parallel corpus: bridging through spanish. In *Proc. of 5th International Conference on Language Resources and Evaluation (LREC)*, pages 65–68.

[de Gispert et al., 2010] Adrià de Gispert, Gonzalo Iglesias, Graeme Blackwood, Eduardo R. Banga, and William Byrne. 2010. Hierarchical phrase-based translation with weighted finite-state transducers and Shallow-n grammars. *Computational Linguistics*, 36.

[DeNeefe and Knight, 2009] Steve DeNeefe and Kevin Knight. 2009. Synchronous tree adjoining machine translation. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 727–736, Morristown, NJ, USA. Association for Computational Linguistics.

[DeNero et al., 2009] John DeNero, David Chiang, and Kevin Knight. 2009. Fast consensus decoding over translation forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 567–575, Stroudsburg, PA, USA. Association for Computational Linguistics.

[DeNero et al., 2010] John DeNero, Shankar Kumar, Ciprian Chelba, and Franz Och. 2010. Model combination for machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 975–983, Stroudsburg, PA, USA. ACL.

[Denkowski and Lavie, 2011] Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*.

[Duan et al., 2009] Nan Duan, Mu Li, Tong Xiao, and Ming Zhou. 2009. The feature subspace method for smt system combination. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1096–1104, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Duan et al., 2010] Nan Duan, Hong Sun, and Ming Zhou. 2010. Translation model generalization using probability averaging for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 304–312, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Dunning, 1993] Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74, March.

[Dyer et al., 2010] Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the Annual Meeting of Association of Computational Linguistics 2010 System Demonstrations track*, ACLDemos '10, pages 7–12.

[Eck et al., 2004] Matthias Eck, Stephan Vogel, and Alex Waibel. 2004. Language model adaptation for statistical machine translation based on information retrieval. In *In Proceedings of LREC*.

[Eidelman et al., 2012] Vladimir Eidelman, Jordan Boyd-Graber, and Philip Resnik. 2012. Topic models for dynamic translation model adaptation. In *Association for Computational Linguistics*.

[Eisner, 2003] Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 205–208, Morristown, NJ, USA. Association for Computational Linguistics.

[F. T. Martins et al., 2008] André F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Honolulu, Hawaii, October. Association for Computational Linguistics.

[Federico et al., 2008] Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. Irstlm: an open source toolkit for handling large scale language models. In *INTERSPEECH 2008, 9th Annual Conference of the International Speech Communication Association*, pages 1618–1621. ISCA.

[Fišer and Ljubešić, 2011] Darja Fišer and Nikola Ljubešić. 2011. Bilingual lexicon extraction from comparable corpora for closely related languages. In *RANLP*, pages 125–131.

[Foster and Kuhn, 2007] George Foster and Roland Kuhn. 2007. Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 128–135, Stroudsburg, PA, USA. ACL.

[Foster et al., 2010] George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 451–459, Stroudsburg, PA, USA. ACL.

[Fung and Yee, 1998] Pascale Fung and Lo Yuen Yee. 1998. An ir approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, pages 414–420. Association for Computational Linguistics.

[Galley et al., 2004] Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of the Human Language Technology and North American chapter of the Association for Computational Linguistics Conference (HLT-NAACL,)*, pages 273–280.

[Galley et al., 2006] Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 961–968, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Garera et al., 2009] Nikesh Garera, Chris Callison-Burch, and David Yarowsky. 2009. Improving translation lexicon induction from monolingual corpora via dependency contexts and part-of-speech equivalences. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 129–137, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Gildea, 2003] Daniel Gildea. 2003. Loosely tree-based alignment for machine translation. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 80–87, Morristown, NJ, USA. Association for Computational Linguistics.

[Gollins and Sanderson, 2001] T. Gollins and M. Sanderson. 2001. Improving cross language retrieval with triangulated translation. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 90–95. ACM.

[Goyal et al., 2012] Amit Goyal, Hal Daume III, and Raul Guerra. 2012. Fast Large-Scale Approximate Graph Construction for NLP. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '12.

[Habash, 2008] Nizar Habash. 2008. Four techniques for online handling of out-of-vocabulary words in arabic-english statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 57–60. Association for Computational Linguistics.

