Learning from Noisy Data in Statistical Machine Translation

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Informatik

des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Mohammed Mediani

aus Adrar

Tag der mündlichen Prüfung: 30.01.2017

Erster Gutachter: Prof. Dr. Alexander Waibel

Zweiter Gutachter: Prof. Dr.-Ing. Rüdiger Dillmann

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe, sowie dass ich die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des KIT, ehem. Universität Karlsruhe (TH), zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, den 28.11.2016

Mohammed Mediani

Abstract

Since its emergence in the early 90s, Statistical Machine Translation (SMT) has attracted great attention from both academia and industry. This is a logical consequence of its outstanding success in many real-life applications. In a great part, this stems from the quality of the translations it presents compared to the low human labor and the short development time required for production.

The development of SMT systems for a new language pair, however, builds on top of a critical assumption: the availability of training data. This fact has been continuously urging researchers from academic and industrial environments to find new cheap data resources providing satisfactory amounts of training data with good quality. The Internet turns out to be such a great resource with an additional interesting feature: the collection process can be automated. The usefulness of such data has been shown in the annual international SMT evaluation campaigns (e.g. WMT and IWSLT). For instance, more than half of the parallel training data available for German-English systems is automatically crawled from the Web. Obviously, this figure gets even larger for monolingual data. It is noteworthy that, in these evaluations, the training corpora are distributed sentence-aligned. However, it is not uncommon to encounter many pair of sentences which are not translations of each other. Mostly, this is a consequence of the imperfection of the sentence alignment process. Another frequent problem with these corpora, is that some unusual sequences will result from the use of automatic preprocessors. For instance, the difference in encodings may lead the preprocessor to remove or split some words, creating thus a non-natural flow of words.

The two aforementioned scenarios are only a small sample of examples which show that the automatically harvested data is not always of high quality.

We term such examples noise. Generally speaking, “noise” means any piece of data whose presence in the model is nonessential or would rather hurt the system’s performance. In this sense, incorrect parallel pairs and sentences which are not likely to be encountered in the test can be considered as good examples of noise. The existence of significant amounts of noise in the training data is likely to degrade the system’s performance while adding considerable computing overhead to the training process. On the other hand, the manual cleaning becomes extremely expensive, if at all possible.

In this work, we propose methods that are capable of reducing the negative effect of noise on an SMT system and thus improving its performance. We approach the problem at two different stages of the learning process: at preprocessing and during modeling. At the preprocessing level, we investigate two ways of enhancing the statistical models by removing parts of the training data. On the other hand, at modeling we explore different ways to weight data instances according to their usefulness.

At the preprocessing, we, first, show the effect of removing the false positives from a parallel corpus; i.e. pairs which are thought to be correct translations while in fact they are not. To do so, we rely on an existing small seed “clean” corpus to design a classifier-based filter. With the help of several lexical features we are able to reliably decide whether a given pair is true or false positive prior to modeling. Among these lexical features, the most important is a bilingual lexicon obtained from the clean corpus. Different heuristics are implemented in the extraction of this bilingual lexicon, which lead to improved performance.

We, then, approach the problem of extracting the most useful parts of the training data. In this task we rank the data based on its relatedness to the targeted domain under the assumption of the existence of a good representative tuning data set. Since such representative is typically limited in size, we exploit word similarities to extend the coverage of the limited indomain representative.

The preexistence of the word similarities in the aforementioned task is crucial. We present several ways of automatically deriving such similarities from monolingual and bilingual corpora. It is shown that the similarities obtained from bilingual corpora are usually of higher quality. However, as the bilingual data faces the data sparsity problem much more than their monolingual counterpart, we propose two approaches to cope with this limitation. First, we explore a technique to integrate the information from the bilingual data into word representations learned from the monolingual corpora. The performance of the similarities computed from the combined representations is, therefore, considerably improved. Second, the similarities could be learned from a richer language pair, as long as the target language of this latter matches the target language of the task at hand.

At the modeling stage, we deal with the noise by weighting the training data based on its "cleanliness". We automatically identify the less reliable sequences and penalize them, biasing thus the model to rely more on the clean data. However, another problem arises as soon as we have real-valued modified counts, resulting from applying the weights. The problem, here, lies in the fact that our estimation of the different models uses smoothing. The smoothing of a probability distribution allows for assigning a non null probability to any event. All commonly used smoothing techniques in our models assume integral counts, which is violated by our weighting. We tackle this, by deriving a smoothing technique which can handle fractional counts.

The size of training data becomes an issue as soon as we start dealing with corpora of large volumes. Here, one faces two major difficulties: the lengthy training time and the memory limitation. We solve the first problem by using a hybrid parallel model. The model comprises distributed processing units, each of which implements a shared-memory parallelism to speedup the computationally expensive operations. On the other hand, to overcome the memory limitations, we use special external memory data structures and algorithms. This allows more efficient training of extremely large models on a resource-constrained hardware.

Zusammenfassung

In dieser Arbeit wurden Methoden entwickelt, die in der Lage sind die negativen Effekte von verrauschten Daten in SMT Systemen zu senken und dadurch die Leistung des Systems zu steigern. Hierbei wird das Problem in zwei verschiedenen Schritten des Lernprozesses behandelt: Bei der Vorverarbeitung und während der Modellierung. Bei der Vorverarbeitung werden zwei Methoden zur Verbesserung der statistischen Modelle durch die Erhöhung der Qualität von Trainingsdaten entwickelt. Bei der Modellierung werden verschiedene Möglichkeiten vorgestellt, um Daten nach ihrer Nützlichkeit zu gewichten.

Zunächst wird der Effekt des Entfernens von False-Positives vom Parallel Corpus gezeigt. Ein Parallel Corpus besteht aus einem Text in zwei Sprachen, wobei jeder Satz einer Sprache mit dem entsprechenden Satz der anderen Sprache gepaart ist. Hierbei wird vorausgesetzt, dass die Anzahl der Sätzen in beiden Sprachversionen gleich ist. False-Positives in diesem Sinne sind Satzpaare, die im Parallel Corpus gepaart sind aber keine Übersetzung voneinander sind. Um diese zu erkennen wird ein kleiner und fehlerfreier paralleler Corpus (Clean Corpus) vorausgesetzt. Mit Hilfe verschiedenen lexikalischen Eigenschaften werden zuverlässig False-Positives vor der Modellierungsphase gefiltert. Eine wichtige lexikalische Eigenschaft hierbei ist das vom Clean Corpus erzeugte bilinguale Lexikon. In der Extraktion dieses bilingualen Lexikons werden verschiedene Heuristiken implementiert, die zu einer verbesserten Leistung führen.

Danach betrachten wir das Problem vom Extrahieren der nützlichsten Teile der Trainingsdaten. Dabei ordnen wir die Daten basierend auf ihren Bezug zur Zieldomaine. Dies geschieht unter der Annahme der Existenz eines guten

repräsentativen Tuning Datensatzes. Da solche Tuning Daten typischerweise beschränkte Größe haben, werden Wortähnlichkeiten benutzt um die Abdeckung der Tuning Daten zu erweitern.

Die im vorherigen Schritt verwendeten Wortähnlichkeiten sind entscheidend für die Qualität des Verfahrens. Aus diesem Grund werden in der Arbeit verschiedene automatische Methoden zur Ermittlung von solche Wortähnlichkeiten ausgehend von monoligual und biligual Corpora vorgestellt. Interessanterweise ist dies auch bei beschränkten Daten möglich, indem auch monolinguale Daten, die in großen Mengen zur Verfügung stehen, zur Ermittlung der Wortähnlichkeit herangezogen werden. Bei bilingualen Daten, die häufig nur in beschränkter Größe zur Verfügung stehen, können auch weitere Sprachpaare herangezogen werden, die mindestens eine Sprache mit dem vorgegebenen Sprachpaar teilen.

Im Modellierungsschritt behandeln wir das Problem mit verrauschten Daten, indem die Trainingsdaten anhand der Güte des Corpus gewichtet werden. Wir benutzen Statistik signifikante Messgrößen, um die weniger verlässlichen Sequenzen zu finden und ihre Gewichtung zu reduzieren. Ähnlich zu den vorherigen Ansätzen, werden Wortähnlichkeiten benutzt um das Problem bei begrenzten Daten zu behandeln. Ein weiteres Problem tritt allerdings auf sobald die absolute Häufigkeiten mit den gewichteten Häufigkeiten ersetzt werden. In dieser Arbeit werden hierfür Techniken zur Glättung der Wahrscheinlichkeiten in dieser Situation entwickelt.

Die Größe der Trainingsdaten werden problematisch sobald man mit Corpora von erheblichem Volumen arbeitet. Hierbei treten zwei Hauptschwierigkeiten auf: Die Länge der Trainingszeit und der begrenzte Arbeitsspeicher. Für das Problem der Trainingszeit wird ein Algorithmus entwickelt, der die rechenaufwendigen Berechnungen auf mehrere Prozessoren mit gemeinsamem Speicher ausführt. Für das Speicherproblem werden speziale Datenstrukturen und Algorithmen für externe Speicher benutzt. Dies erlaubt ein effizientes Training von extrem großen Modellne in Hardware mit begrenztem Speicher.

To my mother and to the soul of my father

Now, that I am parent myself, I realize more than ever that nothing I do may count as a reward to the efforts they invested in me

Acknowledgements

I have always heard people saying “time flies when you are having fun”. This turned out true, at least, for my time in the KIT’s Interactive Systems Lab (ISL). Seven years have flown like a flash of light. The environment in the ISL is able to take one’s awareness of the flow of seemingly such a long period. In ISL I found myself in a genuinely inspiring environment, surrounded by very active and highly skilled researchers. Over the time I spent among them, I have never felt down or helpless with any problem I experienced. This was never limited to scientific questions, but also included every day life’s issues. Their cooperative interactions, their sympathetic spirits, and their tolerance for cultural differences make ISL a unique research environment with conditions hardly could be met elsewhere. Their company turned the lengthy and difficult process of preparing a PhD into enjoyable and unforgettable moments. I owe every one in ISL deep appreciations and heartful thanks.

The first person to thank in the team is my advisor and the ISL head, Professor Alex Waibel. Alex has amazed me with his wonderful personality. He is very smart and always busy with many things at the same time, yet it is hard to get him destructed from a productive scientific discussion. When it comes to science, he is very objective and critical, yet it is hard for me to picture his face but smiling. He travels so often, yet he has a magical talent to keep everyone in his team motivated and focused. I would like to thank Alex for granting me the great privilege of joining the team and giving me the time to develop this work. In all possible linguistic expressions, I can’t thank Alex enough for his generosity, for his encouragement, for his guidance, nor for his patience.

My deep gratitude and recognition go also to Professor Rüdiger Dillmann for accepting to co-advise this thesis, in spite of his busy schedule. His recommendations and advice are the most appreciable.

By virtue of being amongst the MT team, I would like to thank everyone of my colleagues. Everyone of them has helped me in a way or another. In particular, Jan Niehues had a great impact on the progression of this thesis. Jan was always there to comment, to argue, or to recommend directions followed in this thesis, since its early starting. He always surprised me with his ability to quickly understand any idea, to spot its pitfalls, and to suggest remedies. Jan has never turned me down when I asked for his feedback about any chapter in my thesis, even if it was in a short notice. My former colleague Yuqi Zhang taught me how to be easy-going, big-hearted, spontaneous, and yet effective and productive. I literally feel nostalgic to the days when we shared the same office. Teresa Herrmann was always spreading positive energy on her colleagues. Nobody can dispense with her talent in identifying errors in written documents. Everyone in the Lab likes Eunah Cho, she brings the smile on every face with her innocent humor. I will be missing Thanh-Le Ha's presentations in our weekly seminar. Le always chose the brilliant and breaking news papers in the field and made sure everyone has understood every detail.

I would also like to thank all the other members of the Lab, for the help they provided, for the discussions we had, for their overwhelming kindness and support. Thanks to all of them: Silke Dannenmaier, Sarah Fünfer, Jonas Gehring, Michael Heck, Silja Hildebrand, Klaus Joas, Kevin Kilgour, Narine Kokhlikyan, Florian Kraft, Bastian Krüger, Patricia Lichtblau, Christian Mohr, Markus Müller, Huy Van Nguyen, Quan Pham, Bao Quoc Nguyen, Thai Son Nguyen, Margit Rödder, Virginia Roth, Christian Saam, Rainer Saam, Maria Schmidt, Carsten Schnober, Isabel Slawik, Matthias Sperber, Sebastian Stüker, Yury Titov, Joshua Winebarger, Matthias Wölfel and Liang Guo Zhang.

My profound appreciations and gratitude also go to my friend Ahmed AbdelAli from QCRI. Ahmed was extremely kind and helpful by accepting to review my writings. His comments and suggestions always made a better output.

While being busy with the exhaustive PhD process, my lovely wife was busy with an even heavier full-time job: keeping our four-person family secure and happy. Without her hard work, my family and myself would not be able to live these moments. I can't thank her for carrying out the job gladly. I can't remember a single moment seeing her complaining or, even, unenthusiastic about it. I vividly appreciate her patience and tolerance for my late nights and work over the weekends. Honestly, she is the one who should be credited with any achievements I make.

Contents

List of Figures **vii**

List of Tables **ix**

Glossary **xi**

1 Introduction **1**

1.1 Noise in SMT data 4

1.1.1 Monolingual noise 4

1.1.2 Bilingual noise 6

1.1.3 How noisy is the Internet data? 7

1.2 Scope of the work 7

1.3 Main Contributions 8

1.4 Outlook 11

2 Foundations **15**

2.1 Statistical Association Measures 15

2.2 Statistical Machine Translation 19

2.2.1 Estimation 20

2.2.1.1 Preprocessing 21

2.2.1.2 Sentence Alignment 21

2.2.1.3 Word Alignment 23

2.2.1.4 Symmetrization 24

2.2.1.5 Phrase Extraction 25

2.2.1.6 Scoring 26

2.2.1.7 Language Models 26

2.2.1.8 Reordering Models 26
2.2.2 Prediction 27
2.2.2.1 Tuning the Log-Linear Weights 27
2.2.2.2 Decoding 28
2.2.2.3 Evaluation 28
2.2.3 Baseline 28
2.3 Related Work 31
2.3.1 Filtering Mismatched Sentence Pairs 32
2.3.2 Filtering as Data Selection 33
2.3.3 Weighting Training Data 34
2.3.4 Other Related Work 34
2.3.5 Efficient Phrase Scoring 35
2.4 Conclusions 36

3 Semantic Word Associations **37**
3.1 From Aligned Bilingual Corpora 38
3.2 From Monolingual Corpora 40
3.2.1 Matrix Factorization Approaches 41
3.2.2 Neural Network Approach: `word2vec` 42
3.2.3 Global Vectors: GloVe 43
3.2.4 Modified Global Vectors: mGloVe 44
3.2.4.1 Association-based Scoring 44
3.2.4.2 Context Interpretations 46
3.3 Implementation Issues 51
3.4 Evaluations 52
3.4.1 Reference Datasets 52
3.4.1.1 Semantic Similarity Datasets 52
3.4.1.2 Word Analogy Datasets 53
3.4.1.3 Syntactic Similarity Dataset 54
3.4.2 Semantic Associations from Word Alignments 56
3.4.3 Semantic Associations from Word Vectors 58
3.4.3.1 Baseline and Negative Sampling 59
3.4.3.2 Pair Scoring 61

3.4.3.3 Context Interpretation 62
3.4.3.4 Adding Information from Word Alignments 64
3.5 Conclusions 66

4 Parametric Smoothing of Probability Models 69
4.1 Traditional Smoothing Techniques 70
4.1.1 Kneser-Ney Smoothing (KN) 70
4.1.2 Witten-Bell smoothing (WB) 72
4.2 Parametric Discounting 72
4.2.0.1 The Discounting Function 75
4.2.0.2 Parameter Estimation 77
4.2.0.3 The Unseen Events 79
4.2.0.4 Discounting Fractional Counts 82
4.2.0.5 A Note about Complexity 83
4.3 Evaluations 84
4.4 Conclusions 88

5 Removing parallel noise 91
5.1 Introduction 91
5.2 Classifier-based filtering 94
5.2.1 Word alignment 95
5.2.2 Features 96
5.2.3 Lexicon extraction 100
5.2.4 Complexity 102
5.3 Evaluation 103
5.3.1 Intrinsic evaluation 104
5.3.2 Extrinsic evaluation 108
5.3.3 Large Scale Translation Experiments 109
5.3.4 Filtering without a Clean Seed Corpus 109
5.4 Conclusions 110

6 Semantically-Guided Data Selection **113**
6.1 Introduction 114
6.2 Selection Based on Cross-Entropy Difference 116
6.3 Enhancements 116
6.3.1 Drawing an OOD Representative Sample 116
6.3.2 Vocabulary Selection 117
6.4 Extended Cross-Entropy Selection 118
6.4.1 Lexicon of Semantic Associations 118
6.4.2 Extending the Selection LMs 119
6.5 Evaluations 120
6.6 Combination of all methods 123
6.7 Conclusions 124

7 Soft Selection: Noise-Based Weighting **127**
7.1 Weighting in Language Modeling 128
7.1.1 *n*-gram Level Weighting 128
7.1.2 Sentence Level Weighting 130
7.2 Weighting in Translation Model Training 132
7.2.1 Phrase Level Weighting 133
7.2.2 Sentence Pair Level Weighting 134
7.3 Evaluation 136
7.3.1 Weighting in Language Modeling 136
7.4 Conclusions 139

8 Parallel Phrase Scoring for Extra-large Corpora **141**
8.1 Introduction 142
8.2 Phrase Scoring 143
8.3 External Memory Sorting in STXXL 144
8.4 Software Architecture and Algorithms 146
8.4.1 Multithreaded Phrase Scoring 146
8.4.2 Hybrid Parallel Phrase Scoring 148
8.5 Evaluation 149
8.6 Conclusions 152

9 Conclusions and Outlook **153**

Appendices **159**

A Derivations of Formulae **161**

A.1 Count re-estimation from clusters 161

B Examples from Datasets **163**

B.1 Semantic Similarity Datasets 163

B.2 Word Analogy Datasets 164

B.3 Syntactic Similarity Dataset 164

C Example Outputs **165**

C.1 Examples of Semantic Associations 165

D Additional Results **167**

D.1 Additional Smoothing Results 167

References **171**

List of Figures

1.1 Parallel data available for different pairs in millions of pair of sentences . 2
1.2 Schematic summary of contributions 9

2.1 Contingency table for a word pair (w_1, w_2) 17
2.2 SMT Pipeline 20
2.3 Example of aligned sentence pair 25

3.1 Context weighting based on distance from the head word 46

4.1 Comparison of different discounts for KN-based and WB-based ξ 77

5.1 Extract from a noisy phrase table 92
5.2 Example of badly aligned sentences 93
5.3 Example of length-imbalance 97
5.4 Example of lexicon-based word alignment 97
5.5 Example of bad and good distance of alignments 100
5.6 Hard positive En $\rightarrow$ Fr examples 106

6.1 Diagrammatic representation of vocabularies of in- and out-of-domain sources 118

8.1 Multithreaded phrase scoring architecture 147
8.2 Hybrid phrase scoring architecture 148
8.3 Effect of the augmenting number of cores and number of nodes 151

List of Tables

1.1 Examples of noise in monolingual corpora 5
1.2 Examples of noise in bilingual corpora 6
1.3 Proportion of correct parallel sentences in the Crawl and Giga corpora . 7

2.1 Some important sources of the SMT corpora 21
2.2 Phrases resulting from alignments in Figure 2.3 25
2.3 Overview of the parallel training data 29
2.4 Overview of the monolingual training data 30
2.5 Overview of the Dev/test data 30
2.6 BLEU scores of the Baseline systems 31

3.1 English words aligned to the French word “faire” with the number of times they were aligned 39
3.2 Words related to “make” with their association scores 40
3.3 Word similarity evaluation sets 53
3.4 Word analogy evaluation sets 55
3.5 Evaluation of semantic associations obtained from bilingual alignments 56
3.6 Results of basic `word2vec` and GloVe word vectors 60
3.7 Comparison of different scoring approaches 62
3.8 Comparison of different context interpretations 64
3.9 Results of combining monolingual and word alignment word vectors . . 65

4.1 Different forms of our discounter ξ 76
4.2 Different ξ parameter estimates for WB-based ξ 79
4.3 List of unseen estimators used in this work 80
4.4 Examples of p_0 values for different estimators at different n-gram orders 83

4.5 Statistics of corpora and testsets used to evaluate the language model smoothing . . . 84
4.6 Perplexities of Devsets using models computed using different Kneser-Ney-based smoothing techniques on the EPPS-Corpus . . . 85
4.7 Perplexities of the models computed with different Kneser-Ney-based smoothing techniques on different corpora, evaluated on the testsets and trained using the best performing unseen estimator and discounter on the corresponding DevSet . . . 86
4.8 Perplexities of devsets using models computed by different Witten-Bell-based smoothing techniques on the EPPS-Corpus . . . 87
4.9 Perplexities of the models computed with different Witten-Bell-based smoothing techniques on different corpora, evaluated on the testsets and trained using the best performing unseen estimator and discounter on the corresponding Devset . . . 88

5.1 Classifier data sets . . . 104
5.2 Intrinsic evaluation for En $\rightarrow$ Fr . . . 105
5.3 Intrinsic evaluation for Fr $\rightarrow$ En . . . 105
5.4 Intrinsic evaluation for En $\rightarrow$ De . . . 106
5.5 Intrinsic evaluation for En $\rightarrow$ Ar . . . 107
5.6 Aggregated ranks of the different techniques . . . 107
5.7 Translation results for En $\rightarrow$ Fr . . . 108
5.8 Translation results for Fr $\rightarrow$ En . . . 108
5.9 Translation results for large En $\rightarrow$ Fr . . . 109
5.10 Translation results for En $\rightarrow$ Fr with a noisy lexicon . . . 110

6.1 Statistics of the datasets used in this evaluation . . . 121
6.2 Perplexity on Dev of the LMs selected using TED corpus . . . 122
6.3 Translation results on Test . . . 123
6.4 Translation results on Test for bilingual selection . . . 123
6.5 Results of stacking all the previous methods . . . 124

7.1 Examples of n-grams with low association scores from the EPPS corpus (for $n = 2$ and $n = 3$) . . . 129

7.2 Examples of sentences with lowest scores from various English corpora . 131
7.3 Examples of noisy phrases from the pruned phrase table 133
7.4 Examples of phrases having large distance between the cooccurrence and LLR rankings 134
7.5 Examples of bad sentence pairs from the EPPS corpus 135
7.6 Perplexities of weighted models using test sets from Chapter 4 137
7.7 Translation results for the WMT-2014 testset, using weighted and unweighted EPPS language models 138
7.8 Translation results for the WMT-2014 testset 139

8.1 Phrase scoring time span in WMT2011 150
8.2 Phrase scoring in IWSLT2011 150
8.3 Phrase scoring in WMT2012 150

B.1 Word pair semantic similarity examples from different datasets 163
B.2 Question examples from different analogy datasets 164
B.3 Syntactic similarity examples 164

C.1 Top 10 associated words with “*make*” using different association measures 165

D.1 Perplexities of devsets using models computed usin different Kneser-Ney-based smoothing techniques on the TED and GIGA corpora 168
D.2 Perplexities of devsets using models computed usin different Witten-Bell-based smoothing techniques on the TED and GIGA corpora 169

Glossary

***n*-gram**	A sequence of n words occurring in a corpus
SGD	Stochastic Gradient Descent
`word2vec`	A well-known neurla-network-based algorithm to generate word vector embeddings from a corpus
ASR	Automatic Speech Recognition
BiLM	Bilingual Language Model, model in a phrase-based machine translation system to increase the bilingual context
BLEU	Bilingual Evaluation Understudy, the most widely used evaluation metric for machine translation (Papineni et al., 2002)
CPU	Cetral Processing Unit
CS	Computer Science, test set containing computer science lectures used as one of the main translation tasks in this thesis
Dev	Development (Set), data that is used to train the log-linear model
EM	The Expectation-Maximization algorithm
EPPS	European Parliament Plenary Sessions, the largest available parallel corpus for most European languages
GIGA	A large parallel corpus for English – French, prepared by the LDC
Giza++	A popular tool for word alignment
GloVe	Global Vectors, a regression-based algorithm to generate word vector embeddings from a corpus
HMM	Hidden Markov Model
IBM*i*	IBM Model i, $i = 1, 2, \dots 5$
IO	Input Output
IR	Information Retrieval
IWSLT	International Workshop on Spoken Language Translation
KIT	Karlsruhe Institute of Technology
KN	Kneser-Ney Smoothing
LLR	Log-Likelihood-Ratio association measure
LM	Language Model
LOO	Leave-One-Out, an approach to estimate the number of unseen events by leaving one instance from each observed event
LSA	Latent Semantic Analysis
LSI	Latent Semantic Indexing
MDS	Multi-Dimensional Scaling
MERT	Minimum Error Rate Training, most commonly used optimization method for phrase-based machine translation
MKCLS	Word cluster algorithm
ML	Machine learning
MT	Machine Translation
NC	News Commentary, parallel corpus of several European languages
NER	Named Entity Recognition, task of automatically marking names in a text
NFS	Network File System

GLOSSARY

NLP	Natural Language Processing, field in the area of computer science and computer linguistics that concentrates on processing natural language data
OOV	Out-of-Vocabulary (word)
OPUS	the open parallel corpu
PL	Poisson-Lindley distribution
POS	Part-of-Speech, classes describing the function of the word in the sentence
PP	Phrase Pair, basic translation unit in a phrase-based machine translation system
SGD	Stochastic Gradient Descent
SLT	Spoken Language Translation
SMT	Statistical Machine Translation
SRILM	SRI Language Model, most commonly used language model toolkit
STXXL	Standard Template Library for Extra Large Data Sets
SVD	Singular Value Decomposition
SVM	Support Vector Machine, supervised model for machine learning
TED	Technology, Entertainment, Design, global conference, whoes talks are transcribed and translated into many languages.
TER	Translation Error Rate, evaluation metric for machine translation
TF-IDF	Term Frequency - Inverse Document Frequency, common measure in information retrieval for the importance of a word
TM	Translation Model
TSVD	Truncated Singular Value Decomposition
WB	Witten-Bell Smoothing
WMT	Workshop on Machine Translation

1

Introduction

Data is a key prerequisite for any machine learning task. It is the component to which different learning algorithms are applied in order to draw models. Based on these models, predictions will be made and decisions will be taken. From this it follows that the reliability of the whole process depends in a great part upon the “quantity” and the “quality” of the training data. While the former term is a clear concept and simply measures the number of training examples fed to the learning algorithm, the latter is rather vague and roughly corresponds to the degree of match between the training data and the real world. Conventionally, anything which would disturb this match is termed “noise”.

In real world situations, noise is inevitably present in any training data granted that our acquisition tools can by no means attain perfection. Of course, robust learning algorithms exist, but even for the most robust ones the effect of the noise will start to be noticeable as soon as it reaches a certain threshold. Another option to deal with the noise would be to involve the human cognition in preparing and selecting the appropriate training data. Although this choice is often adopted by commercial companies, it usually comes at a very high price. In the best circumstances, this noise can be hopefully tolerated by massive quantities of training data. Unfortunately, it is not always possible to find large amounts of adequate data nor is it easy to process them when they exist.

Compared to other natural language processing (NLP) applications, statistical machine translation (SMT) is probably one of the worst in terms of data availability. In fact, unlike most other applications, two types of data are needed (monolingual and

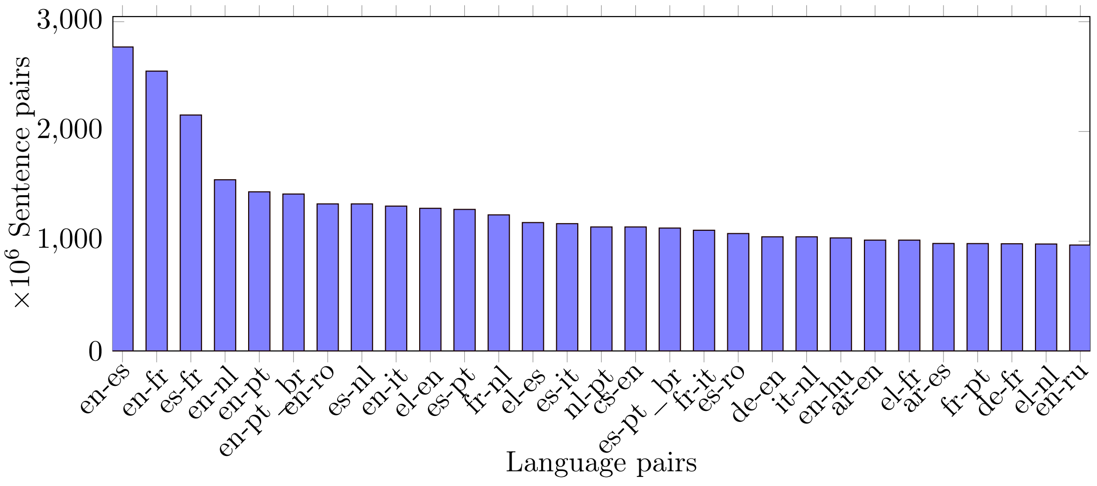


Figure 1.1: Parallel data available for different pairs in millions of pair of sentences

bilingual). While the monolingual data is relatively easy to find, the parallel data is much harder to obtain for most pairs of languages even for the most active ones. Take for instance OPUS,[1] one of the biggest data repositories for SMT. Let's assume that all the data in this repository is perfect. The total number of languages for which some data exists is 267; whereas the number of languages for which some parallel data exists is 381 pairs. This is almost negligible when compared to the number of possible pairs (a bit more than $71,000$). Figure 1.1 compares the amounts of data for the richest 30 pairs of languages. Amazingly, this figure suggests that Zipf's law does not characterize only the occurrences of words in a natural languages, but also the sizes of their corpora. Apart from a very few, the data availability remains a big issue for most of the language pairs.

Regarding noise in training data, there seems to have been a divergence between the commercial and research SMT communities. The commercial community is most concerned with satisfying their clients with a high quality output. For instance, wrongly translating a negation should be viewed as an enormous sin when translating a product manual to different languages. According to a leading company in this domain, this process involves lot of human intervention and would consequently be much more expensive.[2] This becomes obvious if one takes into consideration the manual effort spent

[1]http://opus.lingfil.uu.se

[2]http://www.asiaonline.net/EN/Resources/Articles/CleanDataSMT.aspx

in preparing high quality training data and in evaluating the system's output.

On the other hand, the research community was (and still is up to a certain degree) concerned with the coverage. This translates to finding data or techniques to deal with an unexplored language pair or even finding more data for the well-explored ones. This can be inferred from the International Evaluation Campaigns organized each year between SMT research groups all over the world (such as WMT[1] and IWSLT.[2]) In such evaluations, the automatic evaluation metrics used underestimate rare bad system errors in favor of translating more words. A simple example for this, but which might not exactly fit into the noise scenario, is the aforementioned negation problem. In addition, due to cost factors, most of the data used by this community is news data.

Thanks to these evaluations, most of the data they release is publicly available to the community. For most languages, this data is of important quantity and has been, in general, relatively preprocessed. Most importantly, a large part of this data was collected from the Web.

As for many NLP applications, it turns out that the textual contents on the Internet are a very appreciable data resource for SMT. This applies not only to the monolingual but also to the bilingual data for two main reasons. One reason is its reduced cost. The data collection operation can be, indeed, automated and carried out endlessly by machines. The other reason is its high availability. For instance, in today's world any language should have some associated textual content on the Internet. OPUS is, again, an example of a remarkably successful harvesting process which was held automatically.

In spite of its appealing features, the Internet data comes with an undesirable cost. In fact, it adds another source of errors to the errors already present in the data. The amount of anomalies introduced by the automatic tools used in the harvesting process is not negligible. In some way, this causes the ratio of noise to grow as the volume of the data increases. As a result, even the large data sets obtained from the Web are not able to bring the noise effects down to an imperceptible degree.

In SMT, the noisy data often turns out to be of high importance. One reason for this is the aforementioned data scarceness problem. Another interesting reason is when this data comes from the domain under consideration. Therefore, identifying the most useful subset of this data is very likely to improve its impact.

[1]http://www.statmt.org/wmt16/

[2]http://workshop2015.iwslt.org/

By eliminating the noise, especially the one introduced by the automatic tools, we are enhancing the match between the training data and the real world. More precisely, our benefits are two-fold. First, the estimation process will result in more accurate probability distributions. Ideally, this means that the probability mass is distributed only on the correct items. This, in turn, implies better predictions. In addition, one reduces the training data size, leading to a more efficient learning.

The work presented in this thesis was realized in order to reduce the noise effects in SMT. We achieve this via detecting and ruling out or down-weighing the noisy fractions of the training data. Hence we achieve better or at least comparable SMT quality with improved efficiency. We test and demonstrate the usefulness of our approaches on the type of data used in the international evaluations.

1.1 Noise in SMT data

In general, the SMT data is of textual nature. it comes in two sorts: monolingual and bilingual. As their names suggest, the two types differ in the number of languages in which the text was generated. In addition, in the bilingual case each sentence in one language should have a corresponding in the other. These texts are transcribed and made machine-readable by human operators. Moreover, in the bilingual case, the translation is also generated by humans. This implies that all mistakes which can be made in a human script can also be found in a training corpus.

The noise in SMT corpora can be categorized into two groups according to the type of the underlying corpus. Some noise involves pairs of sentences and therefore can be only encountered in parallel corpora. Other types of noise, by contrast, are concerned with individual tokens or sequences of tokens and consequently can be located in both monolingual and bilingual corpora. In the following, we give examples to illustrate both categories.

1.1.1 Monolingual noise

The noise in monolingual SMT data considers only tokens or sequences of tokens and can be found at different granularities:

1. At character level: this includes incorrect words because of bad character sequences. Typos, encoding conversion errors, orthographic irregularities, and spelling

Incorrect	Possible correction
Panel A: Character level noise	
accroissement dela population	accroissement de la population
great hotel.near to the center	great hotel. near to the center
a tesmimonial : " the Output is classic	a testimonial : " the Output is classic
Panel B: Word level noise	
make a the list of what they need to get there	make a list of what they need to get there
one cannot divide the the body from the mind and spirit .	one cannot divide the body from the mind and spirit .
please , send one copy to each of the to addresses below :	please , send one copy to each of the addresses below :

Table 1.1: Examples of noise in monolingual corpora

errors inconsistencies all fit in this category. In Table 1.1, three examples of these types were extracted from different corpora and are given in Panel A. The first column is the noisy segment and a possible correction is given in the second column. The problem with the first two examples is a missing space. It is likely that the first was due to a human mistake whereas the second was caused by an HTML conversion/extraction tool. The last example is a typo.

2. At word level: ungrammatical sequences of words due mainly to human mistakes or conversion side effects. Panel B of Table 1.1 shows three examples of this type. They seem to be the result of human mistakes.

3. At sentence level: at this level, any sentence from which we cannot learn useful patterns for the task at hand maybe deemed noise. In this sense, the noise should not necessarily correspond to grammatical or orthographic errors. on the contrary, the sentences might be perfectly correct from a linguistic perspective. For example, sentences from a medical text may in fact hurt a language model to be used in the automotive domain. Another good example here, is the usage of historical language corpora in tasks dealing with the same language in its modern form.

English	French
Panel A: Wrong pairs	
downloaded from on April 16 , 2002 .	téléchargé de l' adresse , le 2 mai 2002 .
country commercial guide U.S. commercial service 2004	étude de marché : les aliments halal AAC 2006
for example , the proportion of respondents who considered	les divergences d' opinion régionales étaient parfois marquées .
obtain independent assurance	vérifications indépendantes exécutées à intervalles réguliers .
sound recording development program	programme d' initiatives culturelles
Panel B: Independent translation	
Mr Chairman , you have the floor .	nous vous écoutons .
this is blatantly inconsistent .	il y a là un manque de cohérence flagrant .
there is none ; our double standards are just rank .	notre double langage est fétide .

Table 1.2: Examples of noise in bilingual corpora

1.1.2 Bilingual noise

Unlike the monolingual, the bilingual noise looks at pairs of sentences in different languages. These pairs can be wrong examples and therefore hurt the learned models. In addition, they can be correct translations but carry poor information for the learning algorithm. The first case, simply, addresses the wrong translation pairs, whereas the second stands for those examples generated independently and expressed in totally different ways. Table 1.2 gives examples of both cases. In Panel A, it is clear that these are incorrect pairs. It is very likely that the alignment system was confused because these segments are rather comparable. They address related topics, but there meaning is indeed different.

Although the pairs in Panel B correspond in meaning, we find very little or no lexical correspondence at the word level. As for idiomatic expressions, this will be only problematic if such pairs do not cooccur very often.

	Correct (%)	Partially Correct (%)	Incorrect (%)
Crawl	58	10	32
Giga	62	12	26

Table 1.3: Proportion of correct parallel sentences in the Crawl and Giga corpora

1.1.3 How noisy is the Internet data?

It is very difficult to give accurate estimates of the proportion of each noise type in the raw Internet data. However, the number of wrong parallel sentences may give an impression about the high degree of noisiness of this data. For instance, in the application scenario which was undertaken by Munteanu (2006), he started from 2.5 billion pairs and ended up accepting only 4.5 million pairs as candidates to his binary classifier. The binary classifier judges 2 millions from these candidates to be correct parallel pairs. This means that almost 99% of the initial data was noise.