[Haghighi et al., 2008] Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *ACL*, pages 771–779.

[Harris, 1954] Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

[He et al., 2009] Zhongjun He, Yao Meng, and Hao Yu. 2009. Discarding monotone composed rule for hierarchical phrase-based statistical machine translation. In *Proceedings of the 3rd International Universal Communication Symposium*, pages 25–29.

[Heafield, 2011a] Kenneth Heafield. 2011a. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197.

[Heafield, 2011b] Kenneth Heafield. 2011b. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197.

[Hildebrand and Vogel, 2009] Almut Silja Hildebrand and Stephan Vogel. 2009. CMU system combination for WMT'09. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 47–50, Stroudsburg, PA, USA. ACL.

[Hildebrand et al., 2005] Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the 10th EAMT 2005*, Budapest, Hungary, May.

[Hinton, 1999] Geoffrey E. Hinton. 1999. Products of experts. In *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)*, volume 1, pages 1–6.

[Ho, 1998] Tin Kam Ho. 1998. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, August.

[Hoeting et al., 1999] Jennifer A. Hoeting, David Madigan, Adrian E. Raftery, and Chris T. Volinsky. 1999. Bayesian Model Averaging: A Tutorial. *Statistical Science*, 14(4):382–401.

[Hopkins and May, 2011] Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362. Association for Computational Linguistics.

[Huang and Chiang, 2007] Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151. Association for Computational Linguistics.

[Huang et al., 2011] Chung-Chi Huang, Ho-Ching Yen, Ping-Che Yang, Shih-Ting Huang, and Jason S Chang. 2011. Using sublexical translations to handle the oov problem in machine translation. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(3):16.

[Iglesias et al., 2009] Gonzalo Iglesias, Adrià de Gispert, Eduardo R. Banga, and William Byrne. 2009. Rule filtering by pattern for efficient hierarchical translation. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pages 380–388.

[Jiang and Zhai, 2007] Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic, June. ACL.

[Koehn and Hoang, 2007] Philipp Koehn and Hieu Hoang. 2007. Factored translation models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic. Association for Computational Linguistics.

[Koehn and Knight, 2002] Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition - Volume 9*, ULA '02, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Koehn and Schroeder, 2007] Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 224–227, Stroudsburg, PA, USA. ACL.

[Koehn et al., 2003] Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 127–133, Edmonton, May. NAACL.

[Koehn et al., 2007a] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007a. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. ACL.

[Koehn et al., 2007b] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007b. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics.

[Koehn, 2005] P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5.

[Koehn, 2010] Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition.

[Kumar and Byrne, 2004] Shankar Kumar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 169–176, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

[Kumar et al., 2007] Shankar Kumar, Franz Josef Och, and Wolfgang Macherey. 2007. Improving word alignment with bridge languages. In *EMNLP-CoNLL*, pages 42–50. ACL.

[Kumar et al., 2009] Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 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 ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 163–171, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Lagarda and Casacuberta, 2008] Antonio Lagarda and Francisco Casacuberta. 2008. Applying boosting to statistical machine translation. In *Annual Meeting of European Association for Machine Translation (EAMT)*, pages 88–96.

[Laws et al., 2010] Florian Laws, Lukas Michelbacher, Beate Dorow, Christian Scheible, Ulrich Heid, and Hinrich Schütze. 2010. A linguistically grounded graph model for bilingual lexicon extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 614–622, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Li et al., 2009a] Zhifei Li, Chris Callison-Burch, Chris Dyer, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. 2009a. Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 135–139. Association for Computational Linguistics.

[Li et al., 2009b] Zhifei Li, Jason Eisner, and Sanjeev Khudanpur. 2009b. Variational decoding for statistical machine translation. In *Proceedings of the Joint Conference of the*

47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2, ACL '09, pages 593–601, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Li et al., 2010] Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Ann Irvine, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Ziyuan Wang, Jonathan Weese, and Omar Zaidan. 2010. Joshua 2.0: A toolkit for parsing-based machine translation with syntax, semirings, discriminative training and other goodies. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 133–137. Association for Computational Linguistics.