The noisy data on which we apply our methods can be somehow considered identical to the final output of the aforementioned work. In spite of this, it contains large amounts of noise. We manually checked 50 random parallel pairs from our two main corpora (Common Crawl and Giga corpora) and tag each pair as being correct, partially correct, or incorrect. Where partial correctness means that the two sentences match in the meaning, but one of them has additional content to the matched meaning. The percentages of each category are given in Table 1.3. It is shocking to see that almost third of the parallel sentence pairs are just noise. In this particular example, we also note that the Common Crawl corpus is slightly noisier.

1.2 Scope of the work

The research presented in this thesis arose from the interaction with SMT evaluation data most of which was automatically collected from the Internet (i.e. WMT and IWSLT.) Noise and large sizes are general common features of such data. Therefore, we assume the following scope:

1. **Type of data:** We constrain our focus to the type of evaluation data. i.e large noisy data collected from the Web. Therefore, we don't consider data from the social media and Twitter.

2. **Type of noise:** We mainly consider the most frequent problems encountered in the chosen type of data.

    - Wrong sentence pairs in the bilingual data. i.e. sentences which are aligned in the original data to form a parallel pair, while in fact they are not translations of each other. This kind of noise was explained and exemplified in Section 1.1.2.
    - Sentences with poor information both in bilingual and monolingual data. Since the evaluation data usually comes from multitude of domains, some of it would be of low importance for the task under consideration. This corresponds to the sentence-level noise in Section 1.1.1.
    - Anomalous word sequences in monolingual data. Some words may appear in wrong contexts due to human mistakes or to the preprocessing our data has received. This kind of noise fits in word-level noise described in Section 1.1.1.

3. **Type of handling:** Because of the important data sizes, the noise will be treated by exclusion or penalization. We don't consider alternative ways such as regenerating a corrected token or sequence.

1.3 Main Contributions

The outcome of the work presented in this thesis is the development of six techniques which are able to improve both the efficiency and the performance of SMT system without adding considerable complexity to the learning process. Figure 1.2 gives a quick perception of the different contributions. The arrows in this graph point to the component making use of a given technique. For example, The "Parallelization" is used equally in phrase table scoring (i.e. "Translation Model") and in deriving synonymy relations from bilingual word alignments (i.e. "Semantic Associations"). The final targets in the SMT pipeline, where a given technique will be involved, are the "Translation Model" or the "Language Model", which appear in the right-most group (i.e. "Target SMT Component"). The group labeled "Main Contributions" encapsulates the approaches which are meant to directly tackle the problems in the data under consideration. Specifically, they deal with the two aforementioned characteristics of Web data: "noise" and "large

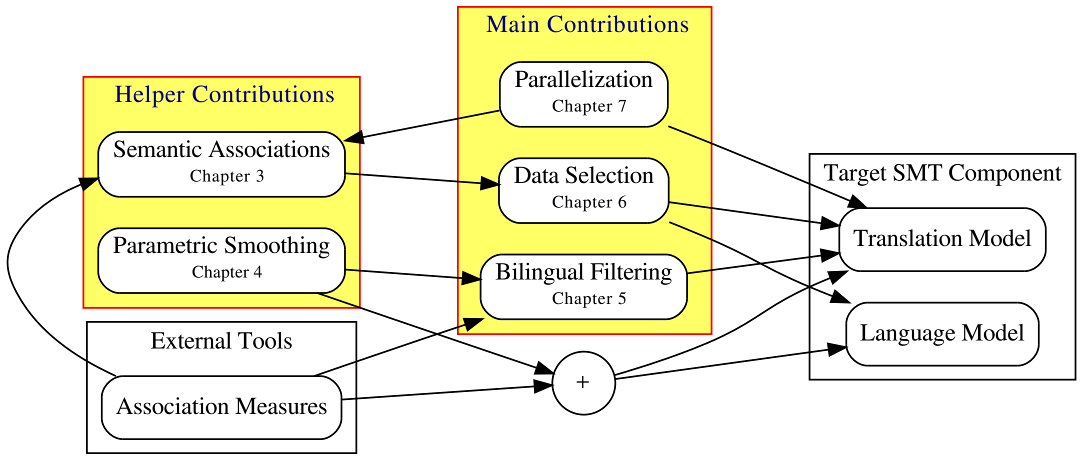


Figure 1.2: Schematic summary of contributions

Each rectangle inside the shaded boxes points to a contributed approach and the corresponding chapter where it is described.

size". The "Parallelization" addresses the latter and the two others are meant to reduce the noise.

The other contributions, stacked in the groups "Helper Contributions", are introduced as a response to a specific need in the main contributions. More precisely, our "Data Selection" approach relies on word synonyms, the reason for our work on "Semantic Associations" between words. Furthermore, using the "Association Measures" to penalize pairs which are likely to be noisy engenders the need for a smoothing that supports fractional counts, which leads to our proposed "Parametric Smoothing".

The "Association Measures" play a central role in our approach. They are, indeed, used in almost all cooccurrence-based models. In general, they have the ability to indicate whether a pair is a potential noise, more than the raw count has.

The proposed techniques can fit in two main axes:

- **Precision-biased models:** Where we attempt to reduce the noise effect on the computed models. We approach the problem at two different stages of the learning process:

  - *At preprocessing:* The techniques are applied to the data before the training. We rule out the noisy parts of the data.

1. Removing bilingual noise: We use a classifier-based technique to detect the wrong sentence pairs and then exclude them. This method relies heavily on a bilingual lexicon. We show that careful extraction of such lexicon yields higher precision classification results. Using an aligned clean corpus, the aligned pairs are attributed their probabilities based on statistical association measures. In addition, the probability of a word being unaligned gets its value through smoothing
2. Monolingual and bilingual data selection: The most useful part of the training data is selected based on the n-gram similarity to a given small data set. We exploit semantic word similarities to soften the n-gram matching. In other words, if a word appears in a context which has never been seen in the small data set, it can still get a high probability if one of its semantically equivalent words appear in this same context.
3. Semantic word associations: The semantic associations which are used in data selection (previous item) are automatically extracted from bilingual and monolingual corpora. We propose to use several weighting techniques to the bilingual and monolingual cooccurrences so that the resulting semantic associations are improved. In the case of bilingual cooccurrences, we proceed by pivoting through one language to obtain semantic associations in the other. In the monolingual case, we propose some modifications to the GloVe word vectors to get better semantic similarities from the resulting vectors.

– *At modeling:* While computing language models (resp. translation models,) we show how penalizing the unreliable sequences (resp. pairs) can help to reduce the effects of noise.

4. The unreliable elements are detected using the statistical association measures. A penalty is applied to an item based on the significance of association between its components.
5. We propose a simple smoothing technique which supports non integral counts. In some of our previously proposed techniques, we compute pseudo-counts which are not necessarily of integral nature. To deal with

this situation properly, we introduce a parametric discounting whose parameters are estimated by maximizing the data likelihood.

- **Efficient training:**

6. Hybrid parallel training: We tackle the expensive training of large corpora by exploiting parallelism in different ways. We present a framework which combines parallelism of processing units (CPU) and input/output (IO) devices. The CPU parallelism itself takes advantage of both shared-memory and distributed parallelism. This kind of massive parallelism is utilized in phrase scoring and phrase-table pruning in order to significantly reduce the training time.

1.4 Outlook

The next chapter presents the foundations of our techniques. In this chapter, we discuss the statistical association measures and how they are applied to word pairs or to n-gram of words. After that, we shed light on the SMT pipeline, which consists of the main steps necessary to build a phrase-based SMT system, starting from a raw text corpus. Special attention is paid to those steps which are more important for our work. We close this introductory presentation of SMT pipeline by giving some preliminary experiments establishing baseline systems. In this chapter, we also review our related work.

In Chapter 3, we look into extracting semantic associations between words from bilingual and monolingual corpora. Semantic associations play a central role in some of our proposed filtering techniques. More precisely, they are used in the process of data selection studied in Chapter 6 to help this task escape over-fitting. We devote a complete chapter to semantic associations since the study we conduct is extensive and too long to fit in the chapter where they are used. State-of-the-art techniques used to draw these associations from corpora are remarkably improved. Indeed, we demonstrate the effectiveness of using the statistical association measures in extracting semantic associations from monolingual and bilingual corpora. Additionally, we propose an approach to augment the associations obtained from a bilingual corpus with information from a monolingual one.

For similar reasons to those leading to a separate chapter for semantic associations, Chapter 4 comes to introduce some techniques which are used in the following chapters. Smoothing is an important operation in estimating language model and translation model probabilities. However, it only considers integral counts. To deal with situations where fractional pseudo-counts appear, we propose new smoothing techniques which support non integral counts. Through evaluations in the following chapters, these smoothing techniques are proved to work well with non-integral counts, and are still able to improve for the integral ones. We start by using them in training lexicons for noise removal from bilingual corpora in the next chapter.

Chapter 5 is devoted to detailing the techniques we use to filter out wrong sentence pairs from a parallel corpus. It is essentially a classifier-based approach. Therefore, we describe and motivate the different features the classifier uses to reliably decide whether a pair is correct. The most important component in this classifier is a bilingual lexicon. We describe how a high precision bilingual lexicon can be automatically extracted from bilingual corpora. We demonstrate the utility of our filter intrinsically and extrinsically. Furthermore, we show the possibility of using a classifier built for a given language pair in order to filter another pair. This demonstrates potential application of the filter to pairs with severe data limitations.

In Chapter 6, our approaches to monolingual and bilingual data selection are dissected. We explain our heuristics to chose a more appropriate representative of the out-of-domain data and vocabulary. We then explore a way of using semantic word associations in order to extend the small language models used for selection. These semantic extensions allow us to infer more realistic probabilities for the unseen sequences. We conclude this chapter by comparing the performance of the selection on monolingual and bilingual data.

Dealing with the noise at the modeling stage will be the theme of Chapter 7. While computing a model from noisy data, we show how penalizing the most unreliable items could enhance the resulting model. This is applied to both translation and language models at different levels of granularity. At a fine level we look into penalizing the most unreliable n-grams or phrase pairs. In this level, the reliability of an item is expressed in terms of statistical association measures. At coarser granularity, we penalize sentences or sentence pairs. For this, we use the cross-entropy of computed using a language model or a bilingual lexicon.

Since noisy data is mostly provided in large quantities, Chapter 8 addresses efficiency in computing models from such large training data. We present our solution for two expensive steps in the SMT training process: phrase scoring and phrase table pruning. A hybrid parallel model is designed in order to exploit parallelism in processing and in IO. We show that our main bottleneck resides in the IO rather than the CPU. This is demonstrated by the boost of performance obtained by taking advantage from a massively distributed architecture which combines CPU and IO distribution.

Finally, we conclude this thesis with a summary of the lessons learned from our research and we give our thoughts for further possible investigation directions. We also include appendices with more results for our proposed smoothing techniques, and more examples for the outputs of different approaches.

2

Foundations

This chapter covers the elementary building blocks on which we rely in the next chapters. We present two elements: Statistical Association Measures (SAM) and Statistical Machine Translation (SMT). Then, we quickly survey the related literature. The SAM are indicators about the strength of relation between cooccurring entities. In many situations we consider them as indicators of noise, when the relation is weak. We introduce the measures we have used in our work. The SMT is the field where our study fits. We show the main components which are related to our work. We also present the configuration of our baseline system which will be improved throughout the rest of the thesis. In the literature review, We survey the most important related work to ours. More detailed discussions of this related material will follow in the corresponding chapters.

2.1 Statistical Association Measures

Statistical Association Measures (SAM) are scores associated with pairs of cooccurring entities. For two given entities which appear together in the data, a SAM score quantifies the strength of the relation connecting these entities. Typically, a small score indicates a weak connection while a larger score means a strong connection.

Many of the models we use in Statistical Machine Translation (SMT) are based on pair cooccurrence counts. The counting of word alignments in bilingual corpora and word cooccurrences in monolingual corpora are instances of this process. n-gram occurrences can be also cast in this process if we take the whole context as a unit and the last word as the other unit. In our work, we use the SAM's as indicators of the reliability of a pair. A pair with a low association score is seen as potential noise. The counts of such pairs will be, therefore, penalized.

Some association measures are statistically-founded; while others define the association strength heuristically. Many of the association measures are statistically-founded. The statistically-founded measures originate from statistical significance testing. Therein, the association is interpreted as the statistical dependence between the two entities. The corresponding score is a random variable with a known distribution (asymptotically at least). The null hypothesis (i.e. the independence) is rejected if the probability of the computed score (i.e. the p-value) falls below a certain threshold (i.e. the significance level). However, omitting these details and simply using the score itself is a common practice in NLP (Moore (2004), Melamed (2000), Munteanu and Marcu (2006)). The heuristic association measures, on the other hand, directly combine the cooccurrence parameters to compute a reasonable association score.

Using the association measures in collocation extraction is probably the most related to our case. This relies on the intuition that a collocating context should be more important for semantic relatedness. In this task, candidate word pairs are ranked based on association scores, and the top ones are returned as the resulting list of collocations. The scores are computed from cooccurrence frequency of the words in a predetermined window in a given corpus. In his PhD dissertation, Evert (2004b) gives an excellent and complete presentation of collocation extraction using SAM's.

The added value of the SAM's as compared to the raw frequency is that they provide a statistical interpretation of this latter. With their help, it is possible to distinguish between meaningful cooccurrences and those due to chance (i.e. due to corpus choice and can not be generalized to other corpora). The main reason behind this robustness is that they exploit more than one cell in a contingency table.

Given a word pair (w_1, w_2), a contingency table gathers all frequency values for this pair in a 2×2 matrix, as shown in Figure 2.1. "cooc" is a function returning the number of times two words cooccur in a corpus and "occ" returns the total number of times a word cooccurs with any other word. In addition to the number of times the considered pair was seen together, the contingency table also encloses the number of times the words in this pair were seen apart and the number of times none of them was seen. The values in a contingency table are sufficient to compute the expected number

$\mathrm{cooc}\,(w_1, w_2) = C$	$\sum_{w \neq w_2} \mathrm{cooc}\,(w_1, w)$	$\mathrm{occ}\,(w_1) = O_1$
$\sum_{w \neq w_1} \mathrm{cooc}\,(w, w_2)$	$\sum_{\substack{w \neq w_1 \\ w\prime \neq w_2}} \mathrm{cooc}\,(w, w\prime)$	$= N - \mathrm{occ}\,(w_1)$
$\mathrm{occ}\,(w_2) = O_2$	$= N - \mathrm{occ}\,(w_2)$	$= N$

Figure 2.1: Contingency table for a word pair (w_1, w_2)

of cooccurrences under the null hypothesis (i.e. assuming their independence). For instance, the expected cooccurrence count would be the multiplication of the words' occurrences divided by the total number of pairs (i.e. $\frac{O_1 \times O_2}{N}$). For the statistically founded measures, the association score serves to infer whether the difference between the expected and the observed frequencies is significant. If so the null hypothesis is rejected and the words under consideration are, in fact, associated. In the following, we will present some measures we will use throughout this work.

Log-Likelihood-Ratio Measure (LLR) LLR is one of the statistically-founded measures. It is also referred to as G^2-test in the literature. Using LLR as an association measure was proposed by Dunning (1993) to analyze cooccurrences in textual data. It is particularly good with the rare events, which are a typical characteristic of the natural languages. The power of this measure originates from its weak dependence on the normality assumptions, unlike other standard Chi-squared tests. For rare events, normality is an unrealistic assumption. In practice, it has been shown that LLR is suitable for several NLP tasks. It was used in bilingual word alignment by Melamed (2000) and Moore (2004). In a closely-related context, it was used by Munteanu and Marcu (2006) in order to derive higher precision lexicons from aligned words. It was also used, and shown to outperform many other measures, for collocation extraction by Evert (2004b) and Pecina (2009).

The LLR test statistic corresponds to the ratio of the likelihood of the data under the null hypothesis and the likelihood under the alternative hypothesis. Using the notations in Figure 2.1, the LLR is given by (Dunning (1993)):

$$\mathrm{LLR}\left(w_1, w_2\right) = -2\log\left(\frac{\mathrm{L}\left(p, C, O_2\right)\mathrm{L}\left(p, O_1 - C, N - O_2\right)}{\mathrm{L}\left(p_1, C, O_2\right)\mathrm{L}\left(p_2, O_1 - C, N - O_2\right)}\right) \tag{2.1}$$

Where

$$\mathrm{L}\left(q, k, n\right) = q^k\left(1 - q\right)^{n-k}$$

and

$$\begin{aligned} p &= \frac{O_1}{N} \\ p_1 &= \frac{C}{O_2} \\ p_2 &= \frac{O_1 - C}{N - O_2} \end{aligned}$$

LLR is a two-sided test, meaning at low values of LLR the null hypothesis cannot be rejected and therefore weak association between the two words is inferred. On the other hand, when LLR is large, the two words should be associated, but this association can be either positive or negative. A positive association implies that the words are expected to cooccur more often and their cooccurrence is meaningful, whereas a negative association means that the two words should not cooccur. The association is positive if $\Pr\left(w_1, w_2\right) > \Pr\left(w_1\right)\Pr\left(w_2\right)$ which is equivalent to $NC > O_1O_2$.

We use the LLR measure to penalize co-occurrence counts which are used to infer semantic word associations from bilingual and monolingual data (cf. Chapter 3). We use them also to penalize n-gram counts (cf. Chapter 7).

Jaccard Measure (Jaccard) The Jaccard measure uses the probability of seeing both words given that one of them is seen as association strength indicator (i.e. $\Pr\left(w_1 \wedge w_2 \mid w_1 \vee w_2\right)$). This measure is closely related to the Dice measure. Indeed, a simple monotonic relationship exists which converts one into the other.[1] These measures are extensively used in information retrieval as pointed out by Evert (2004b). The two measures were also commonly used in analyzing the cooccurrence significance. In

[1]If J denotes a Jaccard score and D the Dice, then $J = \frac{D}{2-D}$

his PhD dissertation Dunning (1998) cited both measures in a general frame-work for analyzing cooccurrence data. The Dice measure was chosen to extract collocations by Smadja et al. (1996).

The Jaccard measure is computed as follows (this is the maximum likelihood estimate of the aforementioned probability):

$$\text{Jaccard}\,(w_1, w_2) = \frac{C}{O_1 + O_2 - C} \tag{2.2}$$

We use the Jaccard measure to compute a higher precision lexicon from word alignments (cf. Chapter 5).

Geometric Mean Measure (GMean) This measure is defined in a way similar to the Jaccard measure. It corresponds to the geometric mean of the probabilities of seeing the pair knowing that one word of the pair is present (i.e. $\Pr(w_1, w_2 \mid w_1)$ and $\Pr(w_1, w_2 \mid w_2)$). Even though this measure has not been very popular in collocation extraction, its very closely-related analogue Pointwise Mutual Information (PMI) measure was extensively used.[1]

The GMean measure is computed as follows (Evert (2004b)):

$$\text{GMean}\,(w_1, w_2) = \frac{C}{\sqrt{O_1 \times O_2}} \tag{2.3}$$

Like the Jaccard measure, the GMean measure is used in building bilingual lexicon from word alignments.

2.2 Statistical Machine Translation

Statistical Machine Translation (SMT) was first inspired by Brown et al. (1990) from the successful application of statistical and machine learning techniques in speech recognition and has been an active area of research since then. Every SMT system consists of a set of parameters, which are first determined in an *estimation* phase and are later used by the decoder in the *prediction* phase. The system parameters are estimated on the basis of a bilingual corpus which is a set of texts in the source language along with their equivalent translations in the target language aligned at the sentence level together with

[1]For an extensive study of the Mutual information measures, we refer to Bouma (2009) and the references therein.

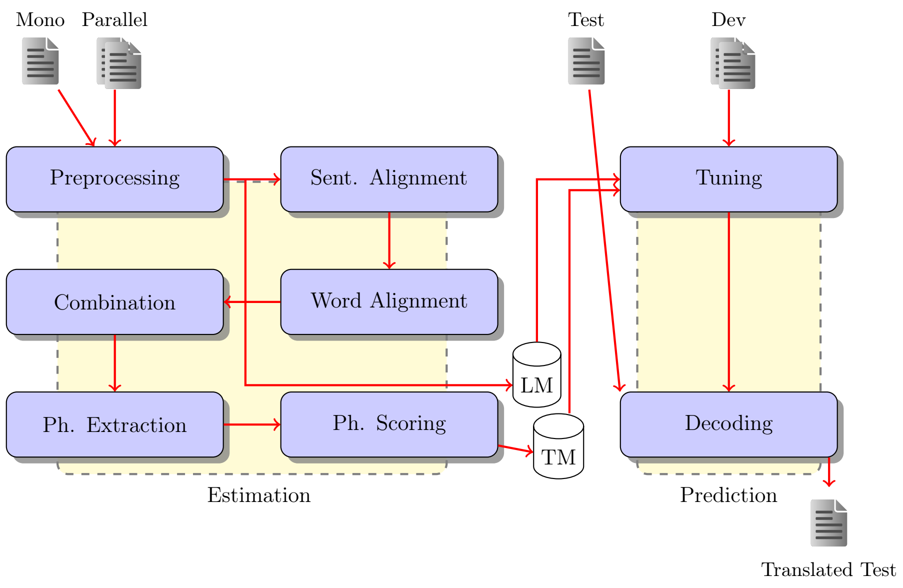


Figure 2.2: SMT Pipeline

another corpus in the target language. When the system is given a new sentence in the source language, it generates several candidate translations, evaluates each candidate's probability, and outputs the most probable sentence. It is almost always beneficial to preprocess the corpora earlier than any other operations. Obviously any preprocessing will have to be reverted by postprocessing the system's output. In the following, we review the main components which constitute the SMT estimation and prediction steps. We need to mention, however, that our presentation is not exhaustive. Rather, we review the components in the context of our work. More detailed description of the different SMT techniques can be found in Koehn (2010). A simplified diagram showing the ordered main operations of the pipeline is presented in Figure 2.2.

2.2.1 Estimation

For the different models to be computed, the given corpora are used. The Translation Model is computed from the parallel corpora, whereas the Language Model is computed from the monolingual corpora. Interestingly, nothing prevents from adding the target part of the parallel corpora to the monolingual data to obtain more data for the language model computation.

The parallel corpora consist of identical linguistic content in two or more languages, while for the monolingual ones this content should be only in one language. Due to this constraint imposed on them, collecting parallel corpora is a much harder task than it is for their monolingual counterparts. However, thanks to the efforts put by the community in collecting them, great deal of parallel and monolingual corpora is available today, most of the time, for no cost. Table 2.1 shows some of the most important resources available for the SMT community.

Name	Languages	Nature	Availability
LDC	Various	laws, news	copyrighted
Hansard	French-English	legislative	free
OPUS	271	heterogeneous	free
Europarl	European languages	parliament proceedings	free
UN	6	political, socio-economic, health, ...	free

Table 2.1: Some important sources of the SMT corpora

2.2.1.1 Preprocessing

In most cases, the corpora need several preprocessing operations before becoming ready for use. some of these operations might be language-dependent implying using different tools for different languages. Examples of preprocessing, we perform, include, but not limited to,: normalization of special symbols, tokenizing strings into words, smart-casing the first word in each sentence, removing very long sentences since they can be a burden for the learning algorithms.

Other morphological operations may have a positive impact on the system performance. For instance, the tokenization of words into stem and affixes would yield less number of unknown words, especially for agglutinative languages such as Arabic or Turkish. Fortunately, a large number of preprocessing resources are made currently freely available.[1]

2.2.1.2 Sentence Alignment

The alignment between two sets of sentences is a subset of their Cartesian product. The number of sentences in the two sets maybe different and the order is not necessarily

[1]For example. Moses: `https://github.com/moses-smt/mosesdecoder`

preserved. The word alignment will be carried between words appearing in the aligned sentences. The translation model, in turn, is extracted on using the resulting word alignments. Consequently, the behavior of the whole SMT system will strongly depend on the sentence alignment. Even though most available parallel corpora are offered sentence-aligned, for the sake of completeness, we briefly present the basic sentence alignment algorithm, in the following.

Sentence alignment algorithms can be length-based, lexical-based. The length-based algorithm relies on the lengths of the sentences, while the lexical-based algorithm uses a bilingual lexicon. However, these two approaches are not exclusive and therefore can be combined. For instance, Varga et al. (2007) demonstrates that a hybrid algorithm to align between English and Hungarian can perform much better than both.

The length based approaches are probably the oldest, but still one of the most effective. The most popular algorithm in this category is due to Gale and Church (1991). Their algorithm is based on a very simple assumption: sentences which are mutual translations of each other should correlate in length. The alignment of two documents is performed in two main steps. First, paragraphs are aligned and then the alignment at the sentence-level is carried out inside every pair of aligned paragraphs. However, aligning paragraphs is a trivial task since paragraph boundaries are usually clearly marked.

In order to align sentences in a source paragraph and its target translation, a dynamic algorithm is used. The algorithm considers the distance between every two sentences and iteratively finds an alignment which minimizes the overall distance between the two paragraphs. The distance between two sentences is basically defined in function of their lengths and the type of match. Six types of matches between sentences are taken into consideration: $1-1$ (substitution), $1-0$ (deletion), $0-1$ (insertion), $1-2$ (expansion), $2-1$ (contraction), $2-2$ (expansion and merging). If we assume that the lengths (in number of characters) of the two sentences which are to be matched are respectively l_1 and l_2. Then, the distance d is estimated by the expression:

$$d = -\log \Pr(\delta \mid m) \Pr(m) \tag{2.4}$$

where m denotes one of the types of matches mentioned above; and where $\delta = \frac{l_2 - l_1 c}{\sqrt{l_1 s^2}}$ is the standardization of a normal random variable expressing the number of characters of the target sentence (l_2) generated by the number of characters in the source sentence (l_1)

and whose mean c and standard deviation s^2 can be estimated from a previously aligned bilingual corpus. Equally the probability of a match $\Pr(m)$ can be also estimated from this previously aligned bilingual corpus. The algorithm computes the first term of equation (2.4), using the assumption: $\Pr(\delta \mid m) = 2(1 - \Pr(\delta))$.

2.2.1.3 Word Alignment

Matching words which are mutual translations from two sentences has a direct impact on the translation model building. Statistical learning-based techniques introduce the alignment as a hidden variable into the posterior probability of a target sentence given a source sentence. It is possible that some of the source words will not be aligned to any of the target words, then it is said that they are aligned to the empty word. In other words, the probability of translating a source sentence into a target sentence is obtained by considering all the possible alignments between their words. However, the best alignment, among all those, is the one which maximizes this probability. In all cases, an Expectation Maximization (EM) algorithm is invoked to train the model.

The original SMT paper proposes five different alignment models which are commonly referred to as IBM models. The first model is the simplest, more parameters are taken into consideration with model 2 and so on. In practice, we mostly rely on the IBM models up to 4, due to the high cost of the fifth with no much gain. We briefly introduce each of these models in the following.

IBM model 1 gives the same chances to every possible alignment. The two sentences are considered as two bags of words where the order of those words in every sentence is not important. The probability of an alignment is the product of the probabilities of the individual words.

The difference between IBM models 1 and 2 is not really significant. Whereas in model 1 no attention was paid to word order, in model 2 the probability of choosing a source language word to be connected to a target word depends on the position of this word in the target sentence, in addition to the lengths of the two sentences. This newly introduced dependence will have an effect on counting connected words from the parallel corpora during the EM training (how likely does the given target word connect to a source word at a given position?).

The IBM model 3 introduces the notion of fertility of source language words. The fertility of a word in the source language is the number of words from the target language aligned to that source word assuming that every word in the target sentence is generated by only one single word from the source sentence (one-to-many). The idea behind the fertility models (model 3 and forth) is to first choose a fertility for each source word, i.e. a number of positions in the target sentence (fertility probability). Then, the words are generated (translation probability). Finally, the words are reordered in order to produce the final target sentence (distortion probability). Some attention should be paid to the words aligned to the imaginary empty word in the beginning of the source sentence, since they don't have to be included in the reordering.

Model 4 is similar to model 3 but with a different distortion assumption. The distortion depends on the classes of words and on their positions. The distortion expression is divided into two parts: distortion of the first word to be placed and distortion of the remaining words. The distortion of the head is the probability of making a relative movement in the target sentence according to the positions of the words aligned to the previous word from the source sentence , given the classes of this head and the previous source word. When the previous word is aligned to multiple target words the average position is taken. All the remaining words after the head must be placed in its right-side. Their distortions measures the distance in words between the head and the word of interest in the target sentence, given the class of the head. Some class of words produce contiguous words whereas others produce words which can be separated.

Och and Ney (2003a) implements the IBM models in a very popular toolkit called Giza++. In our work, we use the parallelized version of Giza++ due to Gao and Vogel (2008).

2.2.1.4 Symmetrization

The IBM models have a fundamental problem: They allow for only one-to-many alignments. In order to support many-to-many alignment, Och and Ney (2003b) proposes symmetrization heuristics which use two one-to-many alignments in the two possible directions. Two possible combinations are straightforward: the union and the intersection. In the former, any alignment point appearing in any direction is included in the final alignment. In the latter, any alignment point not appearing in both alignment will not be added to the final alignment.

A more interesting heuristic lies somewhere between the union and the intersection. This heuristic starts by taking all points in the intersection. Afterwards, neighboring points of an alignment point are added if they appear in the union. This operation is referred to as "growing". Next, the alignment points in the union for words which remained unaligned are also included in the final alignment. This last heuristic is the one commonly used before extracting the phrases, as it results in better performance.

2.2.1.5 Phrase Extraction

In this work, we adopt the phrase-based approach to SMT. The idea is to translate small sequences of words rather than single words. Considering sequences of words is a more natural way of translation, as it is very frequent that a word in one language translates into multiple words in another language. For instance, the french word "merci" translates to the English two-word-phrase "thank you". This task consists of extracting equivalent sequences in the two languages from the parallel corpus.

The extraction of phrases strongly relies on word alignments. In this process,a legal phrase pair is any sequence from the source and target if they are consistent with the underlying alignment. Consistency, simply, means that the sequences should have at least a source word from the source phrase aligned to a target word from the target phrase. In addition, no source word is aligned to any target word other than those in the target phrase, and conversely no target word in the target phrase can be aligned to a source word other than those in the source phrase. Table 2.2 shows the phrase extraction applied to the aligned sentence pair in Figure 2.3. Note that the unaligned words will never break the consistency of a phrase pair. Therefore, the extraction will be greedy with the unaligned words, in that they will be added to the surrounding aligned words during the extraction.

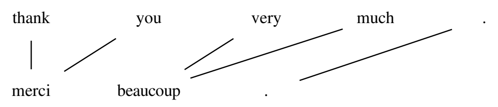


Figure 2.3: Example of aligned sentence pair

2.2.1.6 Scoring

After all phrases have been extracted, they get scored. It is very common to attribute at least four scores to each phrase. Two of these scores will be the conditional probabilities

French	English
merci	thank you
merci beaucoup	thank you very much
merci beaucoup .	thank you very much .
beaucoup	very much
beaucoup .	very much .
.	.

Table 2.2: Phrases resulting from alignments in Figure 2.3

of the source phrase given the target, and vice versa. The other two are lexical weights in both directions.

The conditional probabilities used in our work are smoothed as proposed by Foster et al. (2006a). The lexical weights use the original averaging method proposed by Koehn et al. (2003). More details about these scores and how they are computed will be given in Chapter 8, as we describe our parallelized scoring.

It is noteworthy that the number of resulting phrases is typically huge. Therefore the scoring should be followed by a pruning. For each source phrase only the n-best corresponding target phrases are kept (we use the value $n = 10$). The four scores attributed to a phrase pair are aggregated using a weighted log linear combination whose weights were set empirically. Next, the pairs corresponding to a given source phrase are ranked and only the top n are kept.

2.2.1.7 Language Models

n-gram language models are needed to make the decoder's output look as fluent as possible. In other words, a language model will measure how likely a sequence of words would be from the language they were trained for. The language model helps to pick the better fitting words given their contexts and to select the right word order. The better the word choice is, the higher its probability is, and the better the words are ordered, the higher the probability of the sequence is.

Unless stated otherwise, we use 4-gram language models trained with the modified Kneser-Ney smoothing. To train these models, we use the SRILM toolkit (Stolcke, 2002). However, we will introduce a new smoothing technique in Chapter 4, and for that we implement our own training tools.

2.2.1.8 Reordering Models

One of the major difficulties that machine translation faces is the different ways languages order words in a sentence. It is of course unfeasible to generate all possible permutations and score them with the language model. Therefore, we usually use a model to perform plausible word orderings in the target language.

In our work we rely on the method proposed by Rottmann and Vogel (2007), which performs reorderings on the source side so that the translation is monotone. To perform this reordering, rules are learned from the aligned parallel corpus together with the POS tags of the source side. Now, given a source sentence, the relevant rules are applied to generate many possible reorderings. These reorderings are encoded into a lattice which will be fed to the decoder.

2.2.2 Prediction

All the previously mentioned models (also referred to as features) are combined in a log-linear way. Each of these features has an associated weight. The decoder generates a large number of hypotheses from the combination of target phrases generated by different segmentations of the source sentence. Each of these hypotheses will get a score equal to the summation of the logarithm of each feature multiplied by its weight. The best hypothesis is then output.

We use four features from the Phrase table corresponding to probabilities and lexical weights in both directions. Another feature corresponds to the language model probability. The number of generated target words and the number of phrases used to generate the target sentence are also used as features. In addition, we also use two more features for the reordering, one from the lattice and one from the source positions corresponding to a target phrase. Features which correspond to target phrases are summed up to give a single value for the whole target sentence.

2.2.2.1 Tuning the Log-Linear Weights

In order to determine the weights corresponding to different features, we use the Minimum Error Rate Training (MERT) proposed by Venugopal et al. (2005). This optimization is held using a Development set (DevSet), which is an extremely tiny parallel corpus, which we believe should be very similar to the Test sets on which the system will be working. The process starts from some initial weights, then the DevSet is repeatedly translated to generate an n-best list, using the current weights. At the end of each iteration, the weights are updated so that the evaluation of the DevSet is improved. We usually iterate this process 20 times and pick the weights which gave the best evaluation.

2.2.2.2 Decoding

The decoder takes an input source sentence and uses the different models to perform a search in the space of all possible translations, and then returns the best translation. The decoder we use in our work is described in Vogel (2003). It is able to take reordering lattices mentioned above or normal text as input.

2.2.2.3 Evaluation

To know how well our system is doing in translating a Test set, we compare its output to the provided reference(s). We, mainly, evaluate our translations and report the performances in the Bilingual Evaluation Understudy (BLEU) metric developed by Papineni et al. (2002). BLEU is a very popular evaluation metric due to its simplicity and fair correlation to the human evaluation. The BLEU score has values between 0 and 1. The closer the BLEU score is to 1, the better the translation will be. It should be noted that our reported BLEU scores are given for the true-cased output. The value is almost always slightly higher if one ignores the case while evaluating.

2.2.3 Baseline

Our main baseline system is built using the WMT evaluation data. Our parallel corpora are subsets of the EPPS, Crawl, and Giga corpora. The EPPS gets its data from the European Parliamentary Proceedings Sessions in 21 European languages (Koehn, 2005). It covers the proceedings from 1996 to 2011. It was sentence-aligned using the Gale & Church algorithm described above. We use the French and the English versions of this corpus, and we denote it by EPPS. We consider this corpus as example of clean data.

The Crawl, also referred to as Common Crawl Corpus, data is typical example of Web data. It is a automatically collected from the Web by a non profit organization.[1] A

[1] `http://commoncrawl.org/`

small project was started in Machine Translation Marathon 2012 to extract parallel data from this huge corpus.[2] The outcome of this project was released in next year's WMT evaluation. It has received many additions and enhancements since then. Starting from the raw Common Crawl corpus, URL web pages were first aligned using the technique proposed by Resnik and Smith (2003). Then, again, the extracted pages are sentence-aligned using Gale & Church. We denote the subcorpus extracted from this corpus for our study by Crawl. We found this corpus to be very noisy.

The Giga corpus is by far one of the largest parallel corpora. It was first released in WMT 2009 by Callison-Burch et al. (2009), and is freely available since then. Therein, it was referred to as 10^9 word parallel corpus. The corpus was automatically collected from Canadian, European, and international sources. Simple heuristics were applied to match the downloaded URL pages to gather bilingual content. The sentence-alignment was performed with a hybrid algorithm using IBM model 1 lexicon and the length information as proposed by Moore (2002). In addition, some simple cleaning heuristics were applied, such as removing pairs where the source and the target are identical. The subcorpus in this case is named PGiga, in order to differentiate from the monolingual Giga hereafter. We found this corpus to be also noisy.

Sometimes, when indomain data is needed we use the TED corpus. TED corpus consists of the transcription of the TED talks together with their translations in several languages.[1] A non-profit organization named TED organizes talks given by distinguished people in their field of experience. The talks are usually short and are given in wide variety of topics. However, their style is similar, and each one of them has a specific topic. Therefore, we considered this corpus as topic-specific corpus. We will use the whole TED corpus and it will be referred to as TED. Statistics about the parallel subcorpora we use throughout this thesis are presented in Table 2.3.

For the monolingual data we extracted a subset of Gigaword French and English corpora. They both consist of an archive of newswire in French and English respectively. The are proprietary corpora and are distributed by the Linguistic Data Consortium (LDC) at the University of Pennsylvania. We randomly selected around one millions

[2]The project page can be accessed at: `http://www.statmt.org/mtm12/index.php%3Fn=Projects.ParallelCorpusExtractionFromCommonCrawl`

[1]`https://www.ted.com/`

Corpus	Sent. Pairs(K)	En. Words(M)	Fr. Words(M)	En. Voc.(K)	Fr. Voc.(K)
EPPS	494.0	13.6	15.0	59.0	76.9
Crawl	292.2	7.2	7.9	205.8	231.8
PGiga	385.3	10.5	12.4	193.8	216.0
TED	219.4	4.5	4.8	57.3	75.7

Table 2.3: Overview of the parallel training data

sentences from the large corpora, then deduplicated them. The statistics of the final corpora are gathered in Table 2.4. We will designate this corpus by Giga.

Corpus	Sentences(K)	Words(M)	Voc.(K)
English Giga	977.1	23.5	306.3
French Giga	984.0	27.0	331.4

Table 2.4: Overview of the monolingual training data

We mainly, use the development data distributed in WMT and IWSLT as Dev and Test sets. More precisely, Our main Test set is News2014, and our Dev set is News2012. In addition, sometimes when testing on indomain data is necessary we use IWSLT2013 as test set, and IWSLT2010 as Dev set. We also use News2013 and IWSLT2014 as additional Dev/Test whenever needed. Details should be provided in the evaluation sections of the respective chapters. Statistics of these sets are shown in Table 2.5.

Set	Sentences	Source		Target	
		Words	Voc.	Words	Voc.
News2012	3 003	73 976	9 872	77 850	12 667
News2013	3 000	65 678	9 195	69 828	11 766
News2014	3 003	72 595	10 008	77 055	12 495
IWSLT2010	1 686	27 378	5 621	28 552	6 830
IWSLT2013	1 026	21 898	3 771	23 666	4 499
IWSLT2014	1 305	24 951	3 771	27 931	4 499

Table 2.5: Overview of the Dev/test data

Direction	Conf.	Clean	Noisy
Fr→En	Par. LM only	23.19	25.59
	Giga LM only	27.02	27.71
	Both	27.15	27.90
En→Fr	Par. LM only	22.29	24.93
	Giga LM only	25.95	27.18
	Both	25.99	27.26

Table 2.6: BLEU scores of the Baseline systems

In the Clean system the parallel corpus consists of EPPS only; in the Noisy, the parallel data contains both Giga and Crawl. The first configuration in each direction uses a single language model trained on the parallel data, the second on Giga corpus; the third contains both.

The results of the baseline system are shown in Table 2.6 Two baseline systems were trained for each direction. The one called "clean" uses only EPPS data, whereas the one called "noisy" uses both Crawl and PGiga corpora. The systems use the reordering rules trained on the corresponding system alignments and POS tags. For each system, there are three different configurations of language models. The first corresponds to using only the target part of the parallel corpus for language model training. The second corresponds to using only the Giga data. The last uses the two language models as separate features. The scores recorded in this table are for the News2014, while the tuning was performed for News2012.

From Table 2.6, it is easy to acknowledge the importance of the noisy data. Even though it contains non-negligible amounts of noise, it still significantly outperforms the clean data. It is true that the noisy system is trained on a slightly larger corpus. However, we think the difference should be mainly due to the large gap between their coverages. This can be confirmed by observing the vocabulary sizes in Table 2.3. For example, even though the Crawl corpus contains only roughly half the number of sentences and words in EPPS, the Crawl vocabulary is more than three times the EPPS vocabulary. It can also be noted how important the language model is from these results. The large LM has a big advantage over the smaller one. When used alone, the large LM outperforms the smaller by a very large margin (compare first and second rows for each direction). By contrast, when we use both model the improvement over the large model alone is not as strong.

2.3 Related Work

Our main goal is to deal with the noise encountered in the training data for machine translation. However, the noise has different interpretations in different contexts. In parallel data context, the noise corresponds to mismatched sentence pairs. These are sentences which are taken to be correct translations, where they are not. On the other hand, when good representative exists, the noise can be interpreted as any sentences dissimilar to this representative. Consequently, we will review the most important literature to our work in these two contexts separately.

2.3.1 Filtering Mismatched Sentence Pairs

Identifying parallel sentences in a sentence-aligned corpus is mostly studied in the framework of comparable corpora. The research line on this topic started in the early 2000 and still continues, because it will be always hard to satisfy the large number of possible language combinations. Almost all the techniques in this category start by aligning source documents to one or more target documents. All sentences in two aligned documents are paired, afterwards. Here, one might see the utility of the document alignment operation which is intended to reduce the space of possible pairs. Afterwards, all the resulting pairs are examined for parallelism. The most dominant approach to identifying sentence pair parallelism is to use a binary classifier.

One of the early works to adopt this direction was Munteanu and Marcu (2005). Their binary classifier is based on a bilingual lexicon and a small parallel corpus. The classifier learns the positive examples for this Small corpus. The negative examples, on the other hand, are synthesized from the parallel corpus by randomly shuffling one side of the corpus. Each sentence pair is first aligned using the lexicon and a set of more than 30 features is computed accordingly. The classifier uses these features to decide whether a sentence pair is parallel.

Hunsicker et al. (2012) follow approximately the same approach. They differ from the previous study in using additional heuristics to trim the search space such as imposing more restrictive thresholds on the difference of lengths or the dictionary score. They also use a slightly different feature set, in that they differentiate content and function words and stem the content ones.

This line of research was recently adopted by many projects trying to extract parallel content from Wikipedia. Wikipedia is attractive because of the large number of languages it involves and its diverse content. Additionally, it is structured in a manner that makes the document alignment easier. A follow-up of the previous research conducted the researchers to extract a parallel corpus from Wikipedia of two million high quality pairs (Stefanescu and Ion, 2013).

2.3.2 Filtering as Data Selection

In the literature, filtering monolingual data is referred to as "data selection". It has received lot of attention in the few past years as researchers start to develop translation systems for new hardware-constrained devices such as smart phones. The process consists of extracting the most similar sentences to a representative of the clean data from a noisy corpus. The representative is usually called indomain data, whereas the noisy corpus is known as the out-of-domain or the general-domain data.

The cross-entropy approach to selection has gained popularity because of its simplicity and efficiency. It was first introduced by Moore and Lewis (2010). The secret ingredient in its success is that it enhances the discrimination of the selection by using two models representing indomain and out-of-domain patterns, respectively. While the indomain data is given, the out-of-domain, generally, comprises a sample from the out-of-domain conforming in size to the indomain data. Next, two language models are computed from the two samples. Sentences in the out-of-domain corpus are, then, scored by the difference in cross-entropy values obtained from the two respective models for each sentence. Sentences with differences exceeding a given threshold are retained. Equivalently, A number K could be set to keep only the K-top scoring sentences.

Moore and Lewis (2010) pointed out and compared to two other relevant methods. The first uses only the perplexity (or equivalently the cross-entropy) evaluated with the indomain model; and hence no sample from the out-of-domain is selected. This method was first introduced by Lin et al. (1997), and later adopted by Gao et al. (2002). The problem here is to attribute a subject to a newly received document. The system has several models each of which represents a given subject. The new document is evaluated with bigram models corresponding to the different subjects and then it is classified in the most similar subject.

The other method is due to Klakow (2000). It, somehow, goes in the opposite direction of the previous work. it uses the intuition that sentences whose removal hurts more the likelihood of the indomain data have to be kept. The score of a sentence is the difference in likelihood of the indomain set evaluated by a model from the out-of-domain after removing this sentence. The author uses a unigram model for this purpose.

The method of Moore and Lewis (2010) was, later, adapted to bilingual data by Axelrod et al. (2011). The extension consists of using Moore and Lewis (2010) on both the source and target sides and then each pair is scored with the sum of the two differences in both languages. Using cross-entropy difference for bilingual data selection was further extended, by Mansour et al. (2011), to include alignment information in the model. The sum of the two differences in cross-entropy is combined with another sum of two differences in cross-entropy of an IBM model 1 between the source and target sentences. The two IBM 1 differences match the evaluation of the alignments in two directions. The combination between the LM differences in cross entropies and the IBM 1 differences is performed through a weighted linear expression. The weight is tuned on a held out set.

2.3.3 Weighting Training Data

To our knowledge, the only work which considers data weighting as an alternative to filtering, to emphasize the impact of the good data, is due to Zhang and Chiang (2014). The approach was presented as an application to a proposed adaptation of the Kneser-Ney smoothing to fractional counts. No data is filtered out, but rather every sentence has a weight correlating with its score according to Moore and Lewis (2010). The tricky part here is how to train a language model on weighted sentences. They exchange every true count in Kneser-Ney smoothing with its expected value calculated from the weighted occurrences of different n-grams.

2.3.4 Other Related Work

In addition to the filtering and weighting of data, we have some contributions to extracting semantic word associations and probability smoothing. Here, we point out the most important research which will serve a basis of our work .

Extracting semantic associations between words was undertaken earlier by Callison-Burch et al. (2006). Actually, the units considered in this work are not limited to single word, but also span to phrases. The idea here is to exploit aligned parallel corpora to generate paraphrases. The assumption is that phrases on one side which often translate to the same phrase on the other side, should share some meaning. While this approach works remarkably well, it requires aligned bilingual data which is not always easy to get. Alternatives which make use of monolingual data are numerous, but there is a tendency among the community to converge to low-dimensional word representations. This trend was started with the introduction of the `word2vec` tool by Mikolov et al. (2013a). The vectors in this case are the word projections in the hidden layer of a simple neural network architecture. The goal of this network is to predict the context given the head word, or the other way around. The vectors are computed so that the likelihood of the observed cooccurrences in a training corpus is maximized. A slightly different method was proposed, later, by Pennington et al. (2014). In this method, the vectors are chosen so that their dot product approximates the logarithm of the cooccurrences. This approximation is formulated as a weighted least squares regression.

Probability distribution smoothing while modeling from fractional counts is by far rarely brought up in the literature. Probably because it is only needed in some rare cases. A scenario where this kind of counts may arise is the data weighting discussed above. To train language models on a sentence-weighted corpus, Zhang and Chiang (2014) uses the attributed weights to compute expected (or weighted) occurrences of n-grams. From the expected occurrences, expected values for the counts of counts are estimated. The smoothing is carried out similar to Kneser-Ney by exchanging the true counts by their expected values.

2.3.5 Efficient Phrase Scoring

The process of scoring the phrase pairs extracted from an aligned parallel corpus is an expensive task. Typically, the number of the resulting phrases is extremely large. This implies that in most cases the scoring task cannot be completed in memory, and therefore the usage of an external physical disk is mandatory. However, for very small systems, Hardmeier (2010) proposed to accomplish the operation in memory. Indeed, the phrases are indexed in a lookup hash table. Therefore, direct access is possible to update the underlying phrase counts.

The more common approach to scoring, however, is to use external memory sorting as implemented in Moses (Koehn et al., 2007). This tool uses the system's `sort` command to perform two successive sorts. The first sorting collects phrase cooccurrences and the marginal counts for the source phrases, while the second completes the process by collecting the marginal counts of the target phrases.

Lots of work has been done to implement sorting with the help of external memories, and as a result many software platforms which make the swapping operations transparent to the programmer have been developed. These platforms rely on using fixed size records to facilitate retrieval from the disk. For example, the STXXL framework offers the possibility to overlap processing and the IO and that many disks can be read/written at the same time (Dementiev and Kettner, 2005).

2.4 Conclusions

In this chapter, we presented the core elements used through out this thesis. We presented the statistical association measures which will be used in many situations to discover noisy cooccurrences. For instance, they will be used in Chapter 7 to identify the bad n-grams or the bad phrase pairs. We also glanced over the different components of an SMT system. We concentrated more on those components which will serve special purpose in the next chapters. We also described our Baseline system which will be improved throughout the thesis. A sample of the related research was surveyed, as well. However, detailed discussion of the relation of our work to each of those will be elaborated in the appropriate chapters.

Our contributions will start in the following chapter. We will first present our study on semantic association extraction, which is one of the contributions we labeled as Helper Contributions in Figure 1.2 in the Introduction. The semantic associations will help us develop our work on data selection in Chapter 6.

3 Semantic Word Associations

Establishing semantic relations between words is an old-standing problem in NLP. Synonymy, antonymy, and polesymy are instances of semantic relations.[1] Manual efforts have led to the creation of very useful resources for these relations. WordNet(s) and online thesauri are examples of such resources which are freely available.[2,3] Obviously, the manual creation of these resources is expensive and of low-coverage.

Research in "Distributional Semantics" has introduced automated techniques to overcome the limitations of the manual semantic resource creation. These techniques are mainly based on the distributional hypothesis (i.e. words which appear in similar contexts share some meaning). Consequently, semantic relations are drawn from word cooccurrence data. Deerwester et al. (1990) was one of the early publications in this area. They proposed the Latent Semantic Indexing (LSI) for indexing information retrieval. A more recent review of the applications of distributional semantics is presented in Bruni et al. (2014).

In our work, the semantic associations are used in the context of data selection. Performing selection on noisy data is a very beneficial operation as it reduces both the noise and the training corpus. Usually, data selection relies on exact word matches between a small clean indomain set and the corpus to be filtered. The semantic associations extracted using the methods described in this chapter will allow for also matching semantically related words.

[1]These relations and five more, explained and exemplified, can be found at `https://en.wiktionary.org/wiki/Wiktionary:Semantic_relations`

[2]The English WordNet: `https://wordnet.princeton.edu/`

[3]Wiktionary includes a thesaurus (Wikisaurus): `https://en.wiktionary.org`

For our purposes, we are more interested in synonymy relations. We use them to generate arbitrary equivalent n-grams by simple substitution of synonym words. This procedure is employed for data selection in Chapter 6. In this work, we explored two ways of generating the synonyms, by exploiting second order cooccurrences in bilingual and monolingual data.

3.1 From Aligned Bilingual Corpora

In a word-aligned parallel corpus, the distributional hypothesis can be interpreted as follows: Different source words aligned to the same target word should share some meaning (and vice-versa).[1] For example, the French word "faire" has 256 possible alignments in our Fr→En clean corpus (13.6 million English words and 15 million French words). The top 10 frequent aligned English words are shown in Table 3.1 with the number of times they were aligned to the French word. It is obvious that some of these alignments are noisy. For example "to" appears here because it can precede a good alignment point (e.g. "do") and the system choose to align it rather than leave it unaligned. Apart from these noisy alignments, meaningful associations exist between other English words (e.g. "do" and "make").

In order to quantify the strength of the relation between the different source words aligning to the same target word, we perform a pivoting over the target side. The target is therefore marginalized out by summing over all possible target words (t) which connect two source words (s_1 and s_2). Consequently, the probability of replacing s_2 by s_1 without disturbing the meaning can be calculated as follows:

$$\begin{aligned}
\Pr(s_1 \mid s_2) &= \frac{\Pr(s_1, s_2)}{\Pr(s_2)} \\
&= \sum_t \frac{\Pr(t)\Pr(s_1, s_2 \mid t)}{\Pr(s_2)} \\
&\approx \sum_t \frac{\Pr(t)\Pr(s_1 \mid t)\Pr(s_2 \mid t)}{\Pr(s_2)} \\
&= \sum_t \Pr(s_1 \mid t)\Pr(t \mid s_2)
\end{aligned} \tag{3.1}$$

[1]The use of "Source" and "Target" in this context is arbitrary. The "Source" is one part of the parallel corpus, where the "Target" refer to the other part.

English word	Num. Align.
do	4707
make	3471
to	1822
be	1617
done	864
making	498
doing	431
take	310
have	278
how	242

Table 3.1: English words aligned to the French word "faire" with the number of times they were aligned

In the second line of Equation (3.1), the probability expression is rewritten by introducing the aligned words t from the target side as a latent variable. In the third line, we simplified the expression in the previous line by assuming that source words are independent when conditioned on the target words.[1] The pivoting behavior can be best seen at the last line of the equation. The two terms in this last line correspond to the lexical translation tables (in two directions), which are usually extracted from word alignments in the fourth step of the Moses training process. Based on the example in Table 3.1, semantic relationship can be inferred between the English words "make" and "do". The score, however, will be computed based on all French words which connect these two words.

This approach was used by Callison-Burch et al. (2006) on a phrase level to generate paraphrases in the source language. However, an addition in our work is the use of association measures in order to reduce the effect of noisy alignments. For the same reason (i.e. to reduce the effect of unreliable alignment points), we consider only one-to-one alignments by using the Intersection combination heuristic (cf. Section 2.2.1.4). In spite of these restrictions, we still end-up with a huge number of pairs for a large corpus. This why we implement the approach in a very similar way to our phrase extraction,

[1]Note that this assumption is quite reasonable for reasonably-sized corpora. In fact, knowing the likelihood of one of the source words being aligned to the target word t does not tell much about it being aligned to the other source word.

Word	Assoc. Score
do	0.37
be	0.19
take	0.16
making	0.10
done	0.07
provide	0.06
ensure	0.05

(a) cooccurrence-based scoring

Word	Assoc. Score
do	0.49
take	0.13
making	0.10
done	0.07
are	0.05
doing	0.05
provide	0.04

(b) LLR-based scoring

Table 3.2: Words related to "make" with their association scores

which will be discussed in detail in Chapter 8. Very briefly, the implementation is parallelized both on processing and IO levels.

It is very interesting to note that Equation (3.1) does not assume any restriction on the latent variable t (i.e. the pivot language). Therefore, like Callison-Burch et al. (2006), combining multiple corpora from potentially different language pairs is possible. The only constraint is that these corpora should have one language in common (i.e. the language of s_1 and s_2 in Equation (3.1)).

From the previous example in Table 3.1, the most associated words to "make" are shown in Tables 3.2. To give an impression about the effect of association measures, the scores from two different measures are included in the tables. The normalized scores derived from the plain cooccurrence are in Table 3.2a, whereas Table 3.2b presents the normalized scores based on the log-likelihood-ratio measure. A noticeable difference between these two scorings is that the LLR scoring was able to diminish the high probability attributed to "be" by the plain cooccurrence. This should be an artifact of the high number of times that "be" was aligned to "faire" (or similar words) in a passive sentence.

3.2 From Monolingual Corpora

A more attractive resource to extract semantic associations between words is the monolingual data. This is especially true because this kind of data exists in much larger quantities as compared to its bilingual counterpart. The data which can be collected from the Internet is unbeatable in terms of quantity and topic diversity. However, the cost to pay for such data is to deal with greater amounts of noise.

The distributional hypothesis, in the case of monolingual data, usually means that words which share some context over a fixed window are likely to share some meaning. The common practice is to use the *word-context* cooccurrences to extract low-dimensional word vector representations. These vectors are, subsequently, used to derive word similarities using a similarity measure (typically the cosine similarity measure). Well-established techniques have been proposed to obtain the word vectors from monolingual corpora. Two popular categories of such techniques are matrix factorization and neural-networks.

In the following, we overfly one method from each category. The first category is represented by the Singular Value Decomposition (SVD) method, while `word2vec` serves as our chosen neural network based model. In addition, we present Global Vectors (GloVe), the model on which most of our experiments are based. Afterwards, we describe some improvements and extensions we introduced into this latter.

3.2.1 Matrix Factorization Approaches

Matrix factorization approaches infer low-dimensional word representations from high-dimensional cooccurrence matrix by factorizing this latter using the Truncated Singular Value Decomposition (TSVD) technique. Usually, the cooccurrences are recorded in a matrix ($\mathcal{X}$). The rows of this matrix correspond to the words and the columns to the contexts. Each entry $X_{w,c}$ of the matrix $\mathcal{X}$ reflects the number of times the word w cooccur with the context c.[1] Then, $\mathcal{X}$ is approximated with the factorization

$$\mathcal{X} \approx \mathrm{U}_D \Sigma_D \mathrm{V}_D^{\intercal} \tag{3.2}$$

Where V and U are unitary matrices and Σ_D is a diagonal matrix and D is the desired number of dimensions (i.e. only the D-largest singular values are kept).[2] The word vectors correspond, then, to the rows of U_D.

The nature and the amount of information contained in the derived vectors are heavily determined by the "context" definition. As an illustration, Bansal et al. (2014)

[1]Or alternatively, the entries record the PMI values of w and c.

[2]More details about the Singular Value Decomposition can be found in linear algebra textbooks (e.g. Banerjee and Roy (2014))

pointed out that using shorter contexts results in vectors of more syntactic nature, while longer contexts reveal more semantic vectors. Motivated by the targeted task, the interpretations of the term "context" in the literature vary from single words to very sophisticated relations calling other NLP tasks such as parsing. For instance, both Deerwester et al. (1990) and Landauer and Dumais (1997) used a whole document as context. Alternatively, Lund and Burgess (1996) used single words as contexts. Curran (2004)'s PhD dissertation presents an extensive study of different kinds of contexts and their effect on a thesaurus extraction task.

3.2.2 Neural Network Approach: `word2vec`

A more recent trend to create low-dimensional representations of words was initiated by Mikolov et al. (2013a) where the well-known `word2vec` tool was introduced. `word2vec` is based on a simple neural network architecture of an input, a hidden, and an output layer. Such network is used to predict the probability of a context given a head word or vice versa. Therefore, based on the direction of prediction, two sub-models can be inferred: (i) continuous bag-of-words (CBOW) model predicts the word given the context and (ii) the skip-gram (SG) model does the opposite direction. Both models will be approximately equivalent in the simplest case when considering only one-word context.[1]

The training of the `word2vec` models reduces to learning two matrices of sizes $|V| \times D$, where $|V|$ is the vocabulary size and D is the dimension of the word vectors. These matrices are determined so that the likelihood of the data is maximized. The probability of a pair roughly relates to the exponentiation of the dot product of the corresponding context and word vectors. In the case of CBOW, the likelihood to be maximized can be expressed as:

$$\begin{aligned} J &= \sum_{w,c} \log\left(\Pr\left(w \mid c\right)\right) \\ &= \sum_{w,c} X_{w,c}\left(\mathbf{w}\cdot\mathbf{c}\right) - \sum_{c} X_c \log\left(\sum_{w',c}\exp\left(\mathbf{w}'\cdot\mathbf{c}\right)\right) \end{aligned} \tag{3.3}$$

where $X_{w,c}$ denotes the number of times w and c cooccur in a corpus; $X_c = \sum_w X_{w,c}$. The input corpus is iteratively scanned and the parameters are updated according to

[1]We used the adverb "approximately" to reflect the fact that the update equations at the hidden and the output layer are different.

the cooccurrences in the current sentence. The optimization is performed using the Stochastic Gradient Descent (SGD). Great details about this process are presented by Rong (2014).

3.2.3 Global Vectors: GloVe

Introduced by Pennington et al. (2014), GloVe is an alternative model which stands somewhere in between the two formerly described categories. While its objective and optimization are quite comparable to `word2vec`, it operates on cooccurrence matrix similar to the factorization approaches. For a given word-context pair (w, c), GloVe attributes them vectors whose dot product best approximates the logarithm of the empirical cooccurrence. This approximation is accomplished by a weighted least squares regression, with the objective loss function:

$$J = \frac{1}{2} \sum_{w,c} f\left(X_{w,c}\right) \left(\mathbf{w} \cdot \mathbf{c} + b_w + b_c - \log\left(X_{w,c}\right)\right)^2 \tag{3.4}$$

where $X_{w,c}$ denotes the number of times w and c cooccur in a corpus; b_w and b_c are bias terms corresponding to w and c respectively and $\mathbf{w}$ and $\mathbf{c}$ are their associated vectors. f is a real non-decreasing positive function acting as a weight, having its values typically between 0 and 1. In its derivation, similar to `word2vec` objective, GloVe objective tries to approximate the conditional probability by the normalized exponentiation of the dot product of the vectors. However, it formulates the problem as a least squares and adds the bias terms to absorb any expensive normalizing terms. Here again, the optimization is performed by an SGD algorithm. The summation in Equation (3.4) is carried out over all word-context pairs that can be found in the training corpus.

It is worth mentioning that the performance differences between these models reported in the literature can be misleading. With the right model configuration, similar performance might be achieved by different models. Baroni et al. (2014) is one of the first researches to conduct a systematic comparison between `word2vec` vectors and their traditional mates (including SVD-based approaches). The authors of this comparison have a long history with the traditional distributional semantics and wished to prove that there will be no clear winner between these two approaches. To their surprise, they conclude that the `word2vec` vectors are way superior to the traditional ones. However, they left an open question in their conclusions that more exploration of the hyperparameter space may lead to different conclusions. This doubt was soon proved to be, indeed, apposite by Levy et al. (2015). They run experiments which include also GloVe in addition to the models examined by Baroni et al. (2014). In fact, they demonstrated that with hyperparameter fine-tuning and data preprocessing the performance of different methods could be brought to a comparable range.

3.2.4 Modified Global Vectors: mGloVe

In addition to its simplicity, GloVe, like the traditional methods, explicitly decouples counting cooccurrences from the model fitting. While this will have a major drawback on memory consumption,[1] it gives more control over the training process. It is, indeed, more suitable for applying different cooccurrence weighting and filtering techniques that have shown useful for traditional methods. Furthermore, repeated fitting with different parameter settings can be performed more quickly, as the counting has to be done only once. Certainly, this might also be possible with `word2vec` if one invests some effort in redesigning the implementation, but GloVe offers this flexibility off-the-shelf.

Our modification is driven by the massive number of weighting and cleaning techniques explored in the literature for the traditional approaches and which have not been exploited in GloVe. Most of our modifications are essentially inspired by the extensive experiments presented by Curran (2004) for automatic thesaurus extraction. In mGloVe, we mainly explore :

1. Different scorings of the cooccurrence matrix, and

2. Various context interpretations

3.2.4.1 Association-based Scoring

The first modification we introduce takes place in the preprocessing. The association measures are used to reduce the effect of pairs which are likely to be noisy. Based on the extracted cooccurrences ($X_{w,c}$), we compute the associations between different word-context pairs ($A_{w,c}$). We, then, apply these associations in one of the following ways to get the final score of the pair ($S_{w,c}$):

[1]This is only true if the cooccurrence table has to be fully loaded, which is the case in our implementation.

- As a weighting to the empirical cooccurrence. Our chosen weighting function in this case is of the form: $S_{w,c} = \frac{A_{w,c}}{1+A_{w,c}} X_{w,c}$. This way, the association has a strong influence when it is very small and becomes almost negligible as it grows very large. It is of course assumed that the association values are greater than or equal 0, which is the case for all our selected list of measures.

- As a replacement for the cooccurrence. We simply set $S_{w,c} = A_{w,c}$.

- In cases of non-weighted cooccurrence (i.e. when the raw cooccurrence or the raw association scores are used), the negatively associated pairs can be discarded (i.e pairs (w, c) s.t. $\Pr(w, c) > \Pr(w) \Pr(c)$, see § 2.1).

Using the association scores in the aforementioned manners poses a problem when the association measure is double-sided (e.g. LLR). Both very bad and very good pairs receive a high score by these measures. To avoid such a problem, we take the reciprocal of the score if the association is negative, and we add one to avoid dividing by zero:

$$A'_{w,c} = \begin{cases} 1 + A_{w,c} & \text{if} \Pr(w,c) > \Pr(w) \Pr(c) \\ \frac{1}{1+A_{w,c}} & \text{otherwise} \end{cases} \tag{3.5}$$

In addition, we only allow replacing the cooccurrence by the association score for measures with a value range comparable to the cooccurrence (typically between 0 and 10^6). Take the case of the Jaccard measure. It gives values between 0 and 1 and therefore, we use it only for weighting. If such small scores are used in lieu of the last term of the objective (3.4), the error becomes quickly small enough to let the SGD believe a convergence is reached.

While not explicitly stated in their corresponding papers (Mikolov et al. (2013a) and Pennington et al. (2014)), the implementations of both `word2vec` and GloVe weight the cooccurrences based on the distance between the head word and the context. Such weighting will result in fractional counts as it gives a cooccurrence value of 1 to only the immediate left or right context. Any further/former contexts are given a count value less than 1. As shown in Figure 3.1 for a window size of 20, `word2vec` uses a linear weighting while GloVe uses a hyperbolic (or reciprocal) weighting. We complete the figure by, also, adding an exponential weighting which decays exponentially as we move further from the head word. For the example in Figure 3.1, the sequence of weights for the hyperbolic weighting is $1, \frac{1}{2}, \frac{1}{3} \ldots$; for the linear weighting it is $1, \frac{19}{20}, \frac{18}{20} \ldots$;

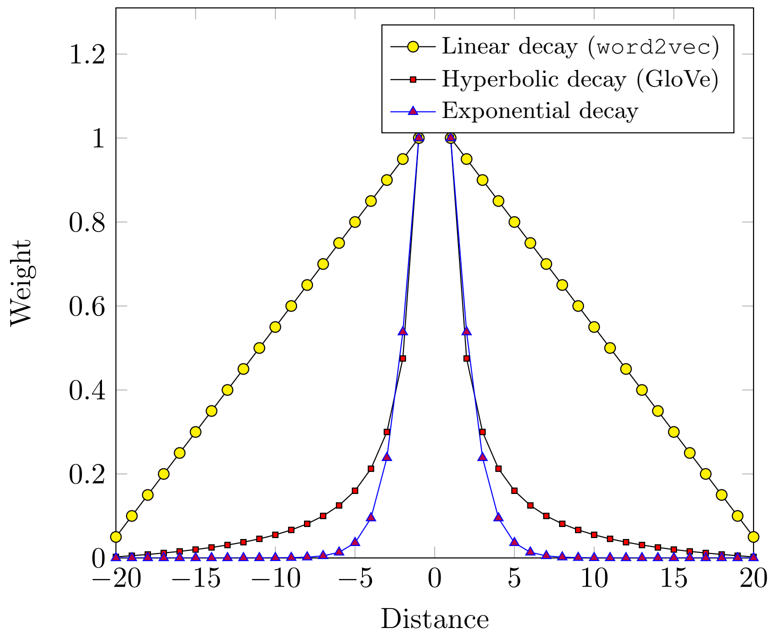


Figure 3.1: Context weighting based on distance from the head word

and for the exponential weighting it is $1, \frac{1}{e}, \frac{1}{e^2} \ldots$. We follow these implementations in weighting the cooccurrences in a window. To compute the association scores, we treat the accumulated fractional counts (from the whole corpus) as if they were integral and plug them into the formulas given previously for each measure (cf. Section 2.1).

3.2.4.2 Context Interpretations

Connections between words are established based on the contexts in which they appear. Therefore, the nature of relations found between the different words will be strongly determined by our definition of the term "context". As discussed before, the context in `word2vec` and GloVe is window-based. Basically, both consider the context of a head word to consist of all words occurring before and after that head word in a fixed-size symmetrical window. We generalize over this interpretation by allowing more flexible contexts. First, we go beyond ordinary contexts of words to also consider clusters of words. In addition, we examine the effects of treating the left and right contexts separately. We also explore ways to combine the information collected from word alignments with the monolingual pairs.

Using word clusters Word clusters are widely used in different NLP tasks. Specifically, Goodman (2000) shows that they can improve the performance of a language model considerably when properly combined with word-based models. Their main advantage is that they immensely reduce the vocabulary size, turning the sparsity, thus, into less of a problem. Moreover, with a small vocabulary, many prohibitively expensive algorithms become manageable.

Using word clusters for word vector extraction is inspired by the common practice of lemmatization and POS tagging in collocation extraction (see Pecina (2009) for details about common linguistic preprocessing for collocation extraction). To define the context of a head word, we consider the classes of the words surrounding it. By doing so and since the vocabulary of classes is much smaller, we are able to use much larger window sizes and/or higher n-gram orders without facing serious memory restrictions.

Various types of word clusters can be used. For instance, the clusters can be generated using a supervised model such as POS tags or lemmas. Alternatively, they can be computed unsupervisedly based on some similarity measure, such as MKCLS clusters. Consequently, the resulting vectors are likely to be biased in the information they capture. Using lemmas as clusters would result in more semantic vectors, while using POS tags would make them more syntactic.

Using the unsupervised clusters as contexts may sound recurrent as the classes themselves are found based on the distributional hypothesis. However, usually the formulation of the clustering problem is different from that of cooccurrence counting. As an illustration, Och (1995) formulates the MKCLS clustering similar to a Hidden Markov Model (HMM), while our cooccurrence extraction is formulated as a bag-of-words. This difference in formulation adds information to our cooccurrence counting process. Additionally, the extra overhead caused by the generation of clusters can be tolerated since these clusters are usually generated anyway during the training process for other purposes. The POS tags, for example, are generated to learn the reordering model and MKCLS classes are used to compute a language model which is used during decoding (cf. Chapter 2).

We use the clusters in one of two ways:

- Directly in lieu of the contextual words.
- To re-estimate a smoothed count for a given word-context pair.

The second method is motivated by the fact that counts collected for the clusters will be more reliable because of the reduced sparsity. Hence, the inferred association scores will be more precise. The re-estimation is performed as follows:

$$A_{w,c} = \sum_{cw} \sum_{cc} \Pr\left(w \mid cw\right) \Pr\left(c \mid cc\right) A_{cw,cc} \tag{3.6}$$

where $A_{w,c}$ is the re-estimated association of the head word w and the context c; cw and cc are their respective clusters, and $A_{cw,cc}$ is the association of clusters computed from the corpus. A derivation of Equation (3.6) is given in Appendix A.1. The summation signs are removed if the clustering is hard (i.e. when each word belongs to exactly one single cluster). As an example, MKCLS gives a hard clustering, while POS tagging is, in general, not hard. All the terms on the right side of Equation (3.6) are estimated from the training corpus.

Directional context Accounting for the direction of the word-context pair was proven to help both syntactic and semantic performance of the vectors. Indeed, experiments run by Curran (2004) show that distinguishing left and right contexts remarkably improve the thesaurus quality. On the other hand, Ling et al. (2015) goes beyond binary directions by recording more positional information for the contextual words for the purpose of generating more syntactically-informed vectors. In their experiments, they consider 5 positions in each direction, which will cause a memory burden for corpora of important size.

A natural example which motivates the directional contexts is to observe the words before and after the determiners. Determiners are followed by names, and often preceded by prepositions. Not distinguishing the direction of word-context pairs would lead to conclude that prepositions will be similar to nouns, which is not useful in general. Of course, this is an oversimplification because prepositions and nouns will also appear in many different contexts.

We follow Curran (2004) by separating left and right contexts. Unlike the other modifications presented earlier which all operated at the preprocessing, differentiating the cooccurrence directions has an impact on the training implementation. In fact, vectors should be allocated for head words, right contexts, and left contexts.

Adding evidence from word alignments It is necessary to combine the benefits from the monolingual and bilingual data. While the former has a wide coverage, we believe the latter delivers more precise counts for the pairs. The reason for this theory is that, in monolingual cooccurrence counting, we consider all surrounding tokens, while in word alignment only the most likely equivalent is linked to the head word. These linked (aligned) pairs should be equivalent in meaning. Therefore, this kind of contextual information is more important for finding synonyms in the source language.

We propose several variants to accomplish this combination:

- Feeding the union of both counts to the learning algorithm. Therefore, the monolingual word-context training examples are augmented by introducing the aligned word pairs as additional examples. This way, we are giving the same weight to the monolingual and bilingual data.

- In order to give more importance to the bilingual examples, we train the vectors in two successive steps:

  1. Train vectors for the intersection vocabulary (i.e. words appearing in the bilingual and monolingual training examples) using only the bilingual data.
  2. Train using the monolingual data by initializing the vectors of the intersection vocabulary with the vectors resulting from the previous step.
  3. During the SGD iterations, allow the vectors of the intersection vocabulary to be updated with a much smaller learning rate.

  As a consequence, our SGD algorithm must be modified to support two different learning rates. In the extreme case, we would keep the vectors of the intersection vocabulary unchanged, in Step 2, by setting the additional learning rate to 0.

- Use the same previous approach by replacing the bilingual counts with the semantic equivalences obtained using the approach described in Section 3.1.

  An important issue with this last variant is that the semantic equivalences we get by applying Equation (3.1) represent similarities rather than cooccurrences, and hence the GloVe objective ((3.4)) is not suitable to learn the corresponding vectors. The literature dealing with this problem is extensive. Multi-Dimensional Scaling (MDS) presented in Cox and Cox (2000) is an instance of this literature. MDS is a powerful approach which aims at learning low-dimensional representations which preserve as much as possible from the pairwise distances. In other words, for a given pair it tries to approximate the distance between the correspond vectors to the pairwise distance of that pair. The distance between the vectors here is, typically, the Euclidean distance:

$$J = \sum_{s_1,s_2} \left| \|\mathbf{s_1} - \mathbf{s_2}\|^2 - d^2_{s_1,s_2} \right| \tag{3.7}$$

where $\|\mathbf{s_1} - \mathbf{s_2}\|$ is the Euclidean distance between the vectors corresponding to words s_1,s_2; and d_{s_1,s_2} is the given pairwise distance. In its standard form, MDS optimizes this equation by computing the top eigen vectors of the pairwise distance matrix.