[Lin, 1998] Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2*, ACL '98, pages 768–774, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Liu and Nocedal, 1989] Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528.

[Liu et al., 2009] Yang Liu, Haitao Mi, Yang Feng, and Qun Liu. 2009. Joint decoding with multiple translation models. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 576–584, Stroudsburg, PA, USA. ACL.

[Liu et al., 2012] Shujie Liu, Chi-Ho Li, Mu Li, and Ming Zhou. 2012. Learning translation consensus with structured label propagation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 302–310, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Mann and Yarowsky, 2001] Gideon S. Mann and David Yarowsky. 2001. Multipath translation lexicon induction via bridge languages. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA.

[Marcu and Wong, 2002] Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 133–139, Morristown, NJ, USA. Association for Computational Linguistics.

[Marton et al., 2009] Yuval Marton, Chris Callison-Burch, and Philip Resnik. 2009. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 381–390, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Matsoukas et al., 2009] Spyros Matsoukas, Antti-Veikko I. Rosti, and Bing Zhang. 2009. Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 708–717, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Melamed et al., 2004] I. Dan Melamed, Giorgio Satta, and Benjamin Wellington. 2004. Generalized multitext grammars. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 661–668, Morristown, NJ, USA. Association for Computational Linguistics.

[Nelder and Mead, 1965] John A. Nelder and Roger Mead. 1965. A simplex method for function minimization. *Computer Journal*, 7:308–313.

[Nivre and McDonald, 2008] Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio, June. Association for Computational Linguistics.

[Och and Ney, 2000] F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the ACL*, pages 440–447, Hongkong, China, October.

[Och and Ney, 2002] Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the Annual Meeting of Association of Computational Linguistics*, pages 295–302.

[Och and Ney, 2003] Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.

[Och and Ney, 2004] Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

[Och et al., 1999] Franz Josef Och, Christoph Tillmann, Hermann Ney, and Lehrstuhl Fiir Informatik. 1999. Improved alignment models for statistical machine translation. In *University of Maryland, College Park, MD*, pages 20–28.

[Och, 2003a] Franz Josef Och. 2003a. Minimum error rate training for statistical machine translation. In *Proceedings of the 41th Annual Meeting of the ACL*, Sapporo, July. ACL.

[Och, 2003b] Franz Josef Och. 2003b. Minimum error rate training in statistical machine translation. In *Proceedings of the Annual Meeting of Association of Computational Linguistics*, pages 160–167.

[Papineni et al., 2002a] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002a. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of Association of Computational Linguistics*, pages 311–318.

[Papineni et al., 2002b] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002b. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Petrov, 2010] Slav Petrov. 2010. Products of random latent variable grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 19–27, Stroudsburg, PA, USA. ACL.

[Rao and Yarowsky, 2009] Delip Rao and David Yarowsky. 2009. Ranking and semi-supervised classification on large scale graphs using map-reduce. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, TextGraphs-4. Association for Computational Linguistics.

[Rapp, 1995] Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 320–322. Association for Computational Linguistics.

[Rapp, 1999] Reinhard Rapp. 1999. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 519–526. Association for Computational Linguistics.

[Ravichandran et al., 2005] Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and NLP: Using locality sensitive hash functions for high speed noun clustering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 622–629, Ann Arbor, Michigan, June. Association for Computational Linguistics.

[Riedmiller and Braun, 1993] Martin Riedmiller and Heinrich Braun. 1993. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *IEEE International Conference on Neural Networks*, pages 586–591.

[Rosti et al., 2007] Antti-Veikko I Rosti, Necip Fazil Ayan, Bing Xiang, Spyros Matsoukas, Richard Schwartz, and Bonnie Dorr. 2007. Combining outputs from multiple machine translation systems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 228–235.

[Sadat et al., 2005] Fatiha Sadat, Howard Johnson, Akakpo Agbago, George Foster, Joel Martin, and Aaron Tikuisis. 2005. Portage: A phrase-based machine translation system. In *In Proceedings of the ACL Worskhop on Building and Using Parallel Texts, Ann Arbor*. ACL.