We compute our vectors from the bilingual semantic similarities using an objective function similar to Equation (3.7), but which is, in spirit, faithful to the GloVe implementation (so that we could re-use as much of the code as possible). Though, two important details about this implementation have to be mentioned:

1. The first detail is how we convert similarities into distances. We rely on Equation (3.1), but to avoid numerical instability, we keep the last term unnormalized (i.e. we replace the conditional $\Pr(t \mid s_2)$ by the joint probability $\Pr(t, s_2)$). Then we set the distance to be some form of the reciprocal:

$$d^2_{s_1,s_2} = \log\left(1 + \frac{1}{1 + \sum_t \Pr(s_1 \mid t)\Pr(t, s_2)}\right) \tag{3.8}$$

   We add 1 in the denominator to avoid division by 0 and add 1 to the distance to get a positive log value.

2. The second point is that we use a least squares formulation (without bias nor weight function), so that we can apply the SGD optimization:

$$J = \frac{1}{2} \sum_{s_1,s_2} \left(\|\mathbf{s_1} - \mathbf{s_2}\|^2 - d^2_{s_1,s_2}\right)^2 \tag{3.9}$$

   and to update a component s_{1_i}, we use the gradient:

$$\frac{\partial J}{\partial s_{1_i}} = 2\left(\|\mathbf{s_1} - \mathbf{s_2}\|^2 - d^2_{s_1,s_2}\right)(s_{1_i} - s_{2_i}) \tag{3.10}$$

Note also that in this optimization the two words are treated equally likely (i.e. no distinction between a contextual word and a head word). This makes the process more memory efficient, since only one single set of vectors is used for both words.

3.3 Implementation Issues

Some of the problems discussed in this chapter are known to suffer from high complexity. In particular, for reasonably sized corpora the bilingual semantic association and the mGloVe counting are very expensive. Here we briefly discuss this issue and present a couple of heuristics to reduce the cost of these computations.

1. **Semantic associations from word alignments:** As can be already realized from Equation (3.1), each two source words have to be connected through all their aligned target words. Therefore, the upper bound of the time complexity is $\mathcal{O}\left(|S|^2 \times |T|\right)$, where S and T are the source and target vocabularies respectively. Additionally, As each pair of source words has to be stored, the memory complexity is $\mathcal{O}\left(|S|^2\right)$. This high complexity is tackled by the following heuristics:

   - Consider only words with alphabetic characters. Any word containing a punctuation or a numeric symbol is avoided.
   - We force an upper threshold on the fertility of a target word. Any target word aligned to more than this threshold is discarded.
   - We force a lower threshold on the number of times a pair of words was aligned. Any pair aligned less than this number of times will not be considered.
   - We force a lower threshold on the scores of source word pairs. This threshold influences only the memory consumption. Any pair with a score less than the threshold is not recorded.
   - The implementation is highly parallel and uses the same framework as our phrase scoring presented in Chapter 8.

2. **mGloVe counting:** Counting itself is not different from GloVe. Since the approach is window-based, cooccurrence of any words appearing in the same window should be recorded over the whole corpus. The upper bound of time complexity of

the counting is $\mathcal{O}(|\mathcal{C}| \times W)$, where $\mathcal{C}$ is the corpus size and W is the window size. The memory complexity, however, is $\mathcal{O}(|V| \times |C|)$, where V is the corpus vocabulary and C is the context vocabulary. If only regular words are considered in the context (i.e. $V = C$), then the memory complexity becomes quadratic in the corpus vocabulary size. The overhead resulting from this complexity can be slightly reduced by considering only words occurring more than a certain threshold.

3.4 Evaluations

In this section, we present intrinsic evaluation of our contributions to the semantic association extraction. Extrinsic evaluation will be carried out in the chapter devoted to data selection (i.e. Chapter 6). The methods are particularly evaluated on English tasks. This limitation is imposed by the gold standard datasets which are mainly available in English. As a consequence, when monolingual data is needed, we utilize our 1 million-sentence corpus used to train our "big" English language model (cf. Chapter 2 for a description of our basic system). On the other hand, when bilingual data has to be used, we exploit our "de-noised" French $\rightarrow$ English corpus resulting from the best filtering settings in Chapter 5. This corpus is sampled from EPPS, NC , Giga, and Common Crawl corpora and consists of around one million pairs of sentences.

We assess the effectiveness of each of the presented methods by measuring their performance in three different tasks: semantic similarity, word analogy, and syntactic similarity. Note, however, that the last task (i.e. syntactic similarity) is added for the sake of completeness, since it slightly diverges from our primary aim to extract synonymous word pairs. In the following we describe the datasets used in each task and the methodology adopted for the evaluation.

3.4.1 Reference Datasets

In total, we use 6 references to carry out the intrinsic evaluation of the techniques proposed in this chapter. 3 are used for the semantic similarity, 2 for word analogy, and 1 for syntactic similarity evaluation. Each of these sets is briefly described bellow.

Dataset name	Num. pairs	Scale	Reference
WS-353	353	0–10	Finkelstein et al. (2001)
MEN	3000	1–50	Bruni et al. (2014)
SimLex-999	999	0–10	Hill et al. (2014)

Table 3.3: Word similarity evaluation sets

3.4.1.1 Semantic Similarity Datasets

This is the most important evaluation for our purpose because it is based on semantic similarities. The measure in this case is the correlation between our produced semantic similarities and those attributed by human evaluators for a set of word pairs. This correlation is commonly reported in terms of Spearman's rank correlation which is the correlation measured between the rankings of the system similarities and the human similarities. The closer to 1 the measure is, the better our similarities are.

Three datasets are used to evaluate the semantic similarities (WS-353, MEN, and SimLex-999). These sets were created in comparable manners: Human subjects are presented with English word pairs, and they have to either score their similarity or compare them to other pairs. The final similarity score is the mean of all human evaluations. It is worth pointing out that the SimLex-999 set was created to address semantic similarity rather than semantic relatedness which is a shortcoming of the WS-353. For instance, the pair *(coast, shore)* is given a high score by both datasets ($\sim$ 9), whereas the pair *(clothes, closet)* is given a low score by SimLex-999 ($\sim$ 2), unlike WS-353 ($\sim$ 8). More details about the three datasets are given in Table 3.3. Random examples of pairs from these datasets can be found in Appendix B.1.

3.4.1.2 Word Analogy Datasets

The word analogy tasks are designed to test the ability of the word vectors to recover word analogies by simple linear operations. As a result, the applicability of this task is not straightforward if we are only presented with similarity values rather than the vectors, which is the case when we obtain the semantic associations from bilingual data.[1]

[1]In reality, the task is perfectly applicable if we have the full similarity matrix. However, for example when we compute the similarities from bilingual word alignments, a large number of the matrix entries will be missing. This will generate a very serious limitation in the number of questions which can be evaluated.

Consequently, we do not consider evaluating on this task for the associations extracted from word alignments.

Word analogy datasets consist of a list of quadruples of words. The system is exposed with the first three and has to guess the fourth. The task is to find the fourth word whose relation to the third is similar to the relation associating the second to the first. For example, if the first word represents the name of the capital city of the country whose name is given as the second word, and the third word is also a capital, then the system has to find the name of the corresponding country. This is simply achieved by computing a new vector using simple operations, and then finding the word whose vector is most similar to this newly made-up vector. If $\mathbf{v}_i$ denotes the vector associated with word w_i, then the task consists of finding the word w_{x^*}, such that

$$x^* = \underset{x}{\operatorname{argmax}} \operatorname{Sim}\left(\mathbf{v}_2 - \mathbf{v}_1 + \mathbf{v}_3, \mathbf{v}_x\right)$$

where Sim is a measure of similarity between vectors, typically the cosine similarity.

The performance is measured in terms of accuracy, which is the percentage of correctly answered questions from all those which could be answered in principle. This means the number of correct answers is divided by the number of the examined questions for which all the underlying words exist in the vocabulary. We measure the accuracy of our vectors on two public sets of questions. The first is commonly referred to as the Google analogy dataset and is due to Mikolov et al. (2013a) (abbreviated as *GOOG-A* hereafter). Moreover, the second was created by Mikolov et al. (2013c) and is known as the Microsoft syntactic analogy dataset (Abbreviated as *MS-A* hereafter). Even though The Microsoft set and a part of the Google set are said to be syntactic, we still find them relevant to our purpose, because the corresponding syntactic relations are usually inflections derived from the same lemma. Consider, for instance, the quadruple (*good, better, rough, rougher*) from the Microsoft set. Albeit it is true that the second and fourth words represent respectively the comparative adjectives of their predecessors, the near semantic equivalence of *rough* and *rougher* is very obvious. Some additional information about the two datasets are presented in Table 3.4. Some examples extracted from both datasets can be found in Appendix B.2.

Dataset name	Num. questions	Type(s)
Google analogy	19544	Semantic/Syntactic
Microsoft analogy	8000	Syntactic

Table 3.4: Word analogy evaluation sets

3.4.1.3 Syntactic Similarity Dataset

We create a list of pairwise syntactic similarities between a set of words. This list is similar, in structure, to the Semantic datasets described in Section 3.4.1.1, and therefore the evaluation is performed in an identical manner. i.e. Spearman's rank correlation between the two lists of pairwise similarities is reported.

We construct the syntactic dataset from manually annotated corpus for Part-of-Speech (POS). For each word, a probability distribution over all possible POS tags is computed based on the annotated corpus. These distributions are then used to infer the pairwise similarities.

In our experiments, we use the Manually Annotated SubCorpus (MASC) which is a subset of the American National Corpus project (ANC).[1] This corpus is the fruit of collaborative efforts from the crowd to create gold standards for several linguistic phenomena. In particular, manual validation is carried out on automatic POS annotations of the corpus. Details about this process, among others, are given in a series of related publications such as Nancy Ide and Passonneau (2008) and Ide et al. (2010).

We consider all words appearing 5 times or more in the corpus. From these we keep only those which are pure linguistic tokens (no punctuation nor alphabetic symbols are included). This results in 8 007 words tagged with 50 different tags. From the corpus, we construct normalized vectors for each word according to its distribution over the 50 different tags. We compute the pairwise similarities using these vectors.

In theory, any similarity measure can be used for the pairwise similarities. However, since the vectors represent probability distributions, we chose to use the Jensen-Shannon divergence (JS) measure. Which is a symmetric version of the Kullback-Leibler divergence (see for example Lin (2006) for details about these measures). JS is a dissimilarity measure having values between 0 and 1, therefore we use $1 - JS$ as our similarity mea-

[1]`http://www.anc.org/`

Technique	WS-353	MEN	SimLex-999	MASC
COOC + Grow-Diag	0.4009	0.5278	0.3416	-0.0808
COOC + Inter	0.5109	0.5169	0.4439	0.06501
LLR + Inter	0.5726	0.5493	0.5092	0.1310
Jaccard + Inter	0.6031	0.5699	0.5855	**0.1456**
GMean + Inter	**0.6247**	**0.5791**	**0.5902**	0.1364
COOC + Inter (EN–FR + EN–DE)	0.4790	0.5020	0.4454	0.0667
COOC + Inter (EN–FR + EN–AR)	0.4215	0.5222	0.4955	0.0672
GMean + Inter (EN–FR + EN–AR)	0.5860	0.5799	*0.6311*	0.1382

Table 3.5: Evaluation of semantic associations obtained from bilingual alignments

sure. By keeping only pairs with a strictly positive similarity, the dataset contains around 15 million pairs in total. A sample of this dataset is given in Appendix B.3.

3.4.2 Semantic Associations from Word Alignments

In the following, we examine the semantic associations retrieved from word alignments which were described in Section 3.1. Through a series of experiments, we evaluate the utility of three traits. First, restricting the type of word alignments to one-to-one discards considerable amounts of noise and brings important improvements for most datasets. We also show the advantage of replacing the raw cooccurrences by association measure scores. In most cases, all association measures outperform the raw cooccurrence. Finally, we shed some light on using additional alignments from a corpus with a different target language. Unfortunately, these additional alignments did not add useful information to the baseline.

The results of our experiments are summarized in Table 3.5. The table consists of three sections (separated by horizontal dashed lines). These sections correspond, in order, to the three aforementioned approaches (i.e. type of alignment, association measures, and using alignments from other language pairs). The near-zero scores in the right-most column are to be expected. In other words, the syntactic information is poorly captured by the associations acquired from word alignments. In great part, this is an implication of the structural differences between the source and target languages. Examining the tagged version of the bilingual data used in this task reveals some interesting sources creating this kind of ambiguity. For example, English comparative adjectives get aligned to an adverb 49% of the time. 78% of these adverbs are "*plus*" or "*moins*". This is a result of using the adverbs "*plus*" or "*moins*" in front of adjectives to build the comparative forms in French. Consequently, some English adverbs like "*increasingly*" will have a strong association with comparative adjectives such as "*greater*". While the semantic relation between these words is apparent, such links will weaken the syntactic informativeness of the inferred associations.

The first two rows in Table 3.5 show the importance of the precision of the alignment points, for the semantic association extraction. As commonly practiced, the alignments of both configurations are created from combining two independent runs of Giza alignments in two directions (i.e. French → English and English → French). The alignments in the second row (COOC+Inter) consist only of points retrieved in both directions. On the other hand, the alignments in the first row (COOC + Grow-Diag) include the intersection points and add some neighboring points from the union when some heuristic conditions are met. Intuitively, the former alignments are more precise than the latter because they contain only points upon which the two alignments agree. This, indeed, results in improved performance in almost all conditions. In particular, the improvement on SimLex-999, which is more important for our purpose, is the most important ($\sim 30\%$).

The second section in Table 3.5 illustrates the effect of using the association measures. The common difference which distinguishes these three configurations from the one immediately above them (i.e. COOC + Inter) is replacing the raw cooccurrence counts by a given association measure score. The tested measures are, respectively, the log-likelihood ratio, Jaccard, and the geometric mean measures (cf. Section 2.1). In all conditions, these measures substantially improve over the raw cooccurrence. Notably, the syntactic performance is doubled. If we reconsider the previous example, the probability of aligning French adverbs to English comparative adjectives drops to 4% with the GMean and to 3% with the Jaccard measures. Except for the syntactic dataset, the best performer is the GMean measure with around 0.60 correlation for all datasets.

The improvement obtained by using the association measures can be explained by their ability to discriminate noisy alignment points. In fact, some alignments happen because some single words in one language are decomposed to multiple words in the other language. One frequent example of this behavior is encountered when a passive sentence is translated into an active one, and the auxiliary in one language gets aligned to the main verb in the other language. Another common example is the English word "*to*" in the infinitive verb forms. It is usually aligned to the infinitive verb in French. As a result, the auxiliary verbs and the word "*to*" will cause noisy associations with a large number of words. Some examples illustrating this idea for the word "*make*" can be found in Appendix C.1.

The last section in Table 3.5 shows the results of adding information from an English–German and an English–Arabic corpora. The former corpus is comparable in size and in domain to the English–French corpus used in all other tasks (i.e. around a million sentence pairs from Parliamentary and News domains), whereas the latter corpus is fifth the size of the English–French corpus and is from TED data. We believe that the main reason behind the limited improvements gained by using this additional data is due to the increased ambiguities of the German and Arabic languages. Both languages are morphologically rich and involve some sort of word compounding. It seems that the challenge introduced by this additional ambiguity cancels the effect of the extra information. However, once again, using the GMean measure, in the last row, helps stressing some of the extra information and ends up in small gains.

3.4.3 Semantic Associations from Word Vectors

In this section, we evaluate the different modifications introduced in the mGloVe model. Since the number of all combinations of modifications is large, we try to group these modifications into four related groups, explored in the following subsections. The first subsection presents Baseline models built using `word2vec` and the original GloVe implementations, where we will discover a serious shortcoming of the GloVe model as compared to `word2vec`. We argue that the `word2vec`'s negative sampling is a key component in the model's success. Consequently, we suggest a straightforward implementation of negative sampling into mGloVe and compare its outcome.

In the second subsection, we examine the modified scoring. More precisely, we show the effect of different weighting methods (linear, hyperbolic, and exponential) and association measures on scoring the cooccurring pairs. After this, we evaluate the different context interpretations. In other words, we investigate the usage of word n-grams and word clusters as contexts as well as the distinction between right and left contexts. Finally, we demonstrate the benefits of combining the semantic information extracted from the bilingual word alignments with that in the monolingual data.

3.4.3.1 Baseline and Negative Sampling

From the two `word2vec`'s submodels, we only consider the Skip-Gram (SG). It is often reported to outperform the CBOW variant (See for instance the results presented in the original `word2vec` paper by Mikolov et al. (2013b)). In addition, GloVe in nature is more similar to the SG model than to the CBOW model; as both GloVe and SG models consider each pair in the window separately, rather than the CBOW way of averaging the whole context into one vector.

The results shown in Table 3.6 were obtained from vectors trained with the following hyperparameters:

- Context window includes 2 words before and 2 after the head word.
- The training was performed using SGD with 15 iterations.
- The word vectors consist of 300 dimensions.
- The GloVe model implements an adaptive learning rate (i.e. AdaGrad which was originally proposed by Duchi et al. (2011)), whereas `word2vec` has a learning rate decaying linearly in the number of observed examples.
- No minimum-frequency cut-off threshold was applied.
- The default negative sampling distribution was used in `word2vec` (i.e. unigram frequency raised to the power 0.75).
- The default weight function was used in GloVe (i.e. $x_{max} = 10$ and the power is $\alpha = 0.75$).
- It is important mentioning that unlike the original GloVe implementation, we do not add the two vectors associated with a given word, once as a head word and once as a context word.[1]

The preliminary results in the first two rows in Table 3.6 reveal a very interesting problem. GloVe in its original form performs very badly as compared to `word2vec` when, as in our case, the training corpus is small. In fact, the comparisons we came

[1]The reason we do not combine vectors in this manner, even though it is very beneficial performance-wise, is that this becomes non-trivial for some context interpretations, such as when we use word clusters.

Model	WS-353	MEN	SimLex-999	MASC	GOOG-A(%)	MS-A(%)
word2vec	**0.5206**	0.4859	**0.3247**	0.2507	20.56	**34.03**
GloVe	0.3094	0.2875	0.1564	0.2540	10.73	25.76
GloVe+NS	0.5332	0.5711	0.3082	0.2081	24.41	34.40

Table 3.6: Results of basic word2vec and GloVe word vectors

word2vec: word2vec original implementation; GloVe: original GloVe implementation; GloVe+NS: GloVe with negative sampling

across in the literature (such as Levy et al. (2015)) and in some online blogs[1] are reported for large datasets (essentially the 6 Billion English Wikipedia corpus) with a very high cut-off threshold, typically between 70 and 100. In these settings, the reported performances are usually comparable. Surprisingly, when the dataset is small and no cut-off threshold is applied the performances become incomparable in favor of the word2vec model. A crucial difference between the two models is, indeed, the word2vec's negative sampling (NS).

The word2vec objective consists of normalized probabilities over the whole vocabulary for each word. The NS is a clever way to approximate these expensive normalization terms (of quadratic complexity) in that it normalizes only over a sample. For each observed pair (positive example), a set of negative pairs is generated by pairing the head word with randomly sampled contextual words. Intuitively, this procedure minimizes the distance between the vectors associated with positive examples while maximizing that between the negative examples. A good pair will cooccur enough times to let the distance minimization beat the maximization due to the random sampling. More interestingly, this sampling procedure will eventually come across many word pairs which will never be encountered in the training corpus. We believe it is this kind of pairs that GloVe is missing when trained on a small dataset. GloVe operates, indeed, on the cooccurrence matrix, which will be highly sparse for small datasets, and the missing entries are simply omitted during the training. This point of view is backed by the last row in Table 3.6. This row is the outcome of the introduction of an operation akin to the word2vec's NS in the GloVe training.

Adopting a procedure very similar to word2vec's NS, each training example generates a number of negative examples proportional to the number of occurrences of

[1]To the date of writing, see for example http://rare-technologies.com/making-sense-of-word2vec/ and http://dsnotes.com/articles/glove-enwiki

this positive training example. The head word of the positive example is kept in all generated negative examples, while the contexts are sampled from the set of contexts except the one appearing in the original positive example. In order to keep the same GloVe objective function, the negative examples are attributed a score smaller than any observed pair score.[1] The last row in Table 3.6, shows how effective this procedure is, even though it is very simplistic.

The aforementioned NS procedure consistently outperforms the original implementation; in most cases by a very large margin. The performance becomes, accordingly, competitive to `word2vec`. Interestingly, by comparing the last two rows of Table 3.6, the impact of the information captured by this NS procedure is more remarkable on tasks with a semantic bias. More precisely, the improvement on the datasets which have a syntactic flavor does not exceed 20% (15% on MASC and 20% on Microsoft analogy questions). On the other hand, the performance boost on semantic datasets ranges between 40% and 50%. This can be explained by the fact that syntactic similarities can be categorized in a very small number of categories (e.g. verbs, adjectives,. . .) and therefore need less data. By contrast, the semantic similarities incorporate more ambiguity arising from their hierarchical nature. As a result, most of the syntactic information is already captured by the positive examples, and the model's semantic informativeness is the one which gains more from the negative examples.

3.4.3.2 Pair Scoring

In this set of experiments, we explore the different techniques to score cooccurring pairs. The results are shown in Table 3.7. The first panel of this table, compares the different weightings of a cooccurrence in function of the distance separating the corresponding pair. In the second panel, we show the effect of applying the Loglikelihood ratio measure (LLR) instead of the raw cooccurrence. We did not include other measures in the table as they all performed significantly worse on all datasets.

The first thing to note from the two panels is that, in average, the Linear weighting, or the Hyperbolic weighting which is equivalent in this case, performs slightly better than the other weighting techniques. It is noteworthy, however, that unlike the linear

[1]In our experiments, we found it reasonable to set this score to the minimum observed score -1.

Model	WS-353	MEN	SimLex-999	MASC	GOOG-A(%)	MS-A(%)
Panel A: Weighting						
No weighting	0.5154	0.5776	0.3078	0.1809	24.81	32.97
Lin/Hyp	0.5332	0.5711	0.3082	0.2081	24.41	34.40
Exp	0.5048	0.5629	0.3068	0.2084	23.69	34.88
Panel B: Association measure						
LLR+ No weighting	0.5466	0.5577	0.3212	0.2206	21.17	26.73
LLR+ Lin/Hyp	0.5658	0.5734	0.3262	0.2165	21.33	28.28
LLR+ Exp	0.5262	0.5712	0.2984	0.2172	22.70	29.59
pLLR+ Lin/Hyp	0.5023	0.5661	0.3067	0.2464	24.03	36.01

Table 3.7: Comparison of different scoring approaches

Hyp: Hyperbolic weighting; Lin: Linear weighting; Exp: Exponential weighting; LLR: Loglikelihood ratio association measure; pLLR: LLR used to penalize the cooccurrences

weights, the hyperbolic weights are independent of the window size.[1] Therefore, further comparison between these two needs to be performed for a larger window.

The next phenomenon is the unequal performance of the LLR measure. It slightly improves the scores on the semantic and syntactic datasets, but on the other hand it remarkably hurts the performance on the analogy tasks. This suggests that the simple algebraic operations on the vectors learned from LLR scores are less meaningful than on the vectors learned from the raw cooccurrence. Further analysis is needed to understand this behavior. Nevertheless, the performance balance could be somehow regained when the LLR scores are used for penalizing to raw cooccurrences in the final row in Table 3.7.

We think that the small differences between the different scoring techniques are most likely due to the window size used (i.e. $w = 2$). In particular, it is traditionally accepted that the unweighted cooccurrence counting corresponds more to a bag-of-words model which should have more influence on the semantic performance. However, due to the narrow window used, this does not hold in these experiments.

3.4.3.3 Context Interpretation

In this section, we show the effect of different interpretations of word contexts. The results are given in Table 3.8. The result of the linear weighted cooccurrence is replicated

[1]For example, the second word in the window will always get a weight of $\frac{1}{2}$ in hyperbolic weighting. Whereas in linear weighting its weight becomes $1 - \frac{1}{w}$, where w is the window size.

as Baseline here, for convenience. The upper panel of this table examines the influence of adding the direction information to the training pairs. If compared to the settings in Table 3.7, here we distinguish the left and right contexts by using two different context matrices. Every training example commits an update to a vector in the matrix corresponding to the head words, and another update exclusively in one of the two context matrices, based on whether the contextual word appears to the left or to the right of the head word.

The first panel in Table 3.8 suggests that the direction of the context is only useful in a syntactic task. In contrast, it hurts in the semantic relatedness tasks (i.e. WS-353 and MEN), and has almost no effect on the semantic association task. It performs particularly well when combined with the LLR scores, where these latter are used as penalizer (the last row of Panel A in Table 3.8). However, like the previous table, using LLR scores to penalize the raw cooccurrence causes the performance on the semantic relatedness tasks to slightly diminish. On the other tasks, it has a positive effect.

Panel B of Table 3.8 gathers the results of using two types of word clusters: POS tags and lemmas. Both were produced using TreeTagger.[1] While the vocabulary of the POS tagged corpus is very limited and consists of 57 different tags, the lemmatization slightly reduces the original corpus vocabulary by 25%.

The direct usage of the two types of clusters is presented in the first two rows of Panel B. It is not surprising to see such a low scores for POS contexts, as most of the semantic information is lost with the extremely reduced vocabulary. It is not surprising, either, that this same row also comprises the highest syntactic score recorded in all experiments, since the MASC dataset itself is built using a manually tagged corpus. The lemmatized contexts, on the other hand, play a decent role in stressing the semantic informativeness of the resulting vectors. This is especially true for the SimLex-999 dataset, for which this kind of categorized contexts record the highest score on this dataset so far. However, these contexts seem to fail in the analogy tasks. This confirms that the analogy tasks are a complex combination of semantic and syntactic information.

In the next two rows (POS+Word and Lemma+Word), we combine the cluster-based and the word-based vectors. The training is performed as follows: Cluster vectors are learned from the the tagged (or lemmatized) corpus. They are trained with a reduced dimensionality (say 10% of the total dimensions; i.e. 30 in our case). These vectors are

[1]`http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`

Model	WS-353	MEN	SimLex-999	MASC	GOOG-A(%)	MS-A(%)
Baseline	0.5332	0.5711	0.3082	0.2081	24.41	34.40
Panel A: Directional context						
Lin/Hyp	0.5046	0.5306	0.3122	0.2843	22.64	38.21
LLR	0.5254	0.5289	0.3271	0.2603	21.39	33.00
pLLR	0.4937	0.5170	0.3266	0.3228	22.16	39.92
Panel B: Using word clusters						
POS	0.1040	0.1354	0.0574	0.7016	0.48	0.32
Lemma	0.5425	0.5635	0.3525	0.1687	6.78	15.11
POS+Word	0.4842	0.4478	0.2680	0.3818	17.97	35.24
Lemma+Word	0.5372	0.5431	0.3248	0.2126	24.69	44.61
+pLLR	0.5297	0.5316	0.3243	0.2688	24.56	44.52

Table 3.8: Comparison of different context interpretations

Hyp: Hyperbolic weighting; Lin: Linear weighting; LLR: Loglikelihood ratio association measure; pLLR: LLR used to penalize the cooccurrences

used to initialize the word vector training. A part of each word vector is initialized with a weighted sum of the cluster vectors (the initialized portion in our case corresponds to the first 30 dimensions). The weights are simply the probability distribution of the corresponding word over all possible clusters, learned from the corpus. The remaining dimensions (i.e. 270) are initialized randomly. The portion of the word vector initialized by the cluster vector, will be kept unchanged.

This procedure, nicely, combines features learned from the plain words and the word clusters. The performance considerably outperforms the previous settings, where only the clusters were used. In particular, the performance on the Microsoft analogy task is remarkably improved when the Lemmas are combined with the words. Again, using the LLR measure as penalizer pushes the performance on the syntactic task further, while keeping the semantic and analogy performances in a comparable range. Nevertheless, there was no specific reason for the number 30 to be the dimensionality of the cluster vectors. We regard this number as an additional hyper parameter, which eventually sustains further tuning.

3.4.3.4 Adding Information from Word Alignments

We saw that using word alignments to derive the semantic associations surpasses all the other techniques in the monolingual setting. The idea of combining the two sounds very

Model	WS-353	MEN	SimLex-999	MASC	GOOG-A(%)	MS-A(%)
Align(fr)	0.3957	0.2380	0.5218	0.1231	15.01	32.34
+mono	0.5501	0.4984	0.4624	0.2983	33.95	47.30
Align(fr+ar)+mono	0.5895	0.5249	0.4726	0.2880	33.69	49.62
Align(fr+de)+mono	0.5787	0.5296	0.4834	0.2912	35.12	49.29

Table 3.9: Results of combining monolingual and word alignment word vectors

Hyp: Hyperbolic weighting; Lin: Linear weighting; LLR: Loglikelihood ratio association measure; pLLR: LLR used to penalize the cooccurrences

appealing as it may lead to the best of the two worlds: larger coverage and more accurate associations. Although the combination will be using more data, and it is difficult to conduct a fair direct comparison to the combined models. This scheme illustrates a common practical situation, where one would like to maximize the exploitation of the data at hand. In this section, we look into a possible realization of this idea. We use the procedure mentioned in Page 49. Vectors are trained from word alignments for the corresponding vocabulary. Then, they are used to initialize the training on the monolingual corpus. However, the SGD updates on the initialized vectors are controlled with a much smaller learning rate than that used for the vocabulary appearing in the monolingual corpus only. The results of this combination are shown in Table 3.9.

The first row in Table 3.9 presents the evaluation of the vectors extracted from word alignments only using the Multidimensional Scaling (MDS). These vectors are learned from the Geometric-mean-based similarities (see the fifth row in Table 3.5). Unfortunately, we fail to reproduce a comparable performance to that obtained from these raw pairwise Geometric-mean similarities. Yet, the gap might be tightened by further tuning in the hyper parameter space, but this is out of our scope.

For us, the most important fact is the large improvement on the SimLex-999, which is most related to our goal. This improvement can be seen in all of the settings in Table 3.5). To our surprise, the highest improvement for this dataset is obtained for the configuration which is the worst for all other datasets (i.e. the MDS from word similarities). Except for this dataset, adding the monolingual data (in the second row) has a very large positive influence on the performance. Depending on the dataset, using the similarities from multiple corpora brings slight improvements here and there. Also, unlike the situation with the pairwise similarities presented in Table 3.5, the German corpus shows a marginal superiority to the Arabic corpus; which should be expected as the former is reasonably larger.

Even though the performance of this combined model could never outperform the raw pairwise similarities, it equips us with a very useful tool. The word vector representations obtained from the combined model are generally more desirable than the pairwise similarities, in many NLP tasks.

3.5 Conclusions

In this chapter, we presented the techniques we use to extract semantically-associated pair of words, automatically. We showed how this automatic extraction can be performed on parallel and monolingual corpora. We demonstrated the effectiveness of these techniques intrinsically on several datasets. An extrinsic validation of the techniques will follow when we use them in data selection. The extracted pairs will serve as a tool for better indomain data selection, when the selection of a sentence will rely not only on the exact word to word match, but also consider the semantic equivalences.

The first approach we explored exploits word-aligned bilingual corpora. The idea comes from the nature of the word alignment, which eventually aligns different occurrences of a target word to different source words. We use this characteristic to connect these different source words. Our contribution here resides in the usage of association measures to clean up the established connections between the source words. We found that the Geometric mean outperforms the other measures and gives about $30\% \sim 40\%$ improvement over the baseline. We, also, showed the possibility of combining multiple bilingual corpora from potentially different languages to extract the semantic associations. While this combination does not always result in an enhanced model, it remains a very good resort in the case of data scarcity.

The second set of techniques extract the semantic associations using word vectors estimated from monolingual corpora. The main advantage of this approach over the bilingual one is its coverage and availability. Indeed, the monolingual corpora are much cheaper to collect and exist in large quantities for almost any language. We suggested different improvements to the GloVe model. For instance, we used association measures, again, to clean up the cooccurrence scores of word pairs. We also proposed combining cluster vectors with word vectors to bias the resulting vectors to hold more semantic or syntactic information. In addition, we presented a simple, but yet, effective way to overload the vectors with more information from the bilingual data. However, more interestingly, the most important improvement of this model appeared after equipping it with “negative sampling”. We added this powerful feature to the model, since it could not compete with the `word2vec` model in a small data scenario. All in all, we could improve over the original implementation of the model by more than 60%.

It is worth mentioning that we experienced many inconsistencies between the datasets used in our intrinsic evaluation. For example, even though WS-353, MEN, and SimLex-999 are commonly used to evaluate the semantic properties of word vectors, we may come across some settings which considerably improve one and in the same time significantly decrease another. The same situation happened for MASC and Microsoft analogy questions which is known to be syntactic. In such cases, we relied on SimLex-999 to decide about the direction to follow next, as we felt that this dataset is the closest to our goals.

It has not escaped our attention that the hyper parameter space is way far from being fully explored. For instance, the window size (which was fixed to 2 in our experiments), the vector size (300 in out experiments), and the number of iterations (fixed to 15) are extremely important parameters which have great influence on the quality of the final vectors. Moreover, using a lower threshold for word frequencies (cut-off) is a common practice and can significantly improve the results. For these parameters and many others, we relied on typical values encountered in the published literature, as full exploration is intractable.

In the next chapter we present our second helping contribution. We propose a smoothing technique which supports fractional counts. The fractional counts arise when we weight data n-grams, sentences, or phrase pairs in Chapter 7.

4

Parametric Smoothing of Probability Models

Language models and translation models are two essential components in any phrase-based statistical machine translation system (PB-SMT). The first approximates the likelihood of a word given a history of words, and the second gives the probability of a target phrase given a source phrase. Their estimation is carried out from count statistics collected from data observed in a corpus. However, the maximum likelihood estimation will assign a 0 probability to any unobserved items. This 0 probability problem is usually remedied using a smoothing technique.

Basically, the smoothing of a probability distribution refers to how this distribution can assign reasonable probability values to all possible outcomes including those never encountered in the training corpus. The common approach to deal with this issue is to reserve a small probability mass for the unseen events. A smoothing technique will attempt to find a good estimate for this mass. Accordingly, the performance of any technique will be strongly decided by its ability to optimize this mass without hurting that attributed to the observed events. The literature is extremely rich in this matter, making the creation of a complete inventory of smoothing methods almost impossible. In practice, however, few methods have known consistent usage in different tasks. Nevertheless, most of these methods are built on the core assumption that the counts are of integral nature, and one can very rarely come across exceptions which deal with non integral counts. We will see soon, in Chapter 7, that this assumption does not hold when we assign weights based on how clean a data item is. The purpose of this chapter is to introduce a new smoothing technique which can deal with this shortcoming.

4.1 Traditional Smoothing Techniques

The excellent comparative study by Chen and Goodman (1999) has been, for long time, the reference survey for the most popular smoothing methods. In this study, the authors describe each of the methods very nicely and in a concise manner and give pointers to the original publications where they first appeared. While Chen and Goodman (1999) discuss the smoothing exclusively in the context of language modeling, the approaches are equally applicable to any count-based estimation. For instance, Foster et al. (2006a) applies some of these techniques to translation model estimation. Among the many smoothing approaches, two widely used are of great interest to our work: Kneser-Ney smoothing and Witten-Bell smoothing. We will limit our review of the traditional techniques to these two, in the following.

4.1.1 Kneser-Ney Smoothing (KN)

KN smoothing has known widespread use because it has consistently shown to outperform other methods. The original method proposed by Ney et al. (1994) reserves unseen mass through absolute discounting. Every n-gram count is deducted by a well-optimized constant parameter $0 < D < 1$.

Chen and Goodman (1999) suggest a modified version of KN smoothing where they use three constants D_1, D_2, D_{3+} instead of the only parameter D in the original technique. D_i is used to discount the count i, and D_{3+} is the discount constant for any count ≥ 3. The gained mass resulting from discounting is reassigned using a lower-order smoothed distribution, which makes this technique an interpolated model. Sticking to the notation used by Chen and Goodman (1999), the probability of a word w_i after a history w_{i-n+1}^{i-1} can be expressed by:[1]

$$\Pr_{KN}\left(w_i \mid w_{i-n+1}^{i-1}\right) = \frac{\max\left\{\mathrm{c}\left(w_{i-n+1}^{i}\right) - \mathrm{D}\left(\mathrm{c}\left(w_{i-n+1}^{i}\right)\right), 0\right\}}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)} + \gamma \Pr_{KN}\left(w_i \mid w_{i-n+2}^{i-1}\right) \tag{4.1}$$

[1]The notation w_x^y means the succession of words, in a given sentence, staring by the word at the position x and ending at position y.

where $\gamma = \frac{\sum_{w_i} \mathrm{D}\left(\mathrm{c}\left(w_{i-n+1}^{i}\right)\right)}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)}$ is the gained mass. The right-most term $\Pr_{KN}\left(w_i \mid w_{i-n+2}^{i-1}\right)$ represents the lower-order smoothing distribution for the $(n-1)$-gram obtained by dropping the oldest word in the history. $\mathrm{D}\left(\mathrm{c}\left(w_{i-n+1}^{i}\right)\right)$ is the discounting constant which depends on the n-gram count, as mentioned above.

Following Ney et al. (1994), Chen and Goodman (1999) derive closed-form expressions for the discounting constants D_i from the corpus. Their estimation is based on the count of counts (n_r, number of n-grams appearing r times). For example n_1 is the number of singleton n-grams, also known as “hapax legomena”. The discounting constant D_i is given by:[1]

$$D_i = i - (i+1)\, Y \frac{n_{i+1}}{n_i} \tag{4.2}$$

where $Y = \frac{n_1}{n_1 + 2n_2}$ is an intermediate term added for convenience of notation of D_i for different values of i. While Chen and Goodman (1999) set i to be between 1 and 3, their expression can be extended to an arbitrary number of discounts. This was, indeed, examined by Sundermeyer et al. (2011) and found to slightly improve the performance sometimes.

Kneser and Ney (1995) added an additional constraint to the smoothing term $\Pr_{KN}\left(w_i \mid w_{i-n+2}^{i-1}\right)$, so that its effect is reduced when w_i has large number of higher-order occurrences, but only with few preceding words. To clarify this, a very useful example was given by Chen and Goodman (1999). The probability of the bi-gram “*Mr. Francisco*” (i.e. $\Pr_{KN}$ (*Francisco* | *Mr.*)), may become large as it receives a high contribution from the smoothing distribution (i.e. $\Pr_{KN}$ (*Francisco*))), which in turn is large because of large unigram count of *Francisco*. But probably in our corpus the word *Francisco* appears only after *San*, and therefore should receive a low unigram probability. Consequently, KN smoothing has a different way of counting lower-order n-grams. A lower-order n-gram count is the number of unique words preceding it in the immediate higher order. However, n-grams which start with a “begin-of-sentence” (`BOS`) marker are excluded from this rule and their true counts are kept as no words preceding them can be found. In summary, KN n-gram counting adjusts the corpus counts as follows:

$$\mathrm{c}_{KN}\left(w_i^j\right) = \begin{cases} \mathrm{c}\left(w_i^j\right) & \text{if } j = i+n-1 \text{ or } w_i = \texttt{BOS} \\ \left|\left\{w : \mathrm{c}\left(w w_i^j\right) > 0\right\}\right| & \text{otherwise} \end{cases} \tag{4.3}$$

[1]Here, we set $D_{3+} = D_3$

This means that the highest order n-grams and those starting with a `BOS` will keep their corpus counts. Other n-gram counts will be set to the number of different types preceding them in the immediate higher order. From now on, whenever we refer to KN smoothing we mean the modified version by Chen and Goodman (1999).

4.1.2 Witten-Bell smoothing (WB)

In practice, we generally fallback to WB smoothing when KN fails to execute in certain situations. An example of such situations is when the corpus is made out of the word classes. In general, this violates some of the assumptions made by the KN smoothing. More specifically, some lower counts do not obey the descending order (i.e. $n_1 \gg 2n_2 \gg 3n_3 \ldots$) under which circumstances discounting constants can not be calculated.

WB smoothing was originally proposed for adaptive text compression. Witten and Bell (1991) propose different methods to predict the probability that the next symbol in a message stream will be novel. In language modeling context, WB smoothing refers to Method (C) of these proposed methods. The intuition behind this method is that the reserved mass should correlate with the proportion of novel incoming symbols. A good approximation of this proportion is the rate of novelty in the already-received symbols. If n symbols were received in total, the number of novel symbols will be the unique (i.e. the types) symbols received u. Therefore, the estimated novelty rate is $\frac{u}{n+u}$, which is also taken to be the reserved mass by WB smoothing.

WB is an interpolated model. The probability is expressed as follows:

$$\Pr_{WB}\left(w_i \mid w_{i-n+1}^{i-1}\right) = \lambda \frac{\mathrm{c}\left(w_{i-n+1}^{i}\right)}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)} + (1-\lambda) \Pr_{WB}\left(w_i \mid w_{i-n+2}^{i-1}\right) \tag{4.4}$$

where $1-\lambda$ is the reserved mass, which, as stated above, corresponds to the novelty rate and is expressed as follows:

$$1-\lambda = \frac{\left|\left\{w_i : \mathrm{c}\left(w_{i-n+1}^{i}\right) > 0\right\}\right|}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right) + \left|\left\{w_i : \mathrm{c}\left(w_{i-n+1}^{i}\right) > 0\right\}\right|} \tag{4.5}$$

4.2 Parametric Discounting

Even though the work on LM smoothing is extensive, support for fractional counts is very limited. Indeed, almost all smoothing studies are built around the inherent assumption that the n-gram counts are of integral nature. While this looks to be the natural and obvious choice in most situations, fractional counts may emerge in some sensible applications. In general, if the n-gram counts from the corpus have been processed somehow, one is likely to end up with non-integral counts. In particular, we face this scenario when we weight the different n-grams based on how significant their cooccurrences are (cf. Section 7.1.1). Another example is when the n-grams and their counts are generated by a process akin to the one where we use semantic substitutions (in Chapter 6).

In reality, some exceptions exist to our claim that smoothing methods do not support fractional counts. The first is the WB implementation in the SRILM toolkit by Stolcke (2002). However, its support is restricted to allowing real-valued c_i's in the first term of the Equation (4.4), and in the first term of λ's denominator. This simple solution does not adapt well to the value range of c_i's. More precisely, the reserved mass may become totally dominated by the number of unique types and therefore, will give a very large weight to the interpolating term when the c_i's are very small. In addition, the toolkit does not allow this for KN smoothing, which is better than WB whenever applicable.

Support for fractional counts in KN smoothing was first considered, independently, by Tam and Schultz (2008) and Bisani and Ney (2008). They propose very similar approaches. Both rely on a single discounting constant and discard any n-grams with counts less than this constant (i.e. attributing them a count of value 0). In the first work, however, this constant was set by hand to a suitable value, while in the second it was optimized on a held-out set so that an error measure, specific to the task, is minimized. Sadly, these two approaches are fine-tuned to specific tasks and hence suffer from the lack of general applicability. In fact, they are not very different from the integral count case, except the aforementioned rejection behavior. Such behavior might be, in itself, undesirable in the general language modeling context. As a matter of fact, the main purpose of our n-gram weighting described in Section 7.1.1 is to not discard any evidence from training data, but rather use every bit of it with various levels of reliability.

A more general approach is Expected KN (EKN), suggested by Zhang and Chiang (2014). In this study, the occurrences of a given n-gram are weighted throughout the corpus (i.e the corpus is weighted sentence-wise). It is straightforward to transform such weights into a probability distribution. This latter is used to compute expected values of an n-gram occurrence and afterwards expected values of the counts of counts relying on a Poisson-binomial mixture distribution. The smoothing, then, is performed in the same way as for integral counts except the fact that all counts and counts of counts are replaced by their expected values. In their experiments, Zhang and Chiang (2014) show that this approach outperforms all other methods supporting fractional counts by a substantial margin. They tested it in two different tasks. In data selection, each sentence in the corpus receives a weight proportional to its closeness to the in-domain set. Interestingly, by doing so , they convert the data selection into a weighting, and consequently no data was thrown away. On the other hand, the alignment task is weighted by design, and the pseudo-counts collected in the E-step play the role of expected counts for the smoothing. This latter is carried out during the M-step.

Even though the EKN approach solves the problems faced by the other methods, it is not suitable when the weighting is computed n-gram-wise. Obviously, the algorithm needs the different weights attributed, sentence-wise, to a given n-gram so that it can estimate its contribution to the expected counts of counts. Armed with the n-gram-wise weights only (as in our weighting), it is hard to "undo" the counting and infer sentence-wise weights. Of course, one can attribute weights to a sentence proportional to the weights of its n-grams, but this is somewhat costly.

As a remedy, we propose a simple discounting method which also supports fractional counts. For this purpose, we use a continuous function, which we name ξ, whose parameters are chosen so that the likelihood of the data is maximized. However, as will be detailed later, in the case of weighted counts, our method assumes that the proportion of the unseen events is constant. While nothing prevents this function from having multiple parameters, for simplicity and ease of computation, we limit our study to the single-parameter case. We summarize the benefits of our approach in the following points:

- Depending on the arguments passed to ξ, our discounting can be interpreted as a generalization of KN or WB discounting.

- Unlike EKN, our method, in its general form, does not necessarily reduce to the classical KN or WB discounting when applied to integral counts. Instead, in most cases it improves over them.

- Compared to KN and WB smoothing, novelties of our approach include the use a continuous discounting scheme and explicitly addressing the problem of estimating the number of unseen events.

- Compared to classical WB smoothing, additional novelties in our approach are the problem formulation and parameter tuning. Indeed, we formulate the problem in a more generalized and parmeterized way. Additionally, the parameters are tuned to maximize the data likelihood similar to KN smoothing.

4.2.0.1 The Discounting Function

Very similar to WB equation (4.4), our probability is written as follows:

$$\Pr_\xi\left(w_i \mid w_{i-n+1}^{i-1}\right) = \alpha \frac{\mathrm{c}\left(w_{i-n+1}^{i}\right)}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)} + (1-\beta)\Pr_\xi\left(w_i \mid w_{i-n+2}^{i-1}\right) \tag{4.6}$$

where $\alpha = \xi_\theta\left(x_{w_{i-n+1}^{i}}\right)$, and $\beta = \frac{\sum_{w_i} \xi_\theta\left(x_{w_{i-n+1}^{i}}\right) \mathrm{c}\left(w_{i-n+1}^{i}\right)}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)}$. There is a significant difference though, the argument of the function ξ_θ depends on the whole n-gram, unlike in Equation (4.4) where it exclusively depends on the context. We denoted this argument by $x_{w_{i-n+1}^{i}}$ and we will soon see how it is defined in terms of counts and how our discounting can be KN-based or WB-based accordingly. The discounter ξ_θ is defined in function of the parameter(s) θ and hence the subscript θ in Equation (4.6). From now on, we omit the subscript θ for convenience.

Our discounting function, ξ, is a real increasing function having values between 0 and 1:

$$\begin{aligned} \xi \colon \mathbb{R}_+ &\to [0,1] \\ x &\mapsto \xi(x) \end{aligned}$$

As special cases, with the following choices of the discounter, the classical discounting methods are restored:

- By taking $\xi\left(x_{w_{i-n+1}^{i}}\right) = 1 - \frac{\mathrm{D}\left(\mathrm{c}\left(w_{i-n+1}^{i}\right)\right)}{\mathrm{c}\left(w_{i-n+1}^{i}\right)}$, it becomes equivalent to KN smoothing.

- By taking $\xi\left(x_{w_{i-n+1}^{i}}\right) = \frac{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)}{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right) + \left|\left\{w_i : \mathrm{c}\left(w_{i-n+1}^{i}\right) > 0\right\}\right|}$, we obtain the WB smoothing.

Name	Definition
Hyperbolic (HYP)	$\xi(x) = \frac{1}{1+\theta/x}$
Power (POW)	$\xi(x) = (1 + 1/x)^{-\theta}$
Exponential (EXP)	$\xi(x) = (1+\theta)^{-1/x}$

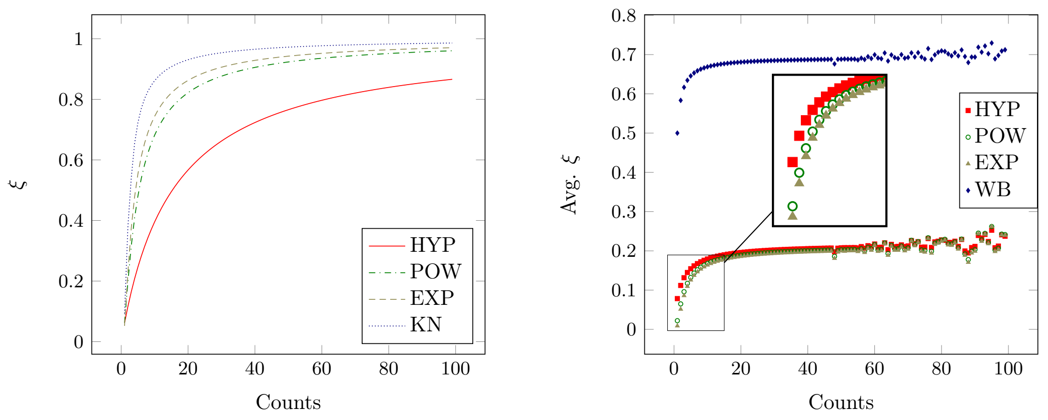
Table 4.1: Different forms of our discounter ξ

Certainly, there is an infinite number of functions which fit our requirements above. However, we select three simple forms having three levels of speed. i.e. how fast the discounter increases towards 1. The slowest one is a hyperbolic function, then comes a power function, and finally an exponential function. The definition of each of these function forms is given Table 4.1. Of course in all cases, the parameter is assumed to be $\theta > 0$.

Now, if the argument of ξ depends solely on the count (i.e. $x_{w_{i-n+1}^{i}} = \mathrm{c}\left(w_{i-n+1}^{i}\right)$) and all n-grams with the same number of occurrences are discounted the same way, then we say that ξ is KN-based. Such discounter is virtually similar to having a very large number of discounting constants D_i in KN smoothing. As this kind of ξ can be interpreted as a generalization of KN smoothing, we adjust the counts of the lower orders using the formula (4.3), as we usually do in KN smoothing.

On the other hand, if the argument depends on the history and all n-grams with the same history are discounted the same way, then our ξ becomes WB-based. We define the argument in this case to be the ratio of the total occurrences of the history to the number of its unique instances. i.e. $x_{w_{i-n+1}^{i}} = \frac{\sum_{w_i} \mathrm{c}\left(w_{i-n+1}^{i}\right)}{\left|\left\{w_i : \mathrm{c}\left(w_{i-n+1}^{i}\right) > 0\right\}\right|}$. This choice rises from the intuition that the factor λ in Equation (4.5) can be rearranged to look like our Hyperbolic ξ with $\theta = 1$. Then we generalize the expression of this special ξ by allowing any type of discounter with any value for θ, while keeping the argument as the aforementioned ratio.

Note the correspondence between WB-based and KN-based discountings for the lowest counts. In WB-based discounting, the histories whose number of unique occurrences is the same as their total number of occurrences (i.e. $x_{w_{i-n+1}^{i}} = 1$) will have the lowest value for the ξ argument and, therefore, will be discounted the most. Since n-grams



(a) KN-based ξ **(b)** WB-based ξ

Figure 4.1: Comparison of different discounts for KN-based and WB-based ξ

KN: original Kneser-Ney; WB: original Witten-Bell; HYP: hyperbolic ξ; POW: Power ξ; EXP: Exponential ξ;

with such histories will have a count of 1, they will be also the ones which will be discounted the most by KN-based ξ. This can be seen in Figure 4.1, where a KN-based and a WB-based ξ sub-plots for the first 100 counts are shown side by side.

Figure 4.1(a) shows, in addition, the difference of speed between different discounting forms. Interestingly, the original Kneser-Ney smoothing is the fastest, as it relies, by design, almost exclusively on the first three counts to discount all the needed mass. On the other hand, different ξ forms distribute this mass over all counts. However, as expected, the exponential ξ is the fastest and the hyperbolic is slowest and the power is always closer to the exponential than to the hyperbolic.

A more interesting behavior can be seen in Figure 4.1(b). This figure plots the average ξ value for each count, since Witten-Bell discounting is not performed by count, and therefore a count may end up having more than one associated ξ values (except 1 of course). Here, it becomes apparent how conservative the original Witten-Bell smoothing is, since it has no parameter which can be tuned to the dataset.

4.2.0.2 Parameter Estimation

Following the line of reasoning adopted by Ney et al. (1994), we first ignore the identities of the different n-grams and look only at their occurrences. Second, we re-write the probability (4.6) in its back-off form. This way, the interpolating distribution is ignored during optimization and thus the estimation becomes much simpler. By simplifying the notation of $\xi_\theta\left(x_{w^i_{i-n+1}}\right)$ to ξ_i and $\mathrm{c}\left(w^i_{i-n+1}\right)$ to c_i, the back-off version of Equation (4.6) can be written as follows:

$$\Pr_\xi\left(w_i \mid w^{i-1}_{i-n+1}\right) = \begin{cases} \frac{c_i\xi_i}{\sum_j c_j} & \text{if } c_i > 0 \\ \left(1 - \frac{\sum_i c_i\xi_i}{\sum_j c_j}\right)\Pr_\xi\left(w_i \mid w^{i-1}_{i-n+2}\right) & \text{otherwise} \end{cases} \tag{4.7}$$

Now, let's assume an estimate of the unseen events exists, call it n_0. Then, the log-likelihood function of the data, parameterized by θ, can be expressed as follows:

$$\mathscr{L}\left(\theta\right) = \sum_i \log\left(\frac{c_i\xi_i}{\sum_j c_j}\right) + n_0 \log\left(\left(1 - \frac{\sum_i c_i\xi_i}{\sum_j c_j}\right)\Pr_\xi\left(w_i \mid w^{i-1}_{i-n+2}\right)\right) \tag{4.8}$$

Simplifying and deleting the constant terms in θ and combining terms with identical counts, the previous log-likelihood function has the same maximizer as:

$$\mathscr{L}'\left(\theta\right) = \sum_i c_i \log\left(\xi_i\right) + n_0 \log\left(\sum_i c_i\left(1 - \xi_i\right)\right) \tag{4.9}$$

where c_i is the count value corresponding to the n-gram i in the count data. It is hard to find closed-form solutions for this equation when ξ is KN-based because of the summation inside the log. Even though with serious approximations and the help of Jensen's inequality one might be able to find a closed-form approximation of a lower-bound for this expression, we found that, in practice, an iterative procedure, such as Powell search (originally proposed by Powell (1964)), runs very efficiently on this optimization.

On the other hand, for WB-based ξ we can easily derive closed-form solutions for each of the ξ forms. In this case and since we are ignoring the identity of the n-grams, there will be one single ξ value for all data points, say ξ_0. The log-likelihood (4.9) is maximized for the same value of θ as:

$$\mathscr{L}_{WB}\left(\theta\right) = \left(\sum_i c_i\right)\log\left(\xi_0\right) + n_0 \log\left(1 - \xi_0\right) \tag{4.10}$$

Deriving this expression for ξ_0, we get:

$$\frac{\partial\mathscr{L}_{WB}\left(\theta\right)}{\partial\xi_0} = \frac{\sum_i c_i}{\xi_0} - \frac{n_0}{1 - \xi_0}$$

Then setting this derivative to 0, gives:

$$\xi_0 = \frac{\sum_i c_i}{n_0 + \sum_i c_i}$$

ξ	$\hat{\theta}$
HYP	$\frac{n_0}{N}$
POW	$\frac{\log\left(1+\frac{n_0}{\sum_i c_i}\right)}{\log\left(1+\frac{\sum_i c_i}{N}\right)}$
EXP	$\left(1+\frac{n_0}{\sum_i c_i}\right)^{\frac{\sum_i c_i}{N}}-1$

Table 4.2: Different ξ parameter estimates for WB-based ξ

n_0: estimated number of unseen events; N: number of types; c_i: number of occurrences of n-gram i

Note the similarity of this expression with that of $1 - \lambda$ in Equation (4.5). This is an identical expression if one takes into account that WB smoothing takes the unique instances as an estimate to the unseen. This ξ_0 value will, then, be used to compute the corresponding ξ parameter, which will be used to discount the different n-grams using their proper argument values. the ξ parameter is calculated by simply equating the corresponding ξ form with ξ_0 where the argument in ξ is replaced by its value for the whole data set. i.e. $x = \frac{\sum_i c_i}{N}$, where N is the total number of n-grams. Table 4.2 gives the optimal values of the parameter for different ξ forms.

4.2.0.3 The Unseen Events

Now we turn to the constant n_0 which we used in the previous section. This is an approximate expectation of the number of the unseen events. In the context of language modeling, all studies which do not use a held-out set estimate this value through leaving-one-out (LOO). This is a clever solution originally used by Ney et al. (1994) and then followed by subsequent studies on KN smoothing such as Chen and Goodman (1999) and Zhang and Chiang (2014). The idea is to remove one instance of each n-gram. As a consequence, only $i - 1$ instances of an n-gram appearing i times, will be observed and therefore n_0 will be exactly the number of n-grams appearing only once.

Estimating n_0 using LOO is just one way out of many others. This problem has been extensively studied in ecological sciences where it is known as the “number of unseen species”. A large number of estimators has been proposed through the history of this field. A somewhat outdated but still very useful review of different estimators is

Name	n_0	Reference
Leave-One-Out (LOO)	n_1	Ney et al. (1994)
Good-Turing (GT)	$\left(\frac{1}{n_1} - \frac{1}{N}\right)^{-1}$	Good (1953)
Chao (CHAO)	$\frac{n_1^2}{2n_2}$	Chao (1984)
Poisson-Lindley (PL)	$\frac{{}^{8n_2^2}/_3 + {}^{n_1^2}/_3 + n_1 n_2 - 3n_1 n_3}{n_3 + {}^{n_2}/_3}$	See bellow

Table 4.3: List of unseen estimators used in this work

n_i $(i = 1, 2, \ldots)$: number of individuals seen i times in the sample.
N: number of unique instances of individuals in the sample (i.e. number of types).

presented by Bunge and Fitzpatrick (1993). A relatively more recent review is given by Gandolfi and Sastri (2004), but only for a sub-class of estimators. The large number of proposed estimators can be justified by the differences between the underlying populations. While some assumptions may hold for a given population, they can turn to be very bad for another.

Estimators are mainly classified as parametric or non-parametric. The first category of estimators provide the estimate as a function of the counts of counts, usually limited to the very few largest ones (i.e. n_1, $n_2, \ldots$). By contrast, the parametric estimators assume a probabilistic model for the counts of counts. Given a sample, the model parameters are determined accordingly and an expected number of the unseen events can then be calculated. Examples of parametric models which are an excellent fit to our problem include the Zipf-Mandelbrot model introduced by Evert (2004a). We find such model to be complex and computationally expensive for our purpose. The complexity arises from the need to fit the model where the distribution itself involves the evaluation of incomplete gamma functions. For this reason, we restrict our study to a few non-parametric estimators. Table 4.3 summarizes the estimators we experiment with in this work. However, following the way of inference commonly adopted by researchers while deriving parametric estimators, we derive a new simple estimator, which performs well with language data in practice.

A special matter which has to be noted with LOO (the first estimator in Table 4.3) is the adjustment in the n-gram counts implied by the leaving one out. Indeed, the number of unique n-grams becomes $N' = N - n_1$ and the sample size becomes $\sum_i {c_i}' = \sum_i c_i - N$. A further issue with the GT estimator (the second estimator in Table 4.3) is that this formula slightly differs from the common one in the literature. More precisely, the GT estimate of n_0 relies on the proportion of the unseen events due to Good (1953) (i.e. $\frac{n_1}{n}$). The difference lies in the denominator of this proportion (i.e. n) which is commonly interpreted as the sample size (i.e. $\sum_i c_i$). We found that this interpretation results in severe underestimation of the unseen and consequently produces very small values of the reserved mass. Interpreting it as the unique instances, on the other hand, gives much better estimates and therefore delivers a substantially higher performance. The formula of GT estimator in Table 4.3 comes from this latter interpretation.

As for the PL estimator (last estimator in Table 4.3), it was derived following the inference commonly adopted by the parametric models, in an attempt to explore the robustness of this family of estimators. The accepted way to derive the parametric models starts by choosing a distribution for the counts of counts and then mix it with the Poisson distribution. In his introduction, Evert (2004a) gives mathematical grounding for this practice. We pick the Lindley distribution, originally proposed by Lindley (1958), as our mixing distribution. Interestingly, its mixed-Poisson distribution expression is very simple and has only one parameter. However, in order to escape the expensive fitting, we derive closed-form expression using the information in the first three counts of counts. This process produces the last estimator in Table 4.3.

The probability density function (PDF) of the Lindley distribution is given by:

$$f_\theta\left(x\right) = \frac{\theta^2}{\theta + 1}\left(1 + x\right)\exp\left(-\theta x\right); x > 0; \theta > 0$$

where θ is the distribution parameter. Mixing this distribution with the Poisson distribution gives the following probability mass function (PMF):

$$\mathrm{PL}_\theta\left(n\right) = \frac{\theta^2}{\left(\theta + 1\right)^3}\frac{n + \theta + 2}{\left(\theta + 1\right)^n}; n = 0, 1, 2, \ldots; \theta > 0$$

Ghitany and Al-Awadhi (2001) provide recursive evaluation formula for this PMF, among others:

$$\left(\theta + 1\right)\mathrm{PL}_\theta\left(n\right) = \left(1 - \frac{\theta}{n}\right)\mathrm{PL}_\theta\left(n - 1\right) + \frac{1}{n}\mathrm{PL}_\theta\left(n - 2\right) \tag{4.11}$$

Using this recursive formula for $n = 2$ and $n = 3$ and replacing the probabilities by the expected values we get:

$$\begin{cases} (\theta + 1)\, n_2 = \left(1 - \frac{\theta}{2}\right) n_1 + \frac{1}{2} n_0 \\ (\theta + 1)\, n_3 = \left(1 - \frac{\theta}{3}\right) n_2 + \frac{1}{3} n_1 \end{cases} \tag{4.12}$$

Eliminating θ using the second equation, replacing it in the first, and then solving for n_0 gives the PL estimator:

$$n_0 = \frac{\frac{8}{3} n_2^2 + \frac{1}{3} n_1^2 + n_1 n_2 - 3 n_1 n_3}{n_3 + \frac{1}{3} n_2}$$

4.2.0.4 Discounting Fractional Counts

The fractional counts we are considering in this work are obtained through n-gram weighting. Each n-gram is associated with a weight strictly positive and which does not exceed 1. This weight should reflect how noisy this n-gram is thought to be. The lower the weight is the noisier we think the n-gram is. The weights are used to penalize the true counts and hence the fractional counts arise.

Basically, computing probabilities from weighted counts is indistinguishable from the conventional procedure. However, two components are adapted to suite this case. First, the number of unseen events, n_0, is re-estimated. Second, in the case of WB-based discounting, in addition to penalizing the counts, the number of unique n-gram instances is also penalized.

As explained in Section 4.2.0.2, an estimate of the number of unseen events n_0 is necessary for determining the discounter parameters θ. Nevertheless, all the estimators of the number of unseen events presented in Section 4.2.0.3 are based on the largest counts of counts. These counts of counts become difficult to compute when the counts are weighted, as the discreteness characteristic of these latter is lost with the weighting. Moreover, the weighting may also overload the number of the smallest count (i.e. n_1) if the counts are binned. For these reasons, we define the proportion of the unseen events p_0, which we take along while moving from unweighted to weighted:

$$p_0 = \frac{n_0}{n_0 + \sum_i c_i}$$

We regard this proportion as a characteristic of the data which can be reliably estimated from the ensemble of unweighted counts. Moreover, p_0 is also a way to adapt the

Estimator	Unigrams	2-grams	3-grams	4-grams
LOO	0.0512	0.2819	0.6463	0.7998
GT	0.1053	0.4878	0.8133	0.9053
CHAO	0.1002	0.4495	0.7762	0.8797
PL	0.1044	0.4884	0.8244	0.9148

Table 4.4: Examples of p_0 values for different estimators at different n-gram orders

number of unseen events n_0 relative to the reductions in the counts $\sum_i c_i$ caused by the weighting. Finally, this proportion is used to recompute a new unseen estimator n_0' from the weighted counts $\sum_i {c_i}'$.

$$n_0' = \frac{p_0}{1-p_0}\sum_i {c_i}'$$

Table 4.4 gives examples of typical values that p_0 takes. LOO gives the lowest values because it relies on all data points to compute its estimate, while the other estimators use at most the three first counts of counts. The values computed by these other estimators are, nonetheless, comparable. The fact that p_0 grows with higher orders would be an expected behavior. It is a consequence of the combinatorial explosion at higher orders. If we consider the data from which Table 4.4 was created, the ratio of observed 3-grams to all possible combinations (i.e. $|V|^3$; where V is the corpus vocabulary) is 6.18×10^{-10}; Whereas for 4-grams this proportion becomes 3.85×10^{-15}.

In addition to adapting n_0 to the weighted case, in our WB-based discounting, the number of unique occurrences of an n-gram have to be also adapted. This number represents the denominator of the argument of our discounter ξ. We adapt this number by simply summing up the weights over all instances of an n-gram. Intuitively, this is equivalent to applying the weights to each unique occurrence, which is valued as 1 regardless of the corresponding count. In fact, the unique counts can be written as follows:

$$\left|\left\{w_i : \mathrm{c}\left(w_{i-n+1}^i\right) > 0\right\}\right| = \sum_{w_i:\mathrm{c}\left(w_{i-n+1}^i\right)>0} 1$$

and then by penalizing each instance, it becomes:

$$\left|\left\{w_i : \mathrm{c}\left(w_{i-n+1}^i\right) > 0\right\}\right|' = \sum_{w_i:\mathrm{c}\left(w_{i-n+1}^i\right)>0} \delta_{w_{i-n+1}^i}$$

where $\delta_{w_{i-n+1}^i}$ is the weight associated with the n-gram w_{i-n+1}^i.

4.2.0.5 A Note about Complexity

Our smoothing adds negligible overhead to the time and memory complexities. Both complexities are linear in the number of distinct counts, which is usually very small. In the case of KN-based discounting, these counts have to be stored together with their counts and traversed at each evaluation of the log-likelihood function. The situation is even simpler for WB-based discounting as closed-form estimators can be used. In this latter case, one pass over the data points to compute the sum of their counts is enough.

4.3 Evaluations

While the main purpose of the proposed smoothing techniques is to support fractional counts, we think evaluating them in the normal conditions is necessary to make sure they are comparable to the traditional techniques. In this section, we examine this concern. We conduct our experiments on French and English corpora from different genres and different sizes. We present an intrinsic evaluation showing the perplexity of the testsets evaluated with different models trained on the aforementioned corpora. Some statistics summarizing the number of sentences and words in the datasets are given in Table 4.5. We use three types of corpora:

- Large corpus, which consists of around a million random sentences from Giga and Common Crawl corpora (named GIGA in the table).

	English		French	
Dataset	Sentences ($\times 10^3$)	Tokens($\times 10^6$)	Sentences($\times 10^3$)	Tokens($\times 10^6$)
WMT-Test	6.00	0.14	6.00	0.15
WMT-Dev	3.00	0.07	3.00	0.08
IWSLT-Test	2.33	0.05	2.33	0.05
IWSLT-Dev	0.90	0.02	0.90	0.02
TED	219.40	4.45	219.40	4.75
EPPS	494.03	13.62	494.03	15.00
GIGA	977.09	23.49	984.05	27.03

Table 4.5: Statistics of corpora and testsets used to evaluate the language model smoothing

		English		French	
Unseen	Discounter	WMT-Dev	IWSLT-Dev	WMT-Dev	IWSLT-Dev
LOO	KN	340.699	229.495	215.495	399.31
	Hyp	334.808	224.029	210.226	395.029
	Pow	342.888	229.323	215.009	403.017
	Exp	344.563	230.603	216.105	405.456
Chao	Hyp	328.823	220.696	207.503	388.748
	Pow	337.211	226.22	212.382	396.874
	Exp	342.579	229.71	215.413	402.829
Good-Turing	Hyp	322.004	216.736	204.168	381.64
	Pow	330.121	222.014	208.635	389.58
	Exp	335.175	225.274	211.408	395.204
Lind	Hyp	322.454	216.891	204.099	382.991
	Pow	330.324	221.989	208.533	390.581
	Exp	335.298	225.193	211.28	396.147

Table 4.6: Perplexities of Devsets using models computed using different Kneser-Ney-based smoothing techniques on the EPPS-Corpus

- Unseen estimators: LOO (leave-one-out); Chao; Good-Turing; Lind (nonparametric-Lindley)
- Discounters: Hyp (hyperbolic); Pow (power); Exp (exponential)

- Moderate corpus, which consists of half a million random sentences from EPPS and News Commentary corpora (named EPPS in the table).
- Small corpus, which refers to a TED corpus of around quarter a million sentences (named TED in the table).

We test on two types of testsets: News2013 and News2014 (named WMT-Test); and IWSLT2014 and IWSLT2013 (named IWSLT-Test). We use News2012 (WMT-Dev) and IWSLT2010 (IWSLT-Dev) as development sets to select the best combination of the unseen estimator and the discounter function.[1]

In order to give an impression about the effect of different KN-based smoothing techniques, we show the perplexities of their corresponding models trained on the middle corpus (i.e. EPPS) and evaluated on the Devsets, in Table 4.6. It happened by

[1]Testsets and Devsets starting with "News" are published during the WMT Evaluation Campaigns, while those starting with "IWSLT" are published during the IWSLT Evaluation Campaign.

		English		French	
Corpus	Config.	WMT-Test	IWSLT-Test	WMT-Test	IWSLT-Test
TED	KN	339.810	113.894	390.412	80.847
		3.5%	1.4%	2.7%	0.8%
	Best Dev.	328.068	112.290	379.791	80.194
EPPS	KN	312.494	231.393	380.479	146.985
		5.2%	5.7%	3.8%	4.3%
	Best Dev.	296.329	218.284	365.952	140.664
GIGA	KN	205.652	153.385	272.133	128.062
		1.7%	1.2%	1.9%	0.7%
	Best Dev.	202.110	151.600	266.989	127.178

Table 4.7: Perplexities of the models computed with different Kneser-Ney-based smoothing techniques on different corpora, evaluated on the testsets and trained using the best performing unseen estimator and discounter on the corresponding DevSet

The small boxes between the rows show the relative improvement over the KN model

chance that the EPPS model is the one showing the largest improvements over the KN model. The perplexities of the models trained on the other two corpora are recorded in Appendix D.1. Afterwards, these perplexities evaluated on the Devsets are our means to select a combination of the unseen estimator and the discounter to be applied on the Testsets. The results are gathered in Table 4.7. Similarly, the results corresponding to Witten-Bell-based smoothing are collected in Tables 4.8 and 4.9. The percentages in the small boxes between the rows in Tables 4.7 and 4.9 give better view of the improvement over the baseline. Therein the reduction in perplexity is expressed as a fraction of the baseline score. Indeed, we are always able to enhance the model's performance, although by very small margin in some cases. Moreover, it is interesting to note that almost in all cases, the proposed smoothing is working better for English than for French.

It is not surprising to see how hard it is to beat KN smoothing, through Tables 4.6 and 4.7. This is especially true when the model records low perplexity for a given testset. The gains range from less than a perplexity point to slightly more than 10 points, depending on the corpus, language, and devset. The largest improvements appear to happen for the EPPS model, with a reduction of $3 \sim 5\%$ in the perplexity as compared to the KN model.

Although it is clear that the Hyperbolic discounter combined with Good-Turing estimator are the winning pair in Table 4.6, we can not see a consistent combination of an unseen estimator and a discounter across corpora, languages and Devsets (see Appendix D.1 for the other results). Therefore, a devset is always needed to achieve

		English		French	
Unseen	Discounter	WMT-Dev	IWSLT-Dev	WMT-Dev	IWSLT-Dev
	WB	578.641	369.368	347.51	606.804
LOO	Hyp	416.282	267.035	255.877	469.374
	Pow	404.199	260.326	248.556	458.701
	Exp	397.643	257.029	244.586	452.966
Chao	Hyp	428.378	274.185	262.672	481.785
	Pow	425.375	273.104	260.644	478.413
	Exp	420.145	270.846	257.411	473.51
Good-Turing	Hyp	416.282	267.035	255.877	469.374
	Pow	413.446	266.073	253.97	466.304
	Exp	409.49	264.484	251.53	462.852
Lind	Hyp	416.565	266.975	255.643	470.628
	Pow	413.994	266.141	253.877	467.751
	Exp	410.068	264.564	251.501	464.318

Table 4.8: Perplexities of devsets using models computed by different Witten-Bell-based smoothing techniques on the EPPS-Corpus

- Unseen estimators: LOO (leave-one-out); Chao; Good-Turing; Lind (nonparametric-Lindley)
- Discounters: Hyp (hyperbolic); Pow (power); Exp (exponential)

the best performance. Nevertheless, the results, also, suggest to use the (Hyperbolic, Nonparametric-Lindley) or (Power, Nonparametric-Lindley) combinations, when no devset can be used. These two combinations are slightly close to the best combination and never degrade the performance considerably, across all corpora, languages, and devsets.

On the other side, the proposed techniques show a much better behavior with the WB-based smoothing. In fact, all the different combinations of unseen estimators and discounters outperform the original WB model by a large margin. The perplexity is reduced by more than 100 points and in the worst case by around 60 points. This is equivalent to a relative reduction of 20 $\sim$ 30% of the baseline perplexity. Like for KN-based smoothing, a devset is needed for optimized performance. However, the Leave-One-Out estimator combined with the Exponential discounter looks to be the best combination, if the tuning has to be avoided.