[Sang, 2002] Erik F. Tjong Kim Sang. 2002. Memory-based shallow parsing. *J. Mach. Learn. Res.*, 2:559–594, March.

[Sankaran et al., 2011] Baskaran Sankaran, Gholamreza Haffari, and Anoop Sarkar. 2011. Bayesian extraction of minimal scfg rules for hierarchical phrase-based translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 533–541.

[Sankaran et al., 2012] Baskaran Sankaran, Majid Razmara, and Anoop Sarkar. 2012. Kriya – an end-to-end hierarchical phrase-based mt system. *The Prague Bulletin of Mathematical Linguistics*, 97(97), April.

[Schafer and Yarowsky, 2002] Charles Schafer and David Yarowsky. 2002. Inducing translation lexicons via diverse similarity measures and bridge languages. In *proceedings of the 6th conference on Natural language learning - Volume 20*, COLING-02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Schapire, 1990] Robert E. Schapire. 1990. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, July.

[Schubert, 1988] K. Schubert. 1988. Implicitness as a guiding principle in machine translation. In *Proceedings of the 12th conference on Computational linguistics-Volume 2*, pages 599–601. Association for Computational Linguistics.

[Schütze and Pedersen, 1997] Hinrich Schütze and Jan O. Pedersen. 1997. A cooccurrence-based thesaurus and two applications to information retrieval. *Inf. Process. Manage.*, 33(3):307–318, May.

[Seymore and Rosenfeld, 1997] Kristie Seymore and Ronald Rosenfeld. 1997. Using story topics for language model adaptation. In George Kokkinakis, Nikos Fakotakis, and Evangelos Dermatas, editors, *EUROSPEECH*. ISCA.

[Shieber and Schabes, 1990] Stuart M. Shieber and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th conference on Computational linguistics*, pages 253–258, Morristown, NJ, USA. Association for Computational Linguistics.

[Smith et al., 2005] Andrew Smith, Trevor Cohn, and Miles Osborne. 2005. Logarithmic opinion pools for conditional random fields. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 18–25, Stroudsburg, PA, USA. ACL.

[Song et al., 2011] Linfeng Song, Haitao Mi, Yajuan Lv, and Qun Liu. 2011. Bagging-based system combination for domain adaptation. In *Proceedings of MT-Summit XIII*, Xiamen, China, September.

[Stein et al., 2011] Daniel Stein, David Vilar, Stephan Peitz, Markus Freitag, Matthias Huck, and Hermann Ney. 2011. A guide to jane, an open source hierarchical translation toolkit. In *The Prague Bulletin of Mathematical Linguistics, No. 95*, pages 5–18.

[Stolcke, 2002a] Andreas Stolcke. 2002a. SRILM – an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286.

[Stolcke, 2002b] Andreas Stolcke. 2002b. SRILM – an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286.

[Surdeanu and Manning, 2010] Mihai Surdeanu and Christopher D. Manning. 2010. Ensemble models for dependency parsing: cheap and good? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 649–652, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Talbot and Osborne, 2007] David Talbot and Miles Osborne. 2007. Randomised language modelling for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 512–519, Prague, Czech Republic. Association for Computational Linguistics.

[Talukdar and Crammer, 2009] Partha Pratim Talukdar and Koby Crammer. 2009. New Regularized Algorithms for Transductive Learning. In *European Conference on Machine Learning (ECML-PKDD).*

[Talukdar et al., 2008] Partha Pratim Talukdar, Joseph Reisinger, Marius Paşca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08.

[Tamura et al., 2012] Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2012. Bilingual lexicon extraction from comparable corpora using label propagation. In *EMNLP-CoNLL*, pages 24–36.

[Terra and Clarke, 2003] Egidio L. Terra and Charles L. A. Clarke. 2003. Frequency estimates for statistical word similarity measures. In *HLT-NAACL*.

[Tiedemann, 2009] Jorg Tiedemann. 2009. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia.

[Tomeh et al., 2010] Nadi Tomeh, Alexandre Allauzen, Guillaume Wisniewski, and François Yvon. 2010. Refining word alignment with discriminative training. In *Proceedings of The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010).*

[Ueffing et al., 2007] Nicola Ueffing, Gholamreza Haffari, and Anoop Sarkar. 2007. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 25–32, Prague, Czech Republic, June. ACL.