Similar to Table 4.6, Table 4.8 shows that the Good-Turing and the Nonparametric-Lindley estimators are not very different, which can also be seen in Tables D.1 and D.2 in

		English		French	
Corpus	Config.	WMT-Test	IWSLT-Test	WMT-Test	IWSLT-Test
TED	WB	540.126	160.83	580.853	109.485
		30.8%	23.6%	26.5%	20.5%
	Best Dev.	373.951	122.7971	426.8941	87.0841
EPPS	WB	524.919	374.941	586.32	224.499
		30.6%	30.2%	25.6%	27.2%
	Best Dev.	364.215	261.726	436.325	163.421
GIGA	WB	290.66	209.776	369.381	178.098
		25.9%	23.4%	23.9%	24.1%
	Best Dev.	215.328	160.66	281.195	135.245

Table 4.9: Perplexities of the models computed with different Witten-Bell-based smoothing techniques on different corpora, evaluated on the testsets and trained using the best performing unseen estimator and discounter on the corresponding Devset

The small boxes between the rows show the relative improvement over the WB model

the appendices. However, these two estimators are not as successful with the WB-based as they are with the KN-based models. With Witten-Bell smoothing, the Leave-One-Out estimator is more favorable. Another remarkable difference, which stands clear from the tables, is that WB-based smoothing prefers the faster exponential discounter, while KN-based smoothing adopts the slower hyperbolic or power discounters. We think that these differences are mainly due to the distinct nature of the two techniques. WB discounting values are applied by context while the discounting occurs by count in KN-smoothing. Additionally, the discounting parameters are determined based on the true counts for all orders in WB-smoothing, while these parameters are chosen based on the adjusted counts for the lower orders in KN-smoothing (refer to Sections 4.1.1 and 4.1.2 for more about these smoothing techniques).

It may appear acceptable that the large perplexity values of the WB models are the reason to achieve the substantial improvements in Table 4.9. After all, this behavior has been already noticed in Table 4.7 with the KN results (The larger the perplexity value is, the more important the improvement will be). We think that this is not the right cause here. For instance, the English WB model trained on TED corpus has a perplexity on the IWSLT-Dev less than the perplexity of the KN EPPS model on the same devset, but nevertheless the improvement on the WB model is way larger than that on the KN EPPS model. These results should be rather due to the parameter tuning performed by the proposed techniques. This implies that the proportion of the unseen events assumed by the WB smoothing is not optimal. The original WB smoothing is equivalent to the hyperbolic discounter with an unoptimized parameter set to 1. If we consider the optimized hyperbolic discounter with Leave-One-Out estimator as our baseline, then the improvements become comparable to those in KN table 4.7.

4.4 Conclusions

This chapter was devoted to introducing a new smoothing technique which does not require the data counts to be of integral nature. Such non integral counts arise when we weight our data instances based on how clean they are. The technique consists mainly of two components: a discounter and an unseen estimator. The discounter is a continuous function having values between 0 and 1 and integrates a tunable parameter determined by the maximum likelihood estimation. The unseen estimator gives an estimated value of the unseen events, which usually relies on the few first counts of the data counts. Using different discounter functions and unseen estimators, the technique consistently outperforms the state of the art; in some cases by a large margin.

We showed how the proposed technique can be interpreted as a Kneser-Ney (KN) or Witten-Bell (WB) smoothing, according to the provided argument. In practice, however, the KN models are always chosen over WB models because of the large difference in performance. With the gap between the two techniques substantially reduced by our WB optimized versions, always preferring to train a KN model becomes a non-trivial choice. Indeed, compared to KN training, WB training is computationally easier. First, we dispose of closed-form discounting formulas, which eliminates the need to the iterative search procedure. More importantly, by design WB smoothing does not need the true n-gram counts to be adjusted. While this second advantage avoids a scan of all n-grams, it also implies that WB models trained for a large n-gram order are more appropriate to be used as lower order models than are the KN models.

We also proposed a new unseen estimator which gives competitive estimates for the unseen events. However, it is based on the Lindley distribution which is exponential in nature. It would be more appropriate to use a power law distribution instead as it is widely accepted that such distribution is a better fit for language data. Unfortunately, the power laws imply more complex computations; therefore we avoided them.

It is also noteworthy that in our experiments we chose a single discounter and unseen estimator for all n-gram orders. More improvements maybe possible by tuning the discounter and the unseen estimator per order. The proposed techniques will find applicability in the remaining of this thesis. The next chapter will present our first filtering approach which removes bad parallel sentences. Therein, our parametric smoothing developed in this chapter will meet its first usage, in training high precision bilingual lexicons.

5

Removing parallel noise

Using freely available corpora is a common preference in the machine translation (MT) community. Thanks to the efforts which have been devoted, over the years, to prepare and release them, the research in statistical machine translation (SMT) has matured rapidly.[1] These corpora are available sentence-aligned. Even though the sentence aligners used to align these corpora perform very well, they still tend to commit a non-negligible amount of errors. This is especially noticeable in noisy corpora such as those automatically crawled from the web.

In this chapter, we present the techniques used to detect and remove bad sentence pairs from a sentence-aligned parallel corpus. Our approach is classifier-based; heavily inspired by the work of Munteanu and Marcu (2005) on comparable corpora. We give a detailed description of the approach. We also highlight and motivate the main differences with the aforementioned work. Finally, we demonstrate the utility of the techniques by showing their impact on several translation tasks.

5.1 Introduction

The presence of an important number of wrong parallel pairs in the training data would have a negative impact on the word alignment process. In fact, two main problems can be easily encountered in this situation: wrong word alignments and unaligned words.

[1]Examples of free MT resource repositories:

- `http://opus.lingfil.uu.se/`
- `http://www.statmt.org/europarl/`

" , a déclaré la		kinds , " continued
" , affirme le		not only their
œuvre ,		, both
êtres humains ont		of awareness raising ,

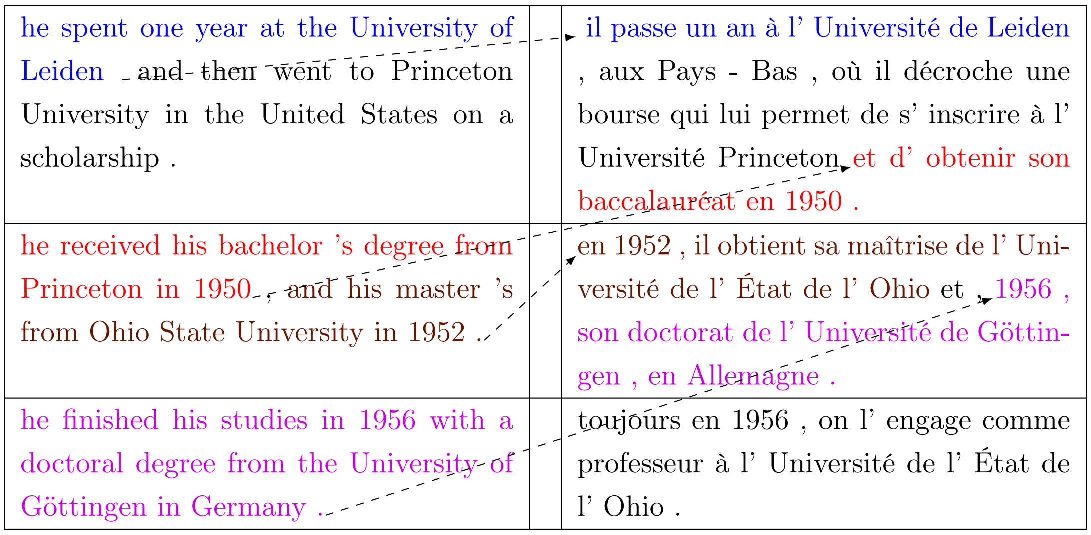
Figure 5.1: Extract from a noisy phrase table

The former, is more likely to happen for the frequent words whereas the latter happens mostly for the less occurring words. Both problems will, in turn, have negative effects on the phrase extraction. Figure 5.1 is an extract from a noisy phrase table which exemplifies the two problems. Aside from the fact that the figure gives a realistic example, it also demonstrates the difficulty of the problem since these phrases appear in the top-ten equivalences for test n-grams.

While the damage caused by the wrong alignments is obvious and affects almost any phrase extraction algorithm, the effect of the unaligned words will depend on the adopted extraction heuristic (see Koehn (2010) for a detailed description of standard phrase extraction algorithms. The problems caused by unaligned words are explicitly addressed in Zhang et al. (2009).) In all situations, the result will not be only to generate inaccurate translation model probability estimates, but also to infiltrate many incorrect phrase pairs into the model. The decoder output will be of low quality, should such pairs get selected.

A wrong training pair can escape a good sentence aligner for many reasons. One possible reason is the assumptions adopted by the sentence aligner for ease of computation or language independence reasons (e.g monotonicity or strongly relying on the length feature.) Another example is due to errors happening earlier in the preprocessing pipeline (e.g. sentence segmentation errors would confuse the aligner since a part of the pair is correct translation.) Figure 5.2 is an instance of the latter case. It shows bad aligned sentences extracted from the English-French Common Crawl corpus. Due to segmentation problems, there is always a part from the previous French sentence which should have been aligned to the next English sentence (matches are illustrated using dashed arrows).

The hope is that such inconveniences will have negligible effect as they get absorbed



he spent one year at the University of Leiden and then went to Princeton University in the United States on a scholarship .		il passe un an à l' Université de Leiden , aux Pays - Bas , où il décroche une bourse qui lui permet de s' inscrire à l' Université Princeton et d' obtenir son baccalauréat en 1950 .
he received his bachelor 's degree from Princeton in 1950 , and his master 's from Ohio State University in 1952 .		en 1952 , il obtient sa maîtrise de l' Université de l' État de l' Ohio et , 1956 , son doctorat de l' Université de Göttingen , en Allemagne .
he finished his studies in 1956 with a doctoral degree from the University of Göttingen in Germany .		toujours en 1956 , on l' engage comme professeur à l' Université de l' État de l' Ohio .

Figure 5.2: Example of badly aligned sentences

by the large volumes of good data. Obviously, the correctness of this assumption depends on the proportions of the two data categories (i.e. good versus bad).

It seems that, for the corpora automatically harvested from the web, the amount of wrong pairs is not insignificant. This is, at least, true for several corpora distributed in the international MT evaluations, namely the French-English Giga corpus and the Common Crawl corpus (French-English and German-English).[1]

We follow Munteanu and Marcu (2005) in using a binary classifier to detect and exclude the wrong sentence pairs. Such a classifier is trained and intrinsically tested on artificially constructed training data. Extrinsic evaluation is conducted in a machine translation task using MT evaluation data. Where it is shown that such a filtering process brings consistent improvements to the MT output.

The benefits of removing noisy pairs are manifold. First, more accurate scores are attributed to the extracted phrases. Ideally, this means that the probability mass is distributed only on the correct phrases. Second, one enhances the match to the test data which is usually carefully prepared. On top of all this, one reduces the training data size, leading to a more efficient learning.

[1]This year's WMT evaluation: `http://www.statmt.org/wmt15/translation-task.html`

5.2 Classifier-based filtering

Generally, the publicly available parallel corpora are distributed sentence-aligned. We suppose that such a corpus is noisy. At this specific situation, this means that a non-negligible number of sentences in one side of the corpus is not an adequate translation of its match in the other side. Our objective is to detect this kind of pairs and discard them.

Supervised binary classification has been a common tool of choice for this task. For example, both Munteanu and Marcu (2005) and Hunsicker et al. (2012) used a log-linear model to assign “parallel” or “non-parallel” labels to each sentence pair. This turns out to be a very convenient way in which one can plug-in as many useful features as desired.

As in the aforementioned works, we assume the existence of a good bilingual lexicon and a small clean parallel corpus. The lexicon is used to compute the different features and will be explained in sections 5.2.2 and 5.2.3. The small clean corpus, which is assumed to be sentence-aligned, is used to build a training and a test sets for the classifier. This parallel corpus will form our positive examples. The negative examples are obtained by randomly shuffling one side of the positive examples (these training and test sets are the artificial data we mentioned in the introduction.) As a matter of fact, the negative examples are a subset of the Cartesian product. We regard the proportion of negative examples as a free parameter which will affect the probability of rejecting an arbitrary pair. Consequently, providing higher number of negative examples would result in a higher precision but may at the same time lower the recall.

Two main differences distinguish our classifier from that of Munteanu and Marcu (2005). First of all, we use a simpler and reduced, but yet effective, set of features. This choice is justified by our need for efficiency. Indeed, this filtering process represents a preprocessing step in a long and complicated system building pipeline. Therefore, using many expensive (or relatively expensive) features would have a noticeable slow down of the training process. For instance, their classifier uses in total thirty-nine features whereas we use only nine. Many of these thirty-nine features are computed based on five types of alignments whereas our features are based on only two types of alignments. Furthermore, we adopt a simpler alignment approach since they need to minimize the number of crossings while we perform it in a more straightforward manner (as explained in Section 5.2.1.) All these simplifications become conceivably reasonable as one takes into consideration that our input is sentence-aligned and has already received a considerable amount of cleaning and filtering (while they consider the Cartesian product of all sentences in two aligned documents.)

The second difference is that we try to bias ourselves more towards precision than to recall. Missing a few correct pairs will not deteriorate our system, given the large amounts of data we are dealing with. This bias is reflected in the way we build our lexicon. Like Munteanu and Marcu (2005), we extract our lexicon from an automatically word-aligned corpus. However, we compute the lexicon probabilities in a slightly different manner (further details will be given in Section 5.2.3.)

5.2.1 Word alignment

With an associative lexicon in hand, the easiest word alignment model is the IBM Model 1 (IBM-1) due to Brown et al. (1993b). This model greedily aligns each target word to the source word with the largest lexical probability (found in the lexicon.) We use this idea to perform the alignment in two directions (e.g. En $\rightarrow$ Fr and Fr $\rightarrow$ En.) One issue which will be frequently encountered while applying this procedure is multiple occurrences of the same source word which is to be aligned to a given target word. All these source words are equally likely for the IBM-1. In our work, we select the source word which is relatively closer to the target word. In other words, the source word with the smallest DA feature (see Section 5.2.2.) This is a reasonable choice as it will prefer the points which are closer to the diagonal. Furthermore, It is noteworthy that the difference in relative distance has been commonly used in discriminative word alignment (DWA) models in order to measure the distortion of an alignment.[1] IBM-1 is very easy to compute and is good enough for our purpose. It gives sufficient evidence about whether a pair is parallel. A pseudocode which illustrates this procedure is given in Algorithm 1. As can be noted, the only addition to the IBM-1 here starts at line 9, where we impose an additional condition to get the source word with the minimal distance value. Note also the default alignment to NULL (initialization to the 0 position at line 3.) Alignment to NULL will affect only the LEX feature. Other features will be left unaffected by such an alignment.

[1]The literature on DWA models is extensive. We refer to a recent work by Niehues and Vogel (2008) and Tomeh et al. (2013) and the references therein.

Algorithm 1 Computing IBM-1 word-alignment

Input: Lex: lexicon, S: Source sentence, T: Target sentence

Output: L: Alignment points

```
1: $L \leftarrow \phi$
2: for $i \leftarrow 1, |T|$ do
3:     $a \leftarrow 0$; $a\_score \leftarrow 0$; $a\_dist \leftarrow \infty$
4:     for $j \leftarrow 1, |S|$ do
5:         if $(T[i], S[j]) \in Lex$ then
6:             if $Lex(T[i], S[j]) > a\_score$ then
7:                 $a\_score \leftarrow Lex(T[i], S[j])$; $a \leftarrow j$; $a\_dist \leftarrow \left|\frac{i}{|S|} - \frac{j}{|T|}\right|$
8:             else if $S[j] = S[a]$ then
9:                 if $a\_dist > \left|\frac{i}{|S|} - \frac{j}{|T|}\right|$ then
10:                    $a \leftarrow j$; $a\_dist \leftarrow \left|\frac{i}{|S|} - \frac{j}{|T|}\right|$
11:                end if
12:            end if
13:        end if
14:    end for
15:    $L \leftarrow L \cup \{(i, a)\}$
16: end for
```

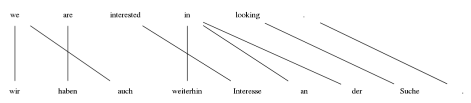
5.2.2 Features

Finding good features is an important step in any classification task. In most cases, it is an art and may require many “trial and error” attempts before settling on a selection. Our intention is to find features which are simple to compute and which give a good impression about whether a pair is parallel. Unfortunately, this might be language dependent and features which show good performance for one pair of languages could become unimportant for another. In this section, we present our collection of features which have shown very good performance for at least three language pairs.

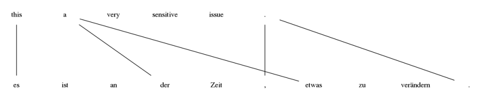
It turns out that the lexicon is of extreme importance as out of nine features, eight are computed based on this lexicon (they correspond to four features for both directions.) More precisely, these features are calculated based on a given word-alignment. This latter is determined beforehand based on the lexicon. For this reason, special attention was paid to the way we infer this lexicon. The remaining feature, is independent of the lexicon and is related to the distance measure in Gale and Church (1991). We also found

pi s ca u	î la %- C le %- à ro ss e
territorial contributions to Canada 's total emissions in 2002 .	Canada en 2002 .

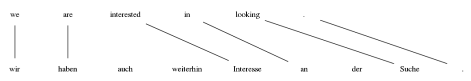
Figure 5.3: Example of length-imbalance



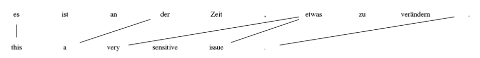
(a) Correct pair, En → De



(b) Incorrect par, En → De



(c) Correct pair, De → En



(d) Incorrect par, De → En

Figure 5.4: Example of lexicon-based word alignment

that using features having values between 0 and 1 gives a slightly better performance.

Difference in length (DL) This feature has been first used by Gale and Church (1991) in sentence alignment. Since then, it became a common practice to use it in related tasks such as Varga et al. (2007) and Moore (2002). It stems from the assumption that good translation pairs have a strong positive correlation in length as expressed in number of words or characters. In other words, long sentences translate into long sentences and vice versa. This feature is useful in detecting sentences with length imbalance. This means that one side has more content than the other or that the pair contains a lot of junk isolated characters resulting from HTML conversions. Figure 5.3 shows examples for both cases extracted from the Giga French-English corpus.

In our work, this feature is nothing but the difference in number of words between the source and target sentences normalized by their sum:

$$\mathrm{DL}\,(s,t) = \frac{||s| - |t||}{|s| + |t|} \tag{5.1}$$

Where s, t are source and target sentences and $|x|$ denotes the number of words in sentence x.

Alignment score (LEX) In a typical good sentence pair, many pair of words are likely to be encountered in other texts (corpora). This fact is acknowledged by the lexicon with high probabilities for such pairs. When aligned with the procedure described in Section 5.2.3, it is very likely that good pairs will be attributed a high total alignment score. This can be seen in Figure 5.4. Note the higher number of links in the correct pair 5.5(a) and 5.4(c) as compared to the incorrect pair in 5.5(b) and 5.4(d) (the two sub-figures for each case are the result of applying the alignment procedure in both directions.) Moreover, all links in the correct pair are common translations and thus should be of high scores in the lexicon. As other alignment-based features, this feature is directional. Therefore, it has two values and consequently needs two lexicons(e.g En→Fr and Fr→En.)

This score is computed as the average of log probabilities of the aligned words (found in the lexicon):

$$\mathrm{LEX}\,(s,t) = \frac{1}{|A|} \sum_{w,w' \in A} \log \left(\Pr_{\mathrm{lex}} (w \mid w') \right) \tag{5.2}$$

Where $\Pr_{\mathrm{lex}}$ is the conditional probability which represents the lexicon and A is the set of alignment points.

Unaligned source words (US) A good indicator of bad translation is the existence of many words in one side without corresponding translation on the other side. These are either unknown for the lexicon or none of their correspondents could be found on the other side. Figure 5.4 is again a good example for this case. Like the previous one, this feature has one value for each direction. It is noteworthy that, similar to typical word alignment models, all target words which are known by the lexicon should be aligned. If no correspondent is found, then this word is aligned to NULL, a special empty word which links to all unaligned target words.[1]

This feature can be expressed as follows:

$$\mathrm{US}\,(s,t) = \frac{|w \in s, \forall w' \in t, (w,w') \notin A|}{|s|} \tag{5.3}$$

Where A is the set of alignment points.

[1]More details about word alignment models can be found in Koehn (2010)

Maximum fertility (MF) Unfortunately, the unsupervised word alignment models align the frequent words (such as the punctuation marks) with an excessive number of different words from the other side. This results in a large number of entries corresponding to these words in the lexicon. For instance, in one of our lexicons, the number of different German words linked to the English comma "," amounts to 237 in a vocabulary of less than 200'000 words. The most probable words among these 237 words are the punctuation marks and conjunctions. Consequently, it is very probable to find a match for such words even if the current pair is incorrect. However, the chances are this kind of connections will be many to one. This feature is added to catch this situation. The maximum fertility is the largest number of target words linked to the same source word. Then, the score is normalized by the length of the longest target sentence so that it does not exceed 1.

$$\mathrm{MF}\left(s,t\right)=\frac{\max_{w\in s}\left(\left|w'\in t,\left(w,w'\right)\in A\right|\right)}{\max_{t'\in\mathrm{Target}}\left(\left|t'\right|\right)}\tag{5.4}$$

Where A is the set of alignment points and Target is the set of all target sentences.

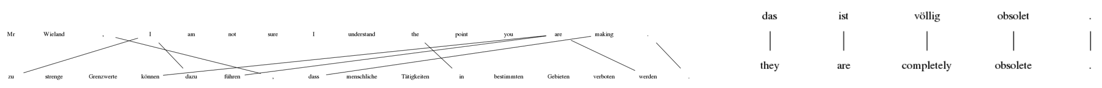
Distance of alignments (DA) This feature has already been mentioned in the discussion about our alignment model (Section 5.2.1.) In the alignment process, we used it to select an appropriate alignment point amongst several identical words. The average of the distances of these selected alignment points is then used as a feature.

A side effect of having a large number of distant alignments is an augmented number of crossings. Indeed, in some incorrect pairs, links of frequent words will be of long distance and will therefore create many crossings. Consequently, minimizing the relative distances will similarly result in a reduced number of crossings. Figure 5.5 shows two examples where in the average distance in the first is 1.42 and in the second it is 0. This figure also shows that many long distance links can be generated in an incorrect pair (the first), whereas the correct ones are usually of smaller distances.

This feature is computed as follows:

$$\mathrm{DA}\left(s,t\right)=\frac{1}{|A|}\sum_{(w,w')\in A}\left|\frac{\mathrm{Pos}\left(w,s\right)}{|s|}-\frac{\mathrm{Pos}\left(w',t\right)}{|t|}\right|\tag{5.5}$$

Where A is the set of alignment points and $\mathrm{Pos}\left(\mathrm{x},\mathrm{y}\right)$ is the position of word x in sentence y.


Mr Wieland , I am not sure I understand the point you are making .
zu strenge Grenzwerte können dazu führen , dass menschliche Tätigkeiten in bestimmten Gebieten verboten werden .
das ist völlig obsolet .
they are completely obsolete .

(a) Large DA value **(b)** Low DA value

Figure 5.5: Example of bad and good distance of alignments

5.2.3 Lexicon extraction

The lexicon is an essential component for our filtering. It is the only resource on which relies our alignment procedure. We use probabilistic lexicons where each entry consists of a source word, target word, and the conditional probability of the source word given the target word. Clearly, all entries corresponding to a given target word will sum up to unity. A lexicon in this sense is directional. Therefore, for a given language pair we have two lexicons, one for each direction (e.g. De $\rightarrow$ En and En $\rightarrow$ De.) Our lexicon is extracted from an automatically word-aligned clean corpus.

Word alignments are usually trained in an unsupervised manner (also referred to as generative models.) Although its supervised counterpart presents superior results, the unsupervised word alignment is chosen for its convenience. For one thing, it does not require human annotated alignments which are expensive to create and thus of very limited availability. Instead, an expectation maximization (EM) algorithm is used in order to deduce the correct alignments.

For computational reasons, many unsupervised alignment implementations assume a 1-to-many type of links. This is especially true for Giza,[1] the tool we use in our work. As a result, the commonly accepted approach is to train two different alignments in both directions and then combine them in some way. For example, Och and Ney (2003b) define three combination heuristics: intersection, union, and a refined version. The latter is a hybridization of the two former ones. It is the intersection augmented with some neighboring points from the union.[2] It is the typical combination used by many translation systems and is known to generate better phrases.

Two shortcomings of unsupervised alignment models will have a negative effect on the precision of our classifier. The first is the "garbage collection" phenomenon.[3] In a

[1]We use a multithreaded implementation of Giza: `http://www.cs.cmu.edu/~qing/giza/`

[2]This kind of combination is referred to as `grow-diag-final-and` in the `Moses` framework.

[3]The term was first used in Brown et al. (1993a). All the subsequent literature which improves on

nutshell, this problem is related to rare words. The EM training algorithm will prefer aligning the rare words to multiple words having no or infrequent correspondents to maximize the likelihood of the joint probability. It embraces this choice as it raises the likelihood of the current sentence without harming the likelihood of other sentences (because, for example in the extreme case, the rare word appears only in this sentence.) Including all these pairs in the lexicon will result in a lower precision. To make this last statement more evident, suppose that our lexicon was extracted from data where many frequent words were "garbage-collected" by many infrequent words. Then potentially these pairs will cooccur even in incorrect pairs (keep in mind that we have defined features, e.g DA and MF, which can deal with the case of frequent-frequent cooccurrences in wrong pairs) and cause them to receive a high lexicon score. This will lead the classifier to wrongly accept an incorrect pair.

We reduce the effect of this problem by selecting only reliable alignment points. We do this by applying the following two heuristics

1. Using the intersection combination. Put differently, we consider only the alignment points on which the alignments of the two directions agree. This avoids the "garbage collection" effect to some extent. Suppose a word behaves like a garbage collector when it is on the target side. When the direction is inverted, we could eventually find the correct alignment, if the translation of that word is not as rare. A similar argument was used by Liang et al. (2006) to motivate their proposed joint training. They jointly train the two directions and encourage them to agree.

2. Use association measures instead of the raw cooccurrences. Usually, the lexicon probabilities are estimated by normalizing the pair cooccurrence by the target word total occurrences (maximum likelihood estimation.) Even though the cooccurrence is a good measure of a pair association, in some situations it is not enough to indicate the association strength (for a more detailed discussion see Section 2.1.) A large number of more precise measures was proposed in the literature to detect word collocations (Evert (2008) gives an excellent presentation of a well-known selection of such measures for collocation detection purpose.) Munteanu and Marcu (2005) and Moore (2004) used the log-likelihood-ratio measure to have a more

the generative model alignments point out this effect. For instance, a good example illustrating this effect can be found in Moore (2004).

precise indicator of pair associations. The former used this measure to create positive and negative high precision lexicons so as to detect parallel substantial segments. Whereas the latter used it to generate a good initialization for the IBM-1 training. Using such measure in lieu of the raw cooccurrence will bias the distributions towards the more precise points.

The second issue corresponds to the probability of aligning to the NULL word. Using combination methods (especially the intersection) will rule out many alignment points. As a result, many words (especially the frequent ones) will end up having a high probability of being aligned to the NULL word. Because of this, we can possibly attribute high alignment score to an incorrect pair just because some words were aligned to NULL with very high probability.

To deal with this last issue, we allow the NULL to get its corresponding probability only through discounting (smoothing) rather than counting how many times a target word was unaligned. This way, we discount the alignment score of a pair without any alignments other than to NULL. For a given target word, we discount all its cooccurrences using a discounting scheme (e.g. Kneser-Ney discounting.) Then, we include the NULL word while redistributing the gained mass. In other words, the gained mass is redistributed over the source words with which the target word was aligned and the NULL word. The smoothing distribution could be either uniform or just the original conditional distribution. It has to be noted though that the latter smoothing distribution will differ from the original by including the NULL word.

5.2.4 Complexity

The computational complexity needs to be examined in the two main phases which make up our filter. These are the training and testing phase and the application phase. Nevertheless, the most dominant operation in the whole process in terms of computational complexity is the alignment computation. This operation has to be performed in both training and application phases. It is quadratic in the average sentence length.[1] This quadratic complexity stems from the fact that for a given pair of sentences, we examine each pair of words from their Cartesian product.

[1]Here we assume efficient hash-map is used to store the lexicon. We use the Python hash-map implementation which is in average $\mathcal{O}(1)$. More details about time complexity of Python data structures can be found in the dedicated URL: `https://wiki.python.org/moin/TimeComplexity`

The complexity of training the classifier is less important as it depends on the number of examples over which we have control. Usually, a couple of thousand sentence pairs is sufficient to obtain a well-performing classifier. After all, the training complexity sums up to the complexity of three components:

1. **Lexicon construction.** This operations is linear in the number of aligned word pairs. One lexicon (for a given direction) can be created by performing two passes over the list of aligned pairs (collecting counts, and then computing probabilities). If word associations are needed, then one has to perform a third pass, but the complexity remains linear.

2. **Feature computation.** this is quadratic in the average sentence length. However, the number of sentences at this phase is much smaller than at the application phase.

3. **Optimization.** The complexity of MaxEnt training implementation we use is approximately linear in the number of features and in the number of examples (See Daumé (2004) for a more elaborated discussion and pseudocode.) [1]

On the other hand, the application phase consists mainly of the alignment computation (if we neglect the additions and multiplications necessary for the score calculation). It has to inevitably be applied to each pair in the corpus which is to be filtered (e.g. this sums up to slightly less than thirty-million pairs in the Giga corpus.) All in all, the complexity of the application phase is $\mathcal{O}\left(NM^2\right)$. Where N is the corpus size and M is the average sentence length.

5.3 Evaluation

The classifier-based filtering is evaluated intrinsically and extrinsically. The Intrinsic evaluation is carried out in four different settings (En $\rightarrow$ Fr, Fr $\rightarrow$ En, En $\rightarrow$ De , and En $\rightarrow$ Ar,) whereas the extrinsic evaluation is performed only for our main pair of interest (En $\rightarrow$ Fr and Fr $\rightarrow$ En.) All the data used in these experiments has received the basic preprocessing. i.e. tokenization and smart casing.

[1]More precisely, the implementation we use in our work performs a predefined number of iterations each of which is linear in the number of features and as well in the number of examples. We use the MegaM package (`https://www.umiacs.umd.edu/~hal/megam/`)

Language pair	Sentence pairs ($\times 10^3$)	Source tokens ($\times 10^6$)	Target tokens ($\times 10^6$)
En $\rightarrow$ Fr	494.03	13.62	15.00
En $\rightarrow$ De	500.28	13.76	13.18
En $\rightarrow$ Ar	178.47	03.59	03.78

Table 5.1: Classifier data sets

5.3.1 Intrinsic evaluation

In these experiments, we use only the clean data. This is EPPS and NC[1] for the pairs: En $\rightarrow$ Fr, Fr $\rightarrow$ En, and En $\rightarrow$ De and TED data for En $\rightarrow$ Ar. The total number of pairs and tokens for each data set is presented in Table 5.1. while the first two configurations are comparable in terms of training data size, the last case tests the classifier in a data limited scenario.

This clean data is used to build the lexicon and the classifier training and test sets. We always make sure to keep these three sets (lexicon data, classifier training data, and classifier test data) strictly disjoint. We construct the classifier training and test by randomly extracting around 12'000 pairs from the aforementioned data. Third of these extracted pairs is used for testing and the rest for training. These sets form the positive examples. Each source positive example is paired with ten incorrect target translations and added to the data set as a negative example. Therefore, the positive to negative ratio is around 10%.

The bigger part of the clean data (i.e. the whole data except the 12'000 pairs which were held out for the classifier) is aligned in two directions using Giza++. Next, the two alignments are combined using the "grow diagonal" heuristic. The resulting combination is then used to compute two lexicons (one for each direction.) Our baseline lexicons are loaded with the maximum likelihood probabilities. We compare the performance of these baseline lexicons with the special NULL smoothing and with using the association measures.

In order to perform the NULL smoothing, different discounting techniques are tested. The first is Chen and Goodman (1996)'s modified Kneser-Ney with three constants. The others are our proposed parametric techniques with different discounting schemes[1] and

[1]In fact, this data is around half million pairs randomly extracted from the EPPS and NC corpora.

[1]The discounting schemes considered are: hyperbolic (Hyp), Power (Pow), and exponential (Exp) schemes.

Lexicon	Precision (%)	Recall (%)	F-measure (%)
Baseline	98.76	97.90	98.33
NULL smoothing (KN)	99.26	97.97	98.61
NULL smoothing (Chao,Exp)	99.31	98.33	98.82
Jaccard (Chao,Exp)	99.80	98.71	99.26
GMean (LOO,Hyp)	99.80	98.64	99.22

Table 5.2: Intrinsic evaluation for En $\rightarrow$ Fr

Lexicon	Precision (%)	Recall (%)	F-measure (%)
Baseline	99.53	97.71	98.61
NULL smoothing (KN)	99.49	98.47	98.98
NULL smoothing (Chao,Hyp)	99.57	98.51	99.04
GMean (Chao,Hyp)	99.84	98.69	99.26
GMean (LOO,Hyp)	99.79	98.71	99.25

Table 5.3: Intrinsic evaluation for Fr $\rightarrow$ En

unseen estimators [2] (cf. Section 4.2.) For a given target word, we use add-1 smoothed distribution to reassign the gained mass to its NULL-augmented set of aligned source words.[3]

From the large list of association measures, we choose three one-sided simple measures (Dice, Jaccard, and Geometric mean.) We excluded two-sided measures (e.g. log-likelihood-ratio measure) because they would need a special workaround for the negatively associated pairs. Indeed, two-sided measures attribute high scores to both positive and negative associations(see Section 2.1 for more details.)

From Tables 5.2 through 5.5, it can be clearly seen that our classifier prefers Precision over Recall. This is a design choice. Recall would be improved by including more forgiving features, such as those computed from the combination of the two alignments (especially the union,) and maybe a fewer number of negative examples. In addition to that, the way the negative examples are created makes them easier to distinguish than the hard positive examples. For instance, it is very difficult for our alignment procedure to align some correct pairs which were generated in both languages simultaneously

[2]The unseen estimators considered are: leave-one-out (LOO), Chao (Chao), Good-Turing (GT).

[3]In a small set of experiments, we found that there was no large difference between the add-1 smoothed and the uniform distributions, with the former being slightly better.

this oral amendment is therefore not put to the vote .		je vois plus de douze membres se manifester , l' amendement est dont rejeté .
this situation must change !		il faut que cela cesse .
that is not a pleasant prospect , given the state of unemployment which we have already .		de beaux présages pour un chômage déjà florissant .

Figure 5.6: Hard positive En → Fr examples

Lexicon	Precision (%)	Recall (%)	F-measure (%)
Baseline	98.95	97.83	98.39
NULL smoothing (KN)	99.43	98.11	98.76
GMean (KN)	99.45	98.63	99.03
GMean (LOO,Hyp)	99.43	98.58	99.00

Table 5.4: Intrinsic evaluation for En → De

and independently (which is the case for most of the EPPS corpus). Figure 5.6 gives examples of some test pairs which have escaped our best En → Fr classifier.

In all experiments, the baseline lexicon consists of the maximum likelihood probabilities and corresponds to the first line of every table. It shows very good performance in all conditions scoring higher than 90% in terms of the F-measure. Nonetheless, in most cases, these lexicons have the lowest Precision and Recall scores compared to those using NULL smoothing and association measures. All in all, NULL smoothing and association measures bring an improvement in Precision ranging from 0.3 to more than 2.5. Whereas the improvement in Recall ranges from around 0.8 to around 2.0. The impact of these techniques seems to be stronger on Precision whenever there is enough room for improvement (with the exception of Tables 5.3 and 5.4, where the Precision starts already at high values.)

The performance of NULL smoothing is weakly effected by the smoothing technique. Indeed, except the limited data case (Table 5.5,) the difference between the Kneser-Ney (KN) smoothing (appearing in the second line in all tables) and other techniques is negligible.

Lexicon	Precision (%)	Recall (%)	F-measure (%)
Baseline	92.99	89.68	91.30
NULL smoothing (KN)	93.41	90.75	92.06
NULL smoothing (LOO,Hyp)	94.33	90.39	92.32
GMean (LOO,Hyp)	95.74	91.56	93.60

Table 5.5: Intrinsic evaluation for En $\rightarrow$ Ar

Association measure	Unseen estimator	Smoothing scheme	Rank score
GMean	LOO	Hyp	1.0 ± 0.0
GMean	LOO	Exp	2.2 ± 0.4
GMean	Good-Turing	Exp	3.5 ± 0.6
Jaccard	loo	Exp	4.4 ± 0.7
GMean	LOO	KN	5.7 ± 4.3

Table 5.6: Aggregated ranks of the different techniques

Unfortunately, we could not observe a pattern of smoothing and association techniques which is consistent in all configurations. we could, nevertheless, see consistency with the same data set during intermediate experiments (even if we change the training and test sets or their sizes.) This makes the smoothing and association techniques data dependent. In spite of that, we performed a Kemeny-Young rank aggregation[1] over the ranked techniques according to the F-measure. [2] Table 5.6 shows the top five methods. It clearly shows that the the Geometric mean measure together with the Leave-One-Out unseen estimator and Hyperbolic (Hyp) discounting scheme is the best choice. We show the scores obtained by this combination at the end of each table (mostly after a dashed line.)

5.3.2 Extrinsic evaluation

The filtering process is applied to our noisy En $\rightarrow$ Fr data (Giga and Crawl,) described in Section 2.2.3. For this purpose, we use one of the models described in Section 5.3.1 with the lexicon resulting from all the clean data. All pairs which score less than a

[1]Details about Kemeny-Young rank aggregation can be found in Kemeny (1959) and Young (1995)

[2]For this purpose, we use the software at: `http://numerical.recipes/whp/ky/kemenyyoung.html`

System	Clean	Noisy	Filtered
Baseline	20.93	22.61	22.53
Large LM only	23.26	24.12	24.36
+Paralle LM	23.46	24.37	24.50

Table 5.7: Translation results for En → Fr

System	Clean	Noisy	Filtered
Baseline	21.55	23.64	23.98
Large LM only	24.18	24.98	25.24
+Paralle LM	24.24	25.11	25.35

Table 5.8: Translation results for Fr → En

certain threshold are rejected. Typically, this threshold would be 0.5. However, in these experiments, we set it to a relatively higher value, so that the resulting filtered corpus is comparable in size to the clean data, aiming for a fair comparison between them. This way, we reject around 26% from each corpus.

Then two systems were independently trained for each type of data for each direction. The "Clean" consists of EPPS and NC data. Whereas, the "Noisy" consists of all Giga and Crawl data. Then, the "Filtered" is the same "Noisy" corpus filtered. This "Filtered" corpus is comparable in size to the "Clean". The systems are tuned on Test2012 and tested on Test2013 and Test2014. Table 5.7 summarizes the BLEU scores for the En → Fr direction, whereas Table 5.8 shows the results for the opposite direction.

As mentioned in Chapter 2.2.3, the baseline system (first line of each table) includes the POS-tag reordering model and a language model trained on the target part of the parallel data. The second and third lines are added to test consistency of the behavior of the data. The difference between the systems in the second line and the baseline is exchanging the language model with a larger one, trained on around one-million sentences of monolingual data. In the third line, we include the language model used in the baseline as an additional feature into the systems of the second line. In the final line, we also add the bilingual language model trained on the aligned parallel data. We add this latter, to see whether the gain obtained by removing the noise gets multiplied as we use models whose quality depends on the quality of training data.

System	Noisy	Filtered
Baseline	27.47	28.25
+ Large LM	28.66	29.38

Table 5.9: Translation results for large En $\rightarrow$ Fr

5.3.3 Large Scale Translation Experiments

In this section, we show the results of our filtering in an Evaluation campaign settings. The systems presented here incorporate the most of our largest models; provided that the En - Fr pair is probably one of the richest pairs in terms of available parallel data. We perform an experiment with extremely large models in the En $\rightarrow$ Fr direction only, because of the resources such system consumes. The results are shown in Table 5.9. The first row corresponds to a configuration using all bilingual data and a language model trained on the target part of the parallel corpus. This baseline also uses the POS-based reordering rules. In the following row, we add another language model trained on all French data. That is all French part of the parallel data added to the Gigaword data which results in a 1.7 billion word corpus.

In both configurations the parallel data filtering improves the performance by more than 0.7 BLEU. The parallel data was reduced by almost 30% of its original size.

5.3.4 Filtering without a Clean Seed Corpus

In some situations, the assumption of the existence of a seed clean corpus might not hold. This will be especially true for the under-studied pairs. Of course, with some manual effort one can create this seed corpus. However, an automatic alternative would be to train the classifier on a pair of languages with enough resources and use it for the under-resourced pair. The lexicon for this latter is obtained from the alignment of the noisy corpus. In this situation, it would be more reasonable to use the "intersection" combination heuristic rather than the "grow diagonal" one. To test the feasibility of this idea, we use the classifier for En $\rightarrow$ De to filter the En $\rightarrow$ Fr corpus. The results are shown in Table 5.10.

System	Noisy	Clean Filtered	Noisy-filtered
Baseline	22.61	22.53	22.51
Large LM only	24.12	24.36	24.48
+Paralle LM	24.37	24.50	24.53

Table 5.10: Translation results for En $\rightarrow$ Fr with a noisy lexicon

5.4 Conclusions

To date,Out of the volumes of parallel data available for statistical machine translation the largest ones are undoubtedly those collected from the Web. The best features of such data are its reduced cost and large quantities. Unfortunately, its weakness resides in the fact that it might contain non-negligible amounts of noise. Moreover, even the data which we believe is noise-free may contain some.In the context of this chapter, this noise means sentences which are thought to be correct translation of one another, while in fact they are not.

Throughout this chapter, we showed the effectiveness of an automatic method to filter out the parallel noise. A binary classifier using 9 features was designed for this purpose. The most limiting prerequisite for this classifier is a probabilistic lexicon derived from clean data. This lexicon is used to obtain an alignment, which in turn is used to compute the different features.

While this technique is not new, it has been used in a slightly different context. In fact, it was used in order to extract parallel data from comparable corpora. Furthermore, we gave a special attention to its main building block. We enhanced the efficiency of the lexicon. In summary, our main contributions to the technique are the following:

1. Smoothing of the unalignment probability, where we allowed this probability to take its value only from a discounted mass rather than from a maximum-likelihood estimate.

2. Using the association measures instead of the raw cooccurrence, which in most cases bias the lexicon towards precision.

We also evaluated the technique extrinsically in different scenarios including very large scale tasks. We, in addition, explored the case when the lexicon could not be derived from clean data and gave an approach to overcome this limitation.

The technique not only did improve the translation quality, but also reduced the size of the data. In small data scenarios, with less than 75% of the data, we were able to achieve an improvement of up to 0.4 BLEU points. This improvement goes to up to 0.7 BLEU points in large scenarios, with around 70% of the data. We demonstrated that comparable improvements could be obtained without starting from a clean corpus by using a model trained for a different pair of languages.

However, it is worth mentioning that we did not perform any tuning on the acceptance threshold. Any pair with a score more than 0.5 was accepted and used in the training, otherwise it thrown away. Manual examination of the pairs in the neighborhood of this threshold suggests that its tuning would result in additional improvements, especially in the small scenarios. In the following chapter, we will see another filtering method, which can be applied to both monolingual and bilingual data.

6

Semantically-Guided Data Selection

In the previous chapter, we examined the case when examples of good parallel data exist. We used these examples to learn how to distinguish between good and noisy parallel data. In the current chapter, we would look into the case when good examples of monolingual data exist. Similar to the parallel objective, here we will use the good examples to select good data from a large monolingual corpus. In the literature, this procedure of selecting monolingual data similar to a given set is commonly referred to as "data selection".

The data selection process is commonly seen as an adaptation approach. That is we select the indomain (ID) data from a large pool of mixed domain data, so that the resulting model fits better the targeted domain. As a result, the large model will be "adapted" to the specific domain. We view this procedure as a kind of "denoising" the monolingual corpus. Although the out of domain (OOD) data itself can be good for other purposes, for a specific-domain task we consider it as noise since it deteriorates the fit of the model for the task. Moreover, when a general domain model is needed, it is always possible to design the set of good examples to be more general by merging data from multiple domains.

In this chapter, we present approaches to data selection. Our work extends the cross-entropy-based method of Moore and Lewis (2010). we introduce enhancements in two of its steps. First, we improve the procedure for drawing the out-of-domain sample data used for selection. Second, we use semantic word associations in order to extend the coverage of the good examples, reducing , thus, the possibility to fall into overfitting.

6.1 Introduction

The similarity between training and test data is a very strong requirement for the success of machine learning algorithms. The higher the similarity is, the more successful the algorithms are. This fundamental problem stems from the assumptions made by the learning algorithms. In particular, the training examples are assumed to be independent and identically distributed. Furthermore, the learned model is supposed to be applied to data drawn from the same distribution as the data on which this model was trained. Of course, such assumptions do not hold in the general case and the system would perform very badly if the difference between the training and test distributions is drastic. One way to address this shortcoming is to perform adaptation (Daumé and Marcu, 2006). This is achieved by using a general model which is tuned to the specific domain whenever a representative of this domain is available.

Statistical Machine Translation (SMT) suffers from this limitation too. For this reason, adaptation has been an active research direction for long time (Carpuat et al., 2012). One way to implement adaptation into SMT systems is by combining both a general domain model and an indomain model. Since our system is a log-linear model, combination can be performed by plugging the different features from both indomain and general models into the process, as applied by Koehn and Schroeder (2007). In SMT, the indomain data, however, is almost always of limited size. This necessitates supplementing it with out-of-domain (OOD) data in order to achieve satisfactory model estimates.

The scenario we are concerned with, in this chapter, is that we are provided with small indomain dataset. This indomain set could be monolingual in the target language only or parallel in both source and target languages. We are, then, requested to extract the most similar training data to this small set from a large corpus so that it can be used as indomain model. For example, in IWSLT tasks we intend to translate TED talks, but we are given large amounts of training data which are not very similar to the test sets. We are also provided with a small parallel corpus made up of TED talks. To take the best advantage of the large training data, we need to extract parts which are more similar to TED talks and use them as a separate model.

This problem was addressed in several studies (Gao et al., 2002; Klakow, 2000; Lin et al., 1997). However, one of the most successful and popular alternatives was proposed by Moore and Lewis (2010). This latter approach can be qualified as one based on the perplexity of the out-of-domain data. The in-domain data used in Moore and Lewis (2010) is the EPPS corpus, which contains more than one million sentences. The authors report their results in terms of perplexity, for which their technique outperforms a baseline selection method by twenty absolute points. Their approach has been shown to be effective for selecting LM training data, at least from the perspective of a SMT system with a specific domain task Durrani et al. (2013); Ha et al. (2013); Wuebker et al. (2011). We note that the main task of these systems was to translate TED talks.[1] The work in Moore and Lewis (2010) was extended to parallel data selection by Axelrod et al. (2011) and Mansour et al. (2011). However, the last work concludes that the approach is less effective in the parallel case.

The approach of differential LM scores used in the aforementioned literature has a long history in the information retrieval (IR) domain (Kraaij and Spitters, 2003; Lafferty and Zhai, 2001). Precisely, the first ref uses KL divergence between two LMs; while the second uses cross-entropy difference However, only unigram language models are considered in the context of IR, since the order in this task is meaningless.

Enriching the LM capability by incorporating word relationships has also been proposed in IR and is referred to as a *translation model* therein (Berger and Lafferty, 1999; Cao et al., 2005). More closely related to our approach, Dagan et al. (1999) uses word similarities to extend LMs in all orders. They show that extended LMs with properly computed word similarities significantly improve their performance at least in a speech recognition task.

We enhance the work of Moore and Lewis (2010) by drawing a better representative sample of out-of-domain data and LM vocabulary. More importantly, we extend this method by using a word-association based on a broad definition of similarity to extend the language models used during the selection process. With this extension, we do not compare solely the exact matching words from indomain and out-of-domain corpora, but also their semantically associated words. These semantic associations can be inferred, as detailed in Chapter 3, through the use of pre-existing non-domain-specific parallel and/or monolingual corpora. Then with a small amount of indomain data we use the aforementioned extended language models to rank and select out-of-domain sentences.

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

Using this extension, we widen the coverage of small indomain set. This will have a special positive impact if the indomain set is extremely small.

6.2 Selection Based on Cross-Entropy Difference

Moore and Lewis (2010) approach starts from two corpora: a large general domain and a much smaller indomain one. The small indomain corpus is the representative of the indomain data. The approach also needs a representative of the OOD data. This latter can be randomly sampled from the large OOD corpus. This sampled representative must be comparable in size to the indomain corpus, for a fair scoring. The ID and OOD representatives are, then, used to train two language models respectively. After that, two cross-entropy values are computed for each sentence in the large OOD using both language models. Then the sentence receives a score equal to the difference between the indomain and out-of-domain cross-entropies.

$$\omega\left(\widetilde{s}\right) = H_{ID}\left(\widetilde{s}\right) - H_{OOD}\left(\widetilde{s}\right) \tag{6.1}$$

where $\widetilde{s}$ is a sentence from the large corpus; $H_M\left(\widetilde{s}\right)$ is cross-entropy of the sentence $\widetilde{s}$ evaluated using model M. The entropy is defined as follows: $H_M\left(\widetilde{s}\right) = -\frac{1}{|\widetilde{s}|}\log \Pr_M\left(\widetilde{s}\right)$ If this difference exceeds a certain threshold the sentence is retained. The threshold can be tuned on a small heldout in-domain set.

Axelrod et al. (2011) adapts this scoring to the bilingual case, by adding the two differences on the source and on the target sides respectively.

$$\omega\left(\widetilde{s},\widetilde{t}\right) = H_{ID_s}\left(\widetilde{s}\right) - H_{OOD_s}\left(\widetilde{s}\right) + H_{ID_t}\left(\widetilde{t}\right) - H_{OOD_t}\left(\widetilde{t}\right) \tag{6.2}$$

where M_s is a model trained on data from the source side and M_t is trained on the target side.

6.3 Enhancements

In this section, we describe three enhancements we introduced to the original selection method. The purpose of these enhancements is to improve the model's distinction capability and to extend its coverage, so that it generalizes better.

6.3.1 Drawing an OOD Representative Sample

In the cross-entropy method of Moore and Lewis (2010) previously described in Section 6.2, the out-of-domain LM is taken simply as a random sample of the larger out-of-domain data upon which we do selection, *OD*. However, randomly-drawn text may represent both in-domain as well as out-of-domain data (OD). The out-of-domain LM should instead represent the kind of data which we seek to exclude from our selection. Since the in-domain data should be the furthest from the latter kind of data, we reasoned that the in-domain LM could be used to intelligently select the data for the out-of-domain LM. We do this by first scoring the sentences in *OD* with the in-domain LM for perplexity (with a closed vocabulary). As some of our data in *OD* comes from web crawls, the sentences with the highest perplexity are mainly "junk" coming from automatic text processors and/or converters. The sentences with the lowest perplexity are mostly in the in-domain set. Therefore we specify some range around the median perplexity (m) as being a legitimate region from which to select sentences for the out-of-domain LM. In our case we chose $m \pm 0.5m$ with m being the median perplexity. Then for our out-of-domain LM we randomly draw an appropriate number of sentences from this range. The probability of any particular sentence being drawn is proportional to its corresponding perplexity.[1]

6.3.2 Vocabulary Selection

Intuitively, we could think of vocabulary words as indicators of the importance of a sentence. Words occurring with high frequency in both in- and out-of-domain data sets would be of lower interest. In contrast, words frequently encountered in the in-domain, only, indicate that the sentence is of high importance. It was not clear to us whether the words which are common in the out-of-domain only would be a negative indicator. That is why we experimented with different ways for choosing the vocabulary on which the LMs are based. The first vocabulary is taken as the intersection of the in- and out-of-domain vocabularies $V_1 = voc\{ID\} \cap voc\{OD\}$. The second vocabulary incorporates the first and adds those words which occur with high frequency in the in-domain source only. This is $V_2 = V_1 \cup hf\{ID\}$. The third incorporates the second (and consequently

[1]For the weighted random sampling without replacement, we use the algorithm described in Efraimidis and Spirakis (2006)

the first,) adding those high-frequency words occurring only in the out-of-domain LM dataset. Thus $V_3 = V_1 \cup hf\{OD\}$ A visual representation of this scheme is depicted in figure 6.1.

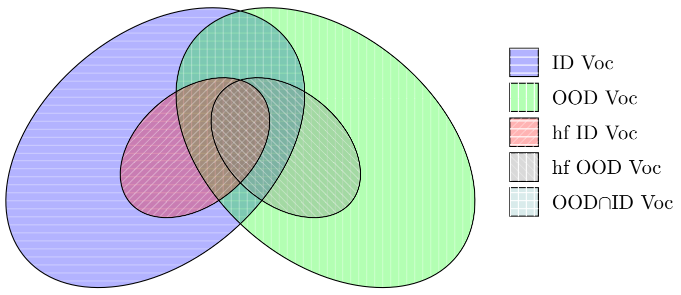


Figure 6.1: Diagrammatic representation of vocabularies of in- and out-of-domain sources

6.4 Extended Cross-Entropy Selection

What we intend to do here is to include additional words in the selection vocabulary based on their associations with the vocabulary words. This achieved by including them in the language model with proper probabilities. The associations are obtained from a lexicon of semantic associations. We discuss how the probabilities are determined and how the lexicon is obtained in the following.

6.4.1 Lexicon of Semantic Associations

A probabilistic model of semantic associations is needed to establish the LM extension. Such model is referred to as *translation model* in the IR related literature. We will use the term "semantic association model" (or just association model for short) in order to avoid confusion with our phrase table which is also called translation model in the SMT literature. This probabilistic model will be a table giving conditional probabilities of a word being associated to other words. We denote such model $\mathrm{t}(w_2 \mid w_1)$. This model should satisfy: $\sum_j \mathrm{t}(w_j \mid w_i) = 1$ for any word w_i.

In Chapter 3, we presented an extensive analysis of extracting semantic associations between words using both bilingual and monolingual corpora. The semantic association model t is straightforward from bilingual alignments. As one may observe from Equation (3.1), the resulting model is the proper distribution we are looking for. On the other hand, the monolingual associations would arm us only with word vectors. Fortunately, the dot product of these vectors in the Glove (and obviously in its improved version mGloVe) approximates the logarithm of the cooccurrence. All we have to do is to compute dot products, exponentiate them and then normalize to end up with a proper distribution.

$$\mathrm{t}\left(w_2 \mid w_1\right) = \frac{\exp\left(\mathbf{w_1} \cdot \mathbf{w_2}\right)}{\sum_j \exp\left(\mathbf{w_1} \cdot \mathbf{w_j}\right)}$$

where $\mathbf{w}$ denotes the vector representation of word w.

Unfortunately, the last procedure pointed out for the monolingual case is very expensive. It is $\mathcal{O}\left(|V|^2\right)$, where V is the vocabulary. Our solution to this shortcoming is to perform a K-Means clustering before applying the procedure. The procedure is run on each cluster independently, afterwards. The complexity, then, becomes $\mathcal{O}\left(N|C|^2\right)$, where N is the number of clusters and C is the average cluster size. For example, by taking $N = \sqrt{|V|}$, the complexity becomes $\mathcal{O}\left(|V|^{3/2}\right)$ in average. In addition, performing the computation intra-cluster only makes the process highly parallelizable.

Another simplification we perform prior to semantic association computation is to consider only useful words. In other words, we let w_1 takes only values from the selection vocabulary and w_2 to take its values only in the general model vocabulary. This approximation should not change the model's performance as we only need to extend the selection vocabulary, and we extend it to only the general vocabulary. This simplification reduces the computation effort drastically.

6.4.2 Extending the Selection LMs

According to the cross-entropy selection, the out-of-vocabulary (OOV) words will have only a small effect on a sentence score. This is due to the fact that they are mapped to <unk> (the unknown word,) and therefore the probability returned from one model (e.g. the in-domain) cancels its counterpart from the other (e.g. the out-of-domain.)[1] Consequently, including more "important" words in the model with a realistic likelihood would conceivably make our model more robust.

To enrich the selection LMs with semantic associations, we add to the unigram order those OOV words which are associated with the words in the selection vocabulary.

[1]This effect will mostly be a penalization. Based on our experiments, almost always the probability of <unk> is larger in the out-of-domain model

Therefore, these new unigrams can contribute to evaluating the sentence probabilities by the back-off mechanism. We found that the rate of backing-off to these new words, in one of our models, is about 20%. Such high back-off rate demonstrates the need for these underlying words. These words would have been recognized as OOV, had not this extension been performed.

The integration of the new unigrams is performed as follows. First, we discount the probabilities of the vocabulary words to free some a priori fixed mass (say $1 - m_0$.) Afterwards, each word added from the lexicon receives a share from m_0 proportional to two factors. The first factor is the LM probability of the associated vocabulary words. The second factor is the strength of the lexicon association connecting the OOV word to the in-vocabulary words. Note that m_0 is a tunable parameter. In our experiments, we found setting $m_0 = \Pr(\text{<unk>})$ to be satisfactory.

Now, we will have a new vocabulary consisting of the original LM vocabulary together with the additional words from the extension. Let's name the first vocabulary VOC_{LM} and the extension vocabulary VOC_{EXT}. Then the probability of a (unigram) word $w \in VOC_{LM} \cup VOC_{EXT}$ can be expressed as follows:

$$\Pr_E XT(w) = \begin{cases} m_0 \Pr_{LM}(w) & \text{if } w \in VOC_{LM} \\ (1 - m_0) \sum_{v:v \in VOC_{LM}} \mathrm{t}(w \mid v) \Pr_{LM}(v) & \text{otherwise} \end{cases} \tag{6.3}$$

$\Pr_{LM}$ is the original back-off LM probability; and $\Pr_{EXT}$ is the new extended model and t is the association table associating a vocabulary word v to a non-vocabulary word w. This procedure results in a new LM whose vocabulary is a superset of the original vocabulary.

Although the generalization of this idea to higher n-gram orders is straightforward, we restrict ourselves to the unigram level only. At higher orders this operation becomes costly and memory greedy.

6.5 Evaluations

In a previous research, we demonstrated the effectiveness of the approaches described in the chapter for models used in Automatic Speech Recognition (Mediani et al., 2014). However, in this evaluation, we will use different datasets. We will use an IWSLT system, trained for TED talks. We adopt this approach because of two main characteristics of this task. First, TED talks have a special structure different from that encountered in News data such as EPPS. This is due to the fact that TED data are transcriptions of talks. Although the talks are in different topics but their style is somehow similar. The second characteristic is that this task has very limited amounts of transcription data (usually less than 300K sentence pairs for any language pair). Using other sources of data such as EPPS and parallel Giga is a common practice in the evaluations. The data selection would be very appropriate in this scenario, in order to take the best advantage of the additional data.

We use the French-English TED corpus and two test sets from the IWSLT campaign. We use one of the two sets for parameter tuning (call it DEV) and the other for testing (call it TEST). In addition, as out of domain data we use the EPPS and Giga corpora, used used and described in the previous chapters. Statistics of this data are shown in Table 6.1.

Set	Sentences	English		French	
		Words	Voc.	Words	Voc.
TED	219 404	4 451 924	57 247	4 682 408	75 628
Dev	887	20 262	3 225	18 822	4 078
Test	818	14590	2431	14 511	3 138
EPPS	494 028			14 720 495	7 6851
Giga	984 047	-	-	26 541 672	331 366

Table 6.1: Statistics of the datasets used in this evaluation

The selection is performed in the out of domain sets (EPPS and Giga) using the indomain set (TED). As explained earlier, a representative is drawn from the out of domain set and two models are used to score the sentences in the out of domain corpus. The scored sentences are, then, ranked and only a percentage of the best scoring sentences is kept. In Table 6.2, we compare three settings of the selection: Moore & Lewis approach, then our introduced enhancement, and finally our extension using semantic associations. For the sake of completeness, also the perplexity from the full model without any filtering is included. Enhancements include random selection around the median, and combining the vocabulary from high frequency indomain words together with the intersection between the indomain and out of domain vocabularies.

Technique	% Retained Sent. (ppl)			
	1	5	10	20
TED only	228.08			
EPPS Corpus				
EPPS	397.888			
Moore & Lewis	290.60	260.49	286.04	319.74
Enhancements	276.91	258.56	280.77	322.81
+ Extension	227.58	233.93	278.451	317.99
Giga Corpus				
Giga	376.491			
Moore & Lewis	297.99	287.78	291.86	298.63
Enhancements	291.86	284.88	288.8	299.63
+ Extension	282.73	277.09	284.28	295.02

Table 6.2: Perplexity on Dev of the LMs selected using TED corpus

The percentages shown in the table vary between 1% and 20%. As in previous experiments, this experiment also confirms that the perplexities start at a large value for the very small percentages decrease towards a minimum at a point around 5% and then start to increase again. We can also observe that our enhancements are consistently outperforming Moore & Lewis around the minimum. The enhancements sometimes worsen the performance, but this always happens in non optimal percentages.

The extension in the last rows for both EPPS and Giga corpora was performed using the combined setting, which includes bilingual and monolingual semantic associations. However, as stated in Chapter 3, these semantic associations were extracted using the EPPS corpus. This means that these extensions, still obviate further improvements by using richer corpora, such as the Parallel Giga corpus after its cleaning. We, also, think that as the associations are trained on the EPPS corpus their influence on this corpus is relatively better. Indeed, examining the extended vocabularies in these corpora, and the vocabularies to which they were extended supports the last claim. The extended vocabulary in EPPS is around 300 words more than the Giga. The EPPS words were extended to an indomain vocabulary including 400 words more than the Giga. Added to this that the EPPS vocabulary itself is way smaller than the Giga vocabulary.

Translation results on Test are shown in Table 6.3. All translation models in differ-

Technique	BLEU scores	
TED only	35.87	
	EPPS+	Giga+
Full	35.98	36.00
Moore & Lewis	36.22	36.04
Enhancement & Ext.	36.24	36.46

Table 6.3: Translation results on Test

ent configurations are built only on TED. We exchange the language models between different settings. The top row is the baseline and includes an from the French TED corpus only. All the following language models are linearly mixed with the TED model. The weights of these mixtures are tuned on Dev. However, the log-linear weights were only tuned once for Dev with the TED corpus, and then used with all other models.

It should be also stressed that the model relying on Giga selection is almost always double the size of the one relying on EPPS. This is because the former is originally double the size of the latter, and the percentage is fixed. Therefore, it is not surprising that the Giga model is slightly better than the EPPS, in most configurations. Our enhancements consistently outperform Moore & Lewis, and in the best conditions, we improve by 0.4 over this method.

Technique	BLEU scores
Moore & Lewis	36.34
Enhancement & Ext.	36.85

Table 6.4: Translation results on Test for bilingual selection

In a further experiment, we performed selection on the bilingual EPPS corpus, and added the selected data to the TED corpus, to build a new translation model. As for the LM data we used the selection from Giga with the highest improvements in the previous experiment. Here the improvement over the original TED corpus performance gets extremely close to 1 BLEU point. The improvement over Moore & Lewis, however, remains comparable to the monolingual case.

6.6 Combination of all methods

We perform an additional experiment to see how the different methods combine. The results presented in Table 6.5 show how nicely the different methods can be stacked to gradually improve the translation quality. The Baseline here is a system trained on TED only (Same as in Table 6.3). The next step is to add all noisy data to the system as an additional model. This addition improves the system by around 0.25 BLEU, thus slightly better than adding the Giga corpus alone. After that, we preform the selection on the monolingual Giga and Crawl corpora, which boosts the performance with an additional 0.64 BLEU. The system could be further improved with around 0.3 BLEU by performing the selection on the associated parallel training data. In the last step we run the parallel filtering examined in Chapter 5 on the selected data, which slightly improves with an additional 0.1 BLEU. This final rather small improvement is due the fact that most of the noise has been already removed in earlier stages by the selection. As a result, stacking the different approaches brings in total around 1.3 BLEU improvement over a basic Baseline.

Technique	BLEU scores
Baseline	35.87
+ Noisy corpora	36.13
+ Mono Selection	36.77
+ Par. Selection	37.08
+ Par. Filtering	37.22

Table 6.5: Results of stacking all the previous methods

6.7 Conclusions

We presented several extensions and enhancements to the state-of-the-art indomain data selection method of Moore and Lewis (2010). Our techniques bring consistent improvements to the performance of the language and translation model. All with a very small portion, usually in the neighborhood of 5% of the training data.

We enhanced Moore & Lewis in two levels. First, at the sampling of the out of domain representative. We found that adding the words appearing in both selection models to increase the distinction power of the scores, as opposed to using only frequent words in the indomain selection model. Additionally, limiting the sampling of the out of domain representative to the neighborhood of the median helps getting better representatives of the out of domain.

More importantly, we perform a guided widening of the selection vocabulary. We add words which are originally OOV for the selection model. We add them if they are semantically associated to a word already in the model, with an appropriate probability. This was also proved to work well, especially if the model has a limited vocabulary (such as EPPS).

However, an interesting question arises here. Should the semantic associations be also adapted to the domain? It should be noted that in our experiments, we use associations obtained from the EPPS corpus. Our intuition suggests that the impact would be better, if the associations come from the same domain. For instance, the word "language" would be associated to "communication" in a linguistic context rather than to "programming" in a computer science context. However, the investigation of this question is out of our scope.

We would also like to stress that many parameters of the extension were not explored, but rather set to arbitrary value. The number of associations per extensible word, and the minimum frequency of associated words are examples of such parameters. tuning such parameters may bring additional small gains to the model.

7

Soft Selection: Noise-Based Weighting

In the previous chapters, we examined two types of noise spanning complete sentences. The idea was to detect the noisy sentences and then proceed by removing them from the training corpus. The detection was accomplished using a provided set of good examples. Three shortcomings maybe obvious to envisage from such an approach. First, in some cases, the set of good examples can be hard to obtain. Second, if the removal is executed aggressively, we might end up removing useful sentences and in the worst case will overfit to the provided examples. Finally and more importantly, some noise comes in higher degrees of granularity, making it difficult to detect using the previous approaches, and even if it can be reliably detected, it might be wasteful to remove a whole sentence because of this small error. A common example of the latter case is a mistyped word in a perfect sentence. This chapter addresses these shortcomings by introducing a weighting mechanism for the data instances based on how clean they are.

Of course, the aforementioned example of a mistyped word will not be harmful if it happened only once. What makes this kind of problems interesting, however, is that they can result from the automatic preprocessing tools of the text corpora; In which situation, the number of the error occurrences will be most likely large, since the automatic tools tend to repeat the same treatment for the same inputs. In the monolingual case, the diversity of text encodings in the documents which build up the corpus may lead the preprocessor to output poor sequences because of the removal of badly encoded words. In the bilingual case, because of the automatic sentence segmentation, we can come across sentence pairs partly matching, capable of escaping the binary classification task in Chapter 5.

The main idea behind our weighting approach is that most of the aforementioned examples can be detected using association measures. In fact, both translation and language models are estimated from cooccurrence data. The significance of a cooccurrence can be rated by an association measure. We establish our weighting on the significance scores returned by the association measure. We give details about monolingual weighting first, and the bilingual version of the weighting will be discussed afterwards.

7.1 Weighting in Language Modeling

Here, we are concerned with noisy sequences of words in a monolingual corpus. Association measures will be able to spot a word if it is placed in an unusual location as compared to its other occurrences in the corpus. This is possible because such unusual placement will result in less significant cooccurrence expressed by a low association score. We first apply this idea on the n-gram level, where the association between the last word and the left $(n-1)$-gram context is used to penalize the n-gram count if it is too bad. After that, we explore weighting complete sentences based on their cleanliness.

7.1.1 n-gram Level Weighting

By applying association measures to the n-gram counts, we will obtain a valuation of how strong the last word is associated to the left context. We interpret this valuation as indication on the n-gram's cleanliness. The higher the value of the association is the cleaner the n-gram is. Some examples of n-grams with low association values, extracted from the EPPS corpus, are given in Table 7.1.

Even though the EPPS corpus is thought to be very clean, to our surprise, most of the examples in Table 7.1 are valid examples of noise. Note that most of these worst examples consist of frequent words. Certainly, some association measures, such as the log-likelihood-ratio, will be more sure about the association of a cooccurring pair if its both sides occur very often. Most of the examples in this table, seem to have some missing words. In some of these examples, the word would be likely missed by the transcriber at the computer input. For instance, the first example in the bigram section in this table would sound correct if we add "that" between "be" and "we", which could

n-gram	OCC	LLR$\times10^{-5}$	GMean$\times10^{-5}$	Example from corpus
			$n=2$	
be we	2	45.59	1.60	we feel the common average should be we allow the Council to say : that is too high .
the are	3	7.98	1.03	elements of the are non - negotiable .
to to	4	4.02	0.97	there is therefore a procedure which allows any Member State that wants to to use vaccination ,
the and	25	2.08	4.38	the EU suggests going beyond the and overriding due process in the event . . .
			$n=3$	
that it the	1	74.99	1.06	that it the situation up to and after Camp David .
of the to	3	11.91	1.26	who have not belonged to the communist or been part of the to fill posts . . .
of the of	4	11.11	1.63	Mr President , high standards of human rights are part of the of the EU
of the and	5	14.17	2.29	we have lived through the disasters of the and the , from both of which we have been . . .
of the .	10	10.21	3.89	we often hear mention of the .

Table 7.1: Examples of n-grams with low association scores from the EPPS corpus (for $n = 2$ and $n = 3$)

OCC: the n-gram occurrence; LLR: loglikelihood-ratio score; GMean: geometric mean score

be missed easily by humans. By contrast, some of these missing words are likely to be removed by a preprocessor. For example, the last example of the bigram section needs a word between "the" and " and", this was missing in the original EPPS corpus. However, a Web search query with this exact sentence finds some results suggesting that the name "acquis communautaire" is the one missing from this sentence. The reason why a preprocessor would filter out such a name is unknown to us. Some other examples include mistaken repetitions, such as the bigram "to to", or typos such as the first example of the trigrams, which will sound more correct if "it" between "that" and "the" is replaced by "is".

Now, to transform the association scores into weights, we assume that they are strictly positive. We assume, as well, that these scores are one sided, meaning that the lowest scores correspond to the worst pairs and vice versa. While the first assumption is fulfilled for the measures we consider in this thesis, the second requires a transformation for the two-sided measures, like the loglikelihood-ratio (LLR) measure. We use the same transformation mentioned in Section 3.2.4.1, precisely using the Equation (3.5). Then, we express the weight of an n-gram w_i^{i+n-1}, as follows:

$$\omega\left(w_i^{i+n-1}\right) = \left(\frac{\mathrm{A}\left(w_i^{i+n-2}, w_{i+n-1}\right)}{1 + \mathrm{A}\left(w_i^{i+n-2}, w_{i+n-1}\right)}\right)^{\alpha}, \ 0 < \alpha \leq 1 \tag{7.1}$$

where w_i^{i+n-2} is the left context and can be composed of one or more words, and w_{i+n-1} is the right-most word in the n-gram, and α is a parameter between 0 and 1.

The weights in Equation (7.1) have values between 0 and 1. However, a desirable feature would be to keep the positively associated pairs with their original counts. That is where the parameter α comes in handy. For very low value of α, most of the larger weights will have no influence as they will converge to 1. On the extreme, if α is set to 0, no weighting is used. The parameter α can also be seen as a control over the degree of influence of the weights on the n-gram counts, and should correlate to the degree of "noisiness" of the corpus.

After the n-gram counts are multiplied by the corresponding weights, most of them will turn into fractional pseudo-counts. To estimate a language model from such counts, we use our parametric smoothing, explained in Chapter 4.

7.1.2 Sentence Level Weighting

At a coarser level of granularity, we attribute scores to sentences depending on how clean they are. Clearly, this operation has a different goal compared to the previous, and thus, will deliver a different outcome. In this level, we are targeting complete sentences which look unusual in the pool of the whole corpus. Therefore, and as can be noted from the examples in Table 7.2, this task results in detecting sentences from other languages or which consist of special codes. Similarly, sequences made up of menu items can also fit here. Such errors are not detectable by the n-gram level approach. Indeed, the sequencing of the words will look perfectly usual for the association measures. For example, the word "da" will be strongly associated with the word "dum" in the first TED example, and so will the word "quelque" with "chose" in the first EPPS example,

From Giga Corpus
UAE HKG IND INA IRI IRQ JPN Jor KAZ KGZ KUW Lao Lib MAS MDV MGL Mya NEP OMA UZB Pak ple PHI qat PRK SIN Sri SYR TJK TPE tha TLS tkm vie YEM
50g soup oignon 60g potage 1boite jumbo 50 petits cubes poulet spaghetti 400g macaroni 500g macaroni 1kg spaghetti 500g rice vermicelli 400g ...
Burkina Faso NGOs Burkina Faso NGOs Burkina Faso NGOs Burkina Faso NGOs Burkina Faso NGOs Burkina Faso NGOs Burkina Faso NGOs Burkina Faso NGOs
Gen ray Henault LGen Marc Dumais MGen Brett Cairns BGen Linda Colwell BGen Stan Johnstone BGen Paul McCabe
From EPPS Corpus
Monsieur Bolkestein , je veux vous dire quelque chose !
Deánaim comhghairdeas ó mo chroí le John Hume as ucth na dúise Nobel a bhuachan .
Putin u0027s Soft Authoritarianism
From TED Corpus
♫ Dum da ta da dum ♫ ♫ Dum da ta da dum ♫ ♫ Da ta da da ♫ That is a lot of power .
reserve component %-% National Guard reserves overwhelmingly Sys Admin .

Table 7.2: Examples of sentences with lowest scores from various English corpora

as it is very unlikely that they will appear with other different words in the English corpus.