[Utiyama and Isahara, 2007] M. Utiyama and H. Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *Proceedings of NAACL-HLT*, volume 7, pages 484–491.

[Vanden Berghen and Bersini, 2005] Frank Vanden Berghen and Hugues Bersini. 2005. CONDOR, a new parallel, constrained extension of powell's UOBYQA algorithm: Experimental results and comparison with the DFO algorithm. *Journal of Computational and Applied Mathematics*, 181:157–175, September.

[Vilar et al., 2010] David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. 2010. Jane: open source hierarchical translation, extended with reordering and lexicon models. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 262–270. Association for Computational Linguistics.

[Vogel et al., 1996] Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics - Volume 2*, pages 836–841, Morristown, NJ, USA. Association for Computational Linguistics.

[Voorhees, 1999] Ellen M. Voorhees. 1999. TREC-8 Question Answering Track Report. In *Proceedings of the 8th Text Retrieval Conference*, pages 77–82.

[Wang et al., 2006] H. Wang, H. Wu, and Z. Liu. 2006. Word alignment for languages with scarce resources using bilingual corpora of other language pairs. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 874–881. Association for Computational Linguistics.

[Watanabe et al., 2006] Taro Watanabe, Hajime Tsukada, and Hideki Isozaki. 2006. Left-to-right target generation for hierarchical phrase-based translation. In *Proceedings of the 21st International Conference on Computational Linguistics (COLING)*, pages 777–784. Association for Computational Linguistics.

[Weese et al., 2011] Jonathan Weese, Juri Ganitkevitch, Chris Callison-Burch, Matt Post, and Adam Lopez. 2011. Joshua 3.0: Syntax-based machine translation with the thrax grammar extractor. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 478–484. Association for Computational Linguistics.

[Wolpert, 1992] David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

[Wu and Wang, 2007] H. Wu and H. Wang. 2007. Pivot language approach for phrase-based statistical machine translation. *Machine Translation*, 21(3):165–181.

[Wu, 1995] Dekai Wu. 1995. Stochastic inversion transduction grammars, with application to segmentation, bracketing, and alignment of parallel corpora. *IJCAI-95: 14th Intl. Joint Conf. on Artificial Intelligence,*, pages 1328–1335.

[Wu, 1997] Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Comput. Linguist.*, 23(3):377–403.

[Xiao et al., 2010] Tong Xiao, Jingbo Zhu, Muhua Zhu, and Huizhen Wang. 2010. Boosting-based system combination for machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 739–748, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Xiao et al., 2013] Tong Xiao, Jingbo Zhu, and Tongran Liu. 2013. Bagging and boosting statistical machine translation systems. *Artificial Intelligence*, 195:496–527, February.

[Yamada and Knight, 2001] Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *ACL*, pages 523–530.

[Yasuda et al., 2008] Keiji Yasuda, Ruiqiang Zhang, Hirofumi Yamamoto, and Eiichiro Sumita. 2008. Method of selecting training data to build a compact and efficient translation model. In *Third International Joint Conference on Natural Language Processing, IJCNLP 2008, Hyderabad, India,*, pages 655–660. The Association for Computer Linguistics.

[Zaidan, 2009] Omar F. Zaidan. 2009. Z-mert: A fully configurable open source tool for minimum error rate training of machine translation systems. *Prague Bulletin of Mathematical Linguistics.*

[Zhang et al., 2012] Jiajun Zhang, Feifei Zhai, and Chengqing Zong. 2012. Handling unknown words in statistical machine translation from a new perspective. In *Natural Language Processing and Chinese Computing*, pages 176–187. Springer.

[Zollmann and Venugopal, 2006] Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *StatMT '06: Proceedings of the Workshop on Statistical Machine Translation*, pages 138–141, Morristown, NJ, USA. Association for Computational Linguistics.

[Zollmann et al., 2008] Andreas Zollmann, Ashish Venugopal, Franz Och, and Jay Ponte. 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical mt. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1145–1152.