In our approach to sentence weighting, a sentence score is a function of its average probability evaluated by models trained on samples randomly drawn from the corpus. We randomly draw a fixed small number of sentences from the corpus, train a language model on that sample, and then use the latter model to evaluate the cross entropy of each sentence in the corpus. This procedure is repeated for a given number of times. A bad sentence will receive low probability from all sampled models and conversely a good sentence will be consistently acknowledged by different samples. Because the drawn samples may not be balanced in terms of vocabulary size, the cross entropies due to a given sample are standardized by subtracting their mean and dividing them by their standard deviation. In addition, this standardization brings the scores into a manageable range. Afterwards, the cross entropies are averaged. Finally, like Zhang and Chiang (2014), we use the sigmoid function to convert the average cross entropies into weights. This procedure can be formalized in the following equation:

$$\omega\left(\widetilde{s}\right) = \left(1 + \exp\left(\frac{1}{|S|}\sum_{S}\frac{H_S\left(\widetilde{s}\right) - \mu_S}{\sigma_S}\right)\right)^{-\alpha}, \ \alpha > 0 \tag{7.2}$$

where $H_S\left(\widetilde{s}\right)$ denotes the cross entropy of the sentence $\widetilde{s}$ evaluated using the model trained on the sample S, and is given by: $H_S\left(\widetilde{s}\right) = -\frac{1}{|\widetilde{s}|}\log \Pr_S\left(\widetilde{s}\right)$, and μ_S and σ_S are, respectively, the mean and standard deviation of the cross entropies corresponding to the sample S. α is a parameter having the same purpose as in Equation (7.1). Counting n-grams in a weighted corpus is achieved by summing up all the sentence weights where the given n-gram appears.

Sentence weighting can also be used in data selection. In their experiments to demonstrate how the fractional Kneser-Ney smoothing works, Zhang and Chiang (2014) weight sentences in function of the difference in cross entropy between an indomain and an out of domain models. Unlike the routine practiced in Chapter 6 which consists of removing low scoring sentences, in the weighted version of data selection all data is kept with an appropriate weight.

7.2 Weighting in Translation Model Training

In general, modeling from parallel data is more vulnerable to the noise introduced by the automatic processing than its monolingual counterpart is. Simply, because the former is carried out in longer automatic pipelines. Usually, both models share the sentence segmentation and tokenization. Building Translation models introduces, in addition, sentence and word alignments, and phrase extraction. Moreover, as these operations are cascaded, the errors get accumulated as the training progresses.

Our focus here is on the outcome of the aforementioned operations. In other words, the translation equivalences delivered in the form of phrase pairs at the end of the phrase extraction process. Even in a perfectly clean parallel corpus, many of these resulting phrase pairs will contain noise. One notable trigger of this noise is the unaligned words. In fact, the extraction heuristic, gradually consumes all surrounding unaligned words of an aligned word, adding them one-by-one and generating a phrase on each addition. Certainly, a large number of the spurious pairs generated by this procedure will be filtered out later in a phrase table pruning step. Nevertheless, a non-negligible number of them is still able to make it into the pruned table. Table 7.3 supports this claim as it is taken from the pruned version of a phrase table trained on the EPPS corpus.

all the phrases in Table 7.3 align on one word only: the English determiner "the" and its French equivalent "le", "la", or "l'". The other words were included because they were not aligned. In reality, the translations in these examples are generated from a paraphrased source sentence, which means that the translation is not literal. Such translations are real challenge to the learning process, in that they can only be modeled by many-to-many alignments. It is well-known that such many-to-many alignments generate considerable amount of noise, since the automatic word alignment handles only one-to-many alignments. Okita et al. (2010) explores this problem and improves

En. Phrase	Fr. Phrase	En. Sent.	Fr. Sent.
the number of	de la	shortcut in the Jobeet project root directory to shorten the number of characters you have to write when running a task .	à la racine du projet Jobeet pour faciliter l' écriture de la commande lorsque vous exécutez une tâche .
of the	le Président	I should like therefore to emphasize some of the especially important aspects of the new agreements .	donc , Monsieur le Président , permettez - moi de souligner quelques aspects particulièrement importants des nouveaux accords .
the Commission	de l'	we have made good progress with what the Commission has put forward , but the Commission 's programme is lacking a great many important parts .	avec ce texte présenté par la Commission , nous allons de l' avant mais il manque tout de même dans ce programme certains chapitres essentiels .

Table 7.3: Examples of noisy phrases from the pruned phrase table

The first example comes from the Giga corpus and the last two from EPPS

the alignment by excluding sentences containing such alignments, and then realigning the retained corpus. We tackle this problem by weighting in two different levels of granularity: phrase level and sentence level.

7.2.1 Phrase Level Weighting

Just like the n-gram weighting, we apply the association measures to the phrase counts. We compute an associated weight for each phrase pair, multiply it by the cooccurrence, and then pretend this weighted cooccurrence is the observed cooccurrence. In the same way as well, we transform the association scores into weights:

$$\omega\left(\widetilde{s},\widetilde{t}\right) = \left(\frac{\mathrm{A}\left(\widetilde{s},\widetilde{t}\right)}{1+\mathrm{A}\left(\widetilde{s},\widetilde{t}\right)}\right)^{\alpha}, 0 < \alpha \leq 1 \tag{7.3}$$

where $\widetilde{s}$ and $\widetilde{t}$ are respectively source and target phrases.

It is worth noting that, under the same principles, association measures were used to prune large phrase tables by Johnson and Martin (2007). They use the Fisher's exact test probabilities and each phrase having a Fisher's test probability less than a certain threshold is removed. It is known, however, that Fisher's exact test is computationally

En. Phrase	Fr. Phrase	COOC	LLR
, with a wide variety	, avec une grande variété	1	38.5892
tiroir de la	drawer of the	1	38.5892
secondary schools and	écoles secondaires et	1	35.8166
remarquerez	'll notice	2	55.5455
et	.	659	0.0003
.	,	2839	0.0011
.	and	2347	0.0007
et	to	1093	0.0010

Table 7.4: Examples of phrases having large distance between the cooccurrence and LLR rankings

demanding. For this reason, Moore (2004) suggests to use the LLR, which is an accurate-enough approximation. In Table 7.4, we give examples of phrase pairs having the largest differences between the cooccurrence and the LLR ranks.

The upper section of Table 7.4 gives examples of phrases with low cooccurrence values but which were given high LLR scores. The lower part is the opposite, i.e. pairs having high cooccurrence but low significance. As previously mentioned, this lower part consists of very frequent single words, because these are the pairs for which the cooccurrence significance can be most reliably estimated. Almost in all the examples shown in this table, the LLR is getting it right. The examples in the upper section are perfect translations. Regardless their utility, most of the examples in the lower section are noisy. For instance, translating "et" to "." or "." to "and" is an artifact of sentence merging/splitting, a common practice followed by human interpreters. Such phrases could be responsible for the insertion of periods in the middle of a sentence or a conjunction at the end.

It was mentioned in Chapter 2 that we adopt the phrase table smoothing proposed by Foster et al. (2006a). Like in n-gram weighting, we resort to our parametric smoothing which supports fractional counts, as the weighting attributes non integral counts to the phrases. This kind of smoothing was presented in Chapter 4.

En. Sentence	Fr. Sentence
so far so good .	nous sommes d' accord .
this is an extremely sad situation .	hélas , trois fois hélas .
he has got what he wanted .	le voilà servi .
we are discussing both of these matters .	le vote sur l' accord - cadre aura lieu à 12 heures précises .
what about the United States ?	on pourrait ajouter bien d' autres choses .
they know what is expected of them .	les réformes politiques ont renforcé la démocratie .

Table 7.5: Examples of bad sentence pairs from the EPPS corpus

7.2.2 Sentence Pair Level Weighting

By analogy to the monolingual weighting, the coarser level in the bilingual case corresponds to weighting sentence pairs. In this task, we will use the lexical connection between the underlying sentences. Pairs with poor lexical connection will be down-weighted. Such sentence pairs, usually, generate noisy phrases. This idea is motivated by the fact that the lexical scores are direct indicators of the alignment strength. These alignments are the backbone around which the phrases are built. Examples among the weakest 20 pairs from the EPPS corpus are shown in Table 7.5.

Although the first three examples in Table 7.5 are correct translations, they correspond to multi-word expressions or idioms which are inadequate to learn any subsentence correspondences, and their lexical contributions will most likely be undesirable. The remaining examples are translations matched incorrectly. We examined one of these examples in the original corpus to find that it was due to sentence segmentation, and two sentences were joined later, on one side, to recover from this shift.

Our approach consists of using a bilingual probabilistic lexicon to compute the cross entropy of a sentence pair using the provided word alignment. The cross entropy values are used to generate pair weights in a similar manner to sentence weights in the monolingual case. However, because we are armed with lexicons and alignments in two directions, we take advantage of both by summing the cross entropies in both directions.

$$\omega\left(\widetilde{S},\widetilde{T}\right) = \left(1 + \exp\left(\frac{H\left(\widetilde{S} \mid \widetilde{T}\right) + H\left(\widetilde{T} \mid \widetilde{S}\right)}{2}\right)\right)^{-\alpha}, \alpha > 0 \tag{7.4}$$

where $\widetilde{S}$ and $\widetilde{T}$ are respectively the source and target sentences; and the cross entropy $H\left(\widetilde{X} \mid \widetilde{Y}\right)$ is the cross entropy considering both aligned and unaligned words in the direction $\widetilde{X} \to \widetilde{Y}$, defined as follows:

$$H\left(\widetilde{X} \mid \widetilde{Y}\right) = -\frac{1}{|A| + |U|}\left[\sum_{(i,j)\in A} \log \Pr\left(x_i \mid y_j\right) + \sum_{j\in U} \log \Pr\left(\epsilon \mid y_j\right)\right]$$

where A is the set of alignment points and $U = \{j | 1 \leq j \leq |\widetilde{Y}| \text{ and } \nexists i, 1 \leq i \leq |\widetilde{X}| \text{ and } (i,j) \in A\}$ is the set of unaligned target words, and where x_i and y_j denote, respectively, the words x and y at positions i and j in sentences $\widetilde{X}$ and $\widetilde{Y}$. The symbol ϵ represents the empty word, also referred to as the NULL word.

In equation 7.4 the lexicons for both directions are given the same weight. It is, of course, possible to give them unequal weights and tune them appropriately, but for simplicity, we do not explore this possibility here.

For the forward and backward lexical probability models, we reuse the techniques from Chapter 5 introduced to build precision-biased lexicons. More precisely, in Section 5.2.3, we presented heuristics which suggest to replace the cooccurrences by the association scores and which estimate the unalignment probability only through smoothing. In addition to this, we also try to reduce the number of phrases from the worst sentence pairs. While this reduction can be accomplished by removing the lowest scoring phrases coming from these sentence pairs, we achieve this goal by using the union combination heuristic on a small portion of the worst sentence pairs. As with all versions of our weighting, here again we use our parametric smoothing described in Chapter 4.

7.3 Evaluation

In this section, the proposed weighting techniques are evaluated. We start with the monolingual version, where the performance will be reported both in terms of perplexity and BLEU scores. For the sake of comparison, we also include the Witten-Bell-smoothed language models. This can be also seen as an extrinsic evaluation of our smoothing proposed in Chapter 4. The parameter α was set to 0.4 in these experiments. We briefly tested couple of values between 0.1 and 0.9 and this selected value performed fairly well in all configurations.

		English		French	
Corpus	Model	WMT-Test	IWSLT-Test	WMT-Test	IWSLT-Test
		Kneser-Ney Smoothing			
TED	KN	339.810	113.894	390.412	80.847
	P.Smooth.	328.068	112.290	379.791	80.194
	NG-weighted	326.289	113.566	375.807	81.8353
EPPS	KN	312.494	231.393	380.479	146.985
	P.Smooth.	296.329	218.284	365.952	140.664
	NG-weighted	294.559	217.859	360.066	140.866
GIGA	KN	205.652	153.385	272.133	128.062
	P.Smooth.	202.116	151.600	266.989	127.178
	NG-weighted	202.005	151.771	265.297	127.979
		Witten-Bell Smoothing			
TED	WB	540.126	160.83	580.853	109.485
	P.Smooth.	373.931	122.7971	426.8941	87.0841
	NG-weighted	371.575	120.953	424.084	85.882
EPPS	WB	524.919	374.941	586.32	224.499
	P.Smooth.	364.215	261.726	436.325	163.421
	NG-weighted	363.84	262.837	431.093	165.774
GIGA	WB	290.66	209.776	369.381	178.098
	P.Smooth.	215.328	160.66	281.195	135.245
	NG-weighted	212.854	159.346	276.306	134.962

Table 7.6: Perplexities of weighted models using test sets from Chapter 4

P.Smooth.: Parametric smoothed model; NG-weighted: n-gram weighted version of the parametric smoothed model

7.3.1 Weighting in Language Modeling

We present evaluations of both n-gram and sentence level weightings. The perplexity evaluations are direct extension for the models in Chapter 4. We use the same corpora and same testsets. The parametric smoothing configurations used here are the same tested on the testsets when we introduced the methods in Chapter 4. These selected configurations were the best performing on the devset.

The translation experiments, on the other hand, extend the baseline systems in Chapter 5. We weight the EPPS corpus and show the weighting effect on a system using only one model trained on this corpus.

The n-grams are weighted using the log-likelihood-ratio measure. We chose this measure because it outperformed the others in some preliminary experiments we con-

Model	French → English	English → French
	Kneser-Ney Smoothing	
KN	23.19	22.31
P.Smooth.	23.20	22.48
NG-weighted	23.35	22.56
	Witten-Bell Smoothing	
WB	22.56	21.99
P.Smooth.	22.67	22.30
NG-weighted	22.85	22.38

Table 7.7: Translation results for the WMT-2014 testset, using weighted and unweighted EPPS language models

P.Smooth.: Parametric smoothed model; NG-weighted: n-gram-weighted version of the parametric smoothed model

ducted on a small sample. The weights are applied in all orders greater or equal 2. Obviously, the association measures assume cooccurrences between items, which are not available at the unigram level. Additionally, as in the smoothing experiments in Chapter 4, all language models are of order 4. The intrinsic evaluation results are shown in Table 7.6, whereas the extrinsic translation results are given in Table 7.7. In these tables the "P.Smooth" rows correspond to using the smoothing without any weighting.

In general, the effect of the weighting is rather smaller on KN models than on WB models. We think the reason for this would be the adjusted counts in KN models. Using the unique occurrences of the right contexts in the lower order models, can be itself considered as some sort of cleaning. Additionally, Table 7.6 suggests that the effect of smoothing is larger than the effect of the weighting. We should remind ourselves, however, that the corpora which are considered noisy here, have already received some cleaning in Chapter 5.

Even though the gains are small in Table 7.7, unlike the perplexity results, they give no clear distinction whether the gains are due to the smoothing or rather to the weighting.

Table 7.8 gathers other weighting experiments for the direction French → English. The first row is the same baseline as in Table 7.7. The next line shows the BLEU score obtained by sentence-weighting the corpus. The following two rows are for phrase and sentence pair level weighting for TM respectively.

Model	French → English
Unweighted	23.19
Sent. weighted	23.28
Phrase. weighted	23.36
Sent. pair weighted	23.81

Table 7.8: Translation results for the WMT-2014 testset

- Sent. weighted: the LM is sentence weighted. The TM is unweighted
- Phrase. weighted: Weighted TM at phrase level
- Sent. pair weighted: Weighted TM at sentence level

The sentence level weighting in the monolingual data seems to have a small effect. Perhaps because the kind of noise this kind of weighting addresses is very limited and has little effect on our test data. Indeed, by examining the output, the approach could reliably detect sentences from foreign language, and sentences consisting of codes only, and so on. Still the effect is not as strong.

Again, the results are very small for Phrase pair weighting for TM. However, The sentence pair level weighting in TM is rather more promising. It improves the baseline by around 0.6 BLEU. We think the reason for the low effect of the phrase level weighting is that it is almost redundant with the pruning we perform prior to decoding. Indeed, for each source phrase only 10 best target phrases are kept. Therefore, the phrases with low weights are already removed before the decoding, which cancels, to some extent, our phrase-based weighting.

The sentence pair weighting, on the other hand, works better. It weights whole sentences, leading to more reliable scores because of the larger context. Additionally, unlike the phrase based weighting, in the sentence pair weighting, all features will be weighted accordingly. More precisely, the lexical weights will also receive weighted counts, by counting through weighted sentence pairs. These latter are not weighted in the case of phrase based weighting, as the weighting happens in a later phase after estimating the lexicons.

7.4 Conclusions

This chapter presented our approaches to noise-base weighting. Instead of filtering some parts of the data, we attribute them weights based on how much we trust them. We perform this operation at different levels of granularity and with different types of corpora. In the monolingual case, we accomplish it at the n-gram or at the sentence level. In the bilingual data, we perform the weighting on the phrase level or on the sentence pair level.

The most promising approach seems to be the weighting of bilingual corpora on the sentence pair level. This weighting is closely related to the filtering we performed in Chapter 5. Conceptually, they are built on the same assumptions. The sentence pairs which share less translated words should be noisy.

The small improvements of the other approaches does not mean they are useless. In fact, manual examination of the process reveals their correct behavior. Unfortunately, this has little impact, either because of the overlap with other operations in the case of phrase weighting, or because of the nature of the corpus in the case of monolingual sentence weighting. More investigation is needed to find the best suitable way these approaches could impact a translation system.

Now that we have overviewed the techniques to deal with the quality of the noisy data, in the next chapter we look into another axis of this kind of data. We will pay attention to the quantity and develop tools to speed up the training.

8

Parallel Phrase Scoring for Extra-large Corpora

Our main concern in the previous three chapters was to improve the quality of the models trained on noisy data. Another characteristic of the noisy data which has not been looked into, so far, is its quantity. As its acquisition is relatively cheap, it usually comes in large volumes. Training an SMT system on such large datasets is time consuming. Even worse, some computations may no longer fit into the main memory.

One of the most expensive operations in the SMT pipeline (Figure 2.2) is the phrase scoring. In this task, cooccurrences and marginal counts are collected for each phrase pair. However, the number of phrases extracted is typically very large, even from a moderately-sized corpus. The common approach to accomplish the scoring is to perform the computation on chunks and use the physical disk to hold the intermediate results. The popular tools implement this approach simplistically, neglecting thus the powerful capabilities offered by most of the hardware platforms available today.

In this chapter, we discuss an implementation of the phrase scoring in phrase-based systems that helps to exploit the available computing resources more efficiently and trains very large systems in reasonable time. Three parallelizing methods are presented. The first exploits shared memory parallelism and multiple disks for parallel IOs while the two others run in a distributed environment. We demonstrate the efficiency and consistency of our methods, in the framework of the Fr-En systems we developed for the WMT and IWSLT evaluation campaigns, in which we were able to generate the phrase table in one third up to one sixteenth of the time taken by *Moses* for the same tasks.

8.1 Introduction

Phrase scoring is one of the most important and yet very expensive steps in phrase-based translation system training. Typically, it consists of estimating the corresponding scores for each unique phrase pair extracted from an aligned parallel corpus. Usually, the scores are estimated based on two directions (from source to target and vice versa). Therefore, the process is accomplished in two runs. In the first run, counts are collected and then the scores are estimated based on the source phrases while in the second run a similar task is performed based on the target phrases.

This process is memory greedy. However, for non large corpora it could be performed efficiently in the physical memory by some implementations. For instance, *memscore* Hardmeier (2010) uses a lookup hash table based on STL[1] maps to index the phrases. Then the hash identifiers are used to directly access the corresponding phrases in order to update the marginal and joint counts. Unfortunately, this does not scale very well for corpora of large sizes. As a matter of fact, a memory requirement of more than 60GiB was reported for a corpus of 4.7M sentence pairs (Hardmeier, 2010).

On the other hand, most systems such as the widely used phrase-based system *Moses* (Koehn et al., 2007), handle the memory limitation by streaming the large data sets, keeping only a limited amount of data into memory, and saving temporary results into disk. In fact, all the pairs which correspond to a given phrase should be kept into memory while gathering the marginal and joint counts for this phrase. Consequently, the streamed data must be sorted depending on whether the computation is being held based on source phrases or target phrases. In *Moses*, this is achieved by performing two sorting operations using the standard Unix *sort* command.[2] Even though, being a good external memory sorting tool, the Unix *sort* command is not optimal when the corpus is very large. For instance, the runs are formed and sorted serially, it lacks support for multiple disks, and the IO could not be overlapped with the computations.

Gao and Vogel (2010) developed a platform for distributed training of phrase-based systems starting from word alignment until phrase scoring. Even though excellent speed gains were reported, this system runs on top of the Hadoop framework, and therefore needs the platform to fit this special infrastructure.

[1]C++Standard Template Library `http://www.sgi.com/tech/stl/`

[2]`http://unixhelp.ed.ac.uk/CGI/man-cgi?sort`

Unlike applications which operate exclusively on data stored in main memory, applications which involve external memories such as hard disks face an additional challenge with the high data transfer latency between the external and main memory. For this purpose, data structures and algorithms have been developed in order to minimize the IO overhead and to exploit the available resources such as parallel disks and multiple processors more efficiently (Vitter, 2008). Luckily, different external memory APIs have been created in order to make the underlying disk access and low level operations transparent to programmers. Such platforms include, but are not limited to, LEDA-SM (Crauser and Mehlhorn, 1999), TPIE (Arge et al., 2002), Berkeley DB (Olson et al., 1999), and STXXL (Dementiev and Kettner, 2005).

The main goals of our tools for phrase scoring are to exploit CPU and disk parallelism in an external memory environment, so that the phrase sorting and score computation are performed more efficiently. The CPU parallelism is ensured by the OpenMP library (Chapman et al., 2007) (eventually coupled with an MPI implementation in distributed environments (Pacheco, 1996)), while the disk parallelism and other external memory functionalities are ensured by the STXXL library. STXXL is preferred over the other environments due to its superior performance, ease of use (STL-compatible interface), and explicit support for parallel disks (Dementiev et al., 2008).

Most of our tools are written in C++. The underlying CPU parallelism comes in two flavors: multithreaded, hybrid. The multithreaded version uses shared memory parallelism and therefore runs on a single node. In the hybrid setting, multiple nodes can be used, each of which also exploits the shared memory parallelism.

8.2 Phrase Scoring

The goal of the Phrase Scoring is to estimate phrase pair conditional probabilities. Traditionally, these probabilities are equivalent to relative frequencies. i.e.

$$\Pr\left(\widetilde{s} \mid \widetilde{t}\right) = \frac{\mathrm{c}\left(\widetilde{s}, \widetilde{t}\right)}{\sum_i \mathrm{c}\left(\widetilde{s}_i, \widetilde{t}\right)} \tag{8.1}$$

where c is a function returning the number of times its arguments cooccur. Of course, this probability is estimated in the other direction too.

A fundamental problem with the relative frequency used in Equation (8.1) is that it overestimates rare phrase pairs. For this reason, Koehn et al. (2005) proposed to decompose the phrases into their word translations. They use the underlying alignment a and a word-to-word translation lexicon w to compute an additional feature, named "lexical weights":

$$\mathrm{lex}\left(\widetilde{s} \mid \widetilde{t}, a\right) = \prod_{i=1}^{|\widetilde{s}|} \frac{1}{|\{j \mid (i,j) \in a\}|} \sum_{(i,j)\in a} w\left(s_i \mid t_j\right) \tag{8.2}$$

In other words, we multiply the lexical probabilities of the aligned words. if a source word is aligned to multiple target words, we average the probabilities. The unaligned source words are taken to be aligned to the NULL word. Again, this feature will be computed in two directions.

Another approach which also addresses the problem of rare phrases uses smoothing. Foster et al. (2006b) smooths phrase probabilities in a way similar to LM smoothing, using Chen and Goodman (1999) version of the Kneser-Ney smoothing. The phrase cooccurrences are discounted and the gained mass is then redistributed using a smoothing distribution:

$$\Pr\left(\widetilde{s} \mid \widetilde{t}\right) = \frac{\mathrm{c}\left(\widetilde{s}, \widetilde{t}\right) - \mathrm{D}\left(\mathrm{c}\left(\widetilde{s}, \widetilde{t}\right)\right)}{\sum_i \mathrm{c}\left(\widetilde{s}_i, \widetilde{t}\right)} + \alpha\left(\widetilde{t}\right) \Pr_b\left(\widetilde{s}\right) \tag{8.3}$$

where D is the absolute discounting constant described in Equation (4.2). $\alpha\left(\widetilde{t}\right) = \frac{\sum_{\widetilde{s}_i} \mathrm{D}\left(\mathrm{c}\left(\widetilde{s}_i, \widetilde{t}\right)\right)}{\sum_i \mathrm{c}\left(\widetilde{s}_i, \widetilde{t}\right)}$ is the gained mass. $\Pr_b$ is the smoothing distribution and can be taken as the maximum-likelihood probability of the source phrase (i.e. count of this source phrase normalized by the total number of source phrases), or the lower order distribution of source phrase (i.e. the number of unique target phrases with which it occurs normalized by total number of unique target phrases).

Smoothing the phrase probabilities is another application where our proposed smoothing could be used. The elaboration is the same as described in Chapter 4. Our proposed smoothing was applied to phrase table, earlier in Chapter 7. At the end of the Scoring step, each phrase is assigned four scores: Two smoothed probabilities and two lexical weights

8.3 External Memory Sorting in STXXL

Due to its extreme importance, the external memory sorting has received continuous improvements over the years. The different techniques can be categorized in two classes:

distribution sorts and merging sorts. Distribution sorts try to partition the data into buckets while keeping all the elements in a given bucket at the same order compared to the elements of any other bucket. The buckets are then sorted in internal memory. Finally, the sorted data is simply the concatenation of the buckets. On the other hand, merging sorts proceed in two phases. In the first phase, the data is scanned in the form of *runs* which could fit into memory. Every run is sorted internally and then saved to disk. In the second phase the sorted runs are gradually merged in several passes. A detailed survey of both approaches can be found in Vitter (2008).

Details about STXXL sort implementation are given in Sanders and Dementiev (2003). In the following, we briefly review its important aspects.

STXXL implements a multiway-merge sort. It assumes that the data records are of fixed size. The processing then could be held on fixed size data blocks. The STXXL library forms the backbone of many sorting benchmark[1] winners in the past years (Andreas et al., 2011; Beckmann et al., 2012; Rahn et al., 2009). The two key steps of STXXL sorting are as follows:

Run formation In a double buffering strategy, two threads cooperate to read/sort the different runs. The first thread sorts the run which occupies half of the sorting memory, while the second thread is either reading the next run or writing the sorted run. The sorter thread creates lighter data structure consisting of only the keys and pointers to the actual elements. After that, it sorts the keys in the new data structure where the sorting method depends on their number (straight line code if it doesn't exceed 4, insertion sort if it is between 5 and 16, otherwise it uses quicksort).

Multiway merging In order to define the order in which blocks will be streamed into the merger, the smallest elements in each block are recorded in a sorted list during run formation. The position of an element in this list defines when its containing block will enter the *merging buffers*. The merger keeps a number of blocks equal to the number of the sorted runs in merging buffers. In order to minimize the time of selecting the current smallest element, the keys of the smallest elements of all blocks in merging buffers are kept in a tree structure.

[1]`http://sortbenchmark.org/`

STXXL uses an *overlap buffer* for reading and a *write buffer* for writing in order to overlap IOs and merging. The size of the overlap buffer depends both on the number of runs and the number of parallel disks while the size of the write buffer depends on the number of disks only. If the write buffer has a number of blocks which exceeds the number of disks, a parallel output is submitted. Similarly, if the overlap buffer has a number of free blocks which exceeds the number of disks, a parallel read is performed.

Distributed External Memory sorting (DEMSort) is an extension of the STXXL sorting so that it fits the distributed case where the sorting is rather performed on multiple machines (Rahn et al., 2010). The key difference here is the introduction of an additional intermediate phase between run formation and multiway merging: the so-called *Multiway selection.*

Like the distribution sorts, the multiway selection tends to find global splitting points over all the sorted runs. By the end of this operation, each node knows its exclusive range of data. Afterwards, the data are redistributed globally over the nodes using an all-to-all operation to satisfy the range constraints. In this case, the MPI interface is used for the inter-node communication. Finally, the merging is done locally as explained before.

8.4 Software Architecture and Algorithms

Like *Moses* scoring tool, our phrase scoring tools take three files as input and produce a phrase table as output. The first input file contains the extracted phrases (called '`extract.0-0.gz`' in Moses convention) and the other files are two bilingual dictionaries which model $w(s \mid t)$ and $w(t \mid s)$ for every source and target words s and t if they are aligned at least once ('`lex.0-0.f2e`' and '`lex.0-0.e2f`' in Moses convention).

Typically, the phrase table records 4 scores for every extracted phrase pair. Relative frequency and lexical score for each direction (source to target and vice versa). Our lexical score is identical to the one produced by Moses Scoring tool, whereas our relative frequency is smoothed using modified Kneser-Ney smoothing as described in Foster et al. (2006b).

The development of our tools led to two different levels of parallelism: multithreaded, hybrid. The multithreaded version forms the core of the other version. In the following, we explain each of these versions.

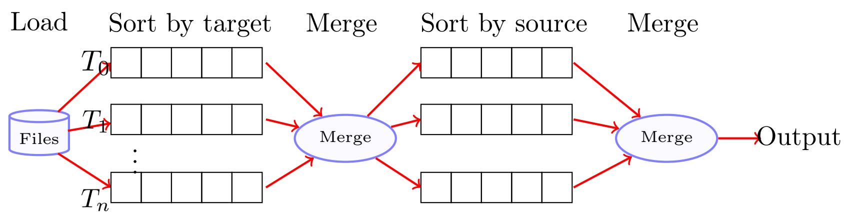


Figure 8.1: Multithreaded phrase scoring architecture

8.4.1 Multithreaded Phrase Scoring

The basic data structure used in this software is STXXL vector whose interface is similar to STL vector but it rather stores data which does not all reside in memory. STXXL vector elements are stored in the form of key, value. The keys of this vector are the phrase pairs (source and target phrases concatenated) and the values are the different counts. In order to satisfy the fixed size record of STXXL vectors, the keys are represented by a fixed-length string.

As depicted in Figure 8.1, the process consists of several threads, each of which takes care of one large STXXL vector of data. The phrase table is the result of five consecutive steps. Details about each of these steps are presented in what follows.

Loading the data First of all, the lexical dictionaries are loaded into two STL maps (one for each direction). Afterwards, each thread reads one phrase pair at a time, computes its lexical score, and then loads it into its corresponding STXXL vector. This multithreaded way allows for computations and IOs to be overlapped.

There are two ways to read pairs from the file into memory. The fast way: where all the threads read the same file concurrently one line at a time. In this case, the input file should not be zipped. The alternative way allows to read directly from the zipped file, the master reads from the file and pushes the lines into a FIFO queue. The other threads pop lines from the queue and process them.

As soon as the loading is complete, the lexical maps are disposed since they will not be needed anymore.

Sorting by target phrases Every thread sorts its vector by simply calling the STXXL sort function which performs a multiway merging sort on the corresponding vector.

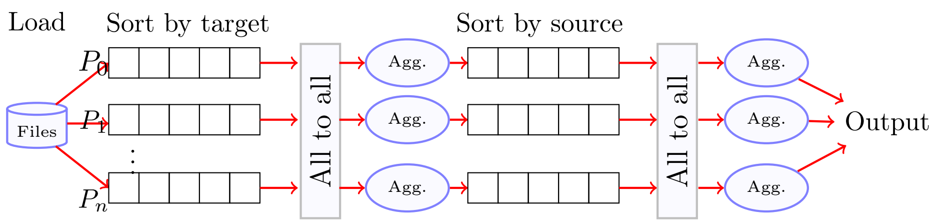


Figure 8.2: Hybrid phrase scoring architecture

Merging and computing the target-based scores The merging follows the same approach as the multiway merging. The first elements from all vectors are organized in a tree structure. Whenever an element is taken out, it is replaced with the next element from the same vector.

Parallel threads acquire a lock on the tree and get all the pairs with the same target phrase in a local vector, then release the lock for the next thread. After collecting the pairs, every thread updates the corresponding count fields and writes the updated records to a new STXXL vector. Since the identical pairs have to be uniquified in this step, our implementation allows choosing one lexical score and one alignment based on maximal lexical score or the most occurring one.

Sorting by source phrases Again, this is done in parallel by the STXXL sort.

Merging and computing the source-based scores This operation is identical to the merge based on target phrases.

Writing out the phrase table Like the loading phase, two writing ways are possible. The way which supports writing zipped phrase table is performed by a single thread while the multithreaded way writes only unzipped files.

Optionally, all the counts can be recorded for further use (as in the distributed version). It is as well possible to write out an optional abridged phrase table containing only phrases which match a list of given n-grams.

8.4.2 Hybrid Parallel Phrase Scoring

The extension DEMSort allows us to efficiently sort an STXXL vector spread over multiple interconnected machines. There are only few changes in the architecture compared to the previous version. We suppose that the nodes dispose of a shared disk space. First of all, all the nodes build the lexical maps in the same way. Afterwards, every node reads a quota of the input file of phrase pairs into an STXXL vector. Running the DEMSort could raise the following issue: the phrase pairs which correspond to a given phrase could be spread between two adjacent nodes due to the redistribution as explained in Section 8.3. To fix this, every node sends all the phrase pairs corresponding to the first phrase to its immediate predecessor. As a consequence of this sorting approach, no further data exchange between the nodes is needed. Figure 8.2 schematizes the hybrid scoring. This figure shows that the main differences to the shared memory version lays in aggregation operation, which consisted of a merging in the shared memory case. In the hybrid case, it comprises a communication followed by a local aggregation.

Every node performs the local merging and scoring strictly identical to the multithreaded version. In our development process, this resulted in an unbalanced load between the nodes. Consequently, we extended the merging with a dynamic load balancing strategy. The final merging procedure executed on every node looks as follows:

1. Execute a multithreaded merging and listen to signals from other nodes
2. If request for sharing is received from another node, then send half of the remaining pairs to that node
3. When finished, signal all other nodes
4. If all nodes have no remaining pairs, then exit
5. Receive half of the remaining pairs from the node with the largest remaining number of pairs
6. Go to 1

The output is done in a similar manner to the previous system where all the nodes write to the same file concurrently. The position from which a node starts writing in the common output file is estimated based on the number of entries in this node's vector.

8.5 Evaluation

In this section, we show some performance comparisons between the different versions of our scoring tools. We compare them as well to Moses. The hardware environment where these experiments took place is a cluster consisting of 8 core machines with 32GiB of memory and 16 core machines with 64GiB memory.

All the machines have access to a RAID NFS shared space and dispose of a local disk of 1.7TiB. In all experiments the parallel scorers use two disks for the STXXL vectors (the local disk and NFS). The first set of experiments (in WMT2011) was held on the 8 core machines, while the others were held on the 16 core machines.

Experiments in the WMT2011 In this set of experiments, the Multithreaded version was run on a 16-core machine, whereas the hybrid was run on four different machines (using 4 cores out of 8 on each one). Table 8.1 compares the speed of different tools used in this experiment. The underlying phrase tables are built based on three parallel corpora (merged into a single large corpus): EPPS, NC, and UN. The total number of parallel sentences is 13.8 millions. Clearly, the best choice here is, as should be expected, the hybrid balanced version. It is 7.5 times faster than Moses scorer.

System	Time span
Moses	53h 34m
Multithreaded	28h 49m
Hyb. unbalanced	8h 45m
Hyb. balanced	7h 08m

Table 8.1: Phrase scoring time span in WMT2011

Experiments in the IWSLT2011 Experiments in this context are shown for Moses vs. the multithreaded version for the same corpora as the previous. For every corpus and system, Table 8.2 gives the corresponding time span. As in the previous experiment, the speed up becomes more apparent as the corpus size augments. However, the slight difference (Table 8.2, column +UN compared to Table 8.1) is mainly due to a different set of disks.

System	EPPS+NC	+UN
Moses	11h 23m	49h 34m
Multithr.	9h 34m	27h 44m
Hybrid	27m	-

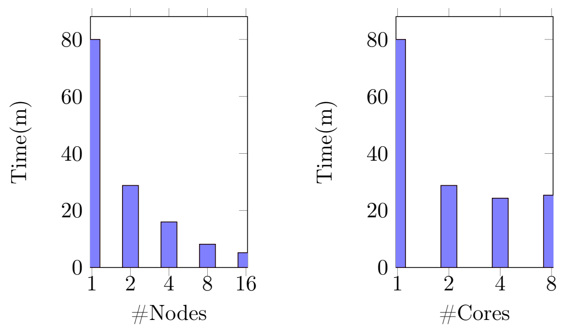
Table 8.2: Phrase scoring in IWSLT2011

Note that the Hybrid variant here, deviates a bit from other settings. It uses 14 nodes, each of which with 8 threads and using only the local disk.

System	Time
Moses	92h 46m
Hybrid	14h 42m

Table 8.3: Phrase scoring in WMT2012

Experiments in the WMT2012 This set of experiments is held between Moses and the hybrid version. In addition to the EPPS, NC, and UN corpora, the training data here includes the Giga corpus as well (resulting in 29.4 millions parallel sentences). Table 8.3 records the time spent in spent in scoring. It is shown here that the hybrid gives a comparable speed up as in the previous setting. i.e. more than 7 times faster.



(a) 2 Cores **(b)** 2 Nodes

Figure 8.3: Effect of the augmenting number of cores and number of nodes

Figure 8.3 compares the effect of number of nodes with the number of cores. In Figure 8.3(a), the number of cores is fixed to 2 while the number of nodes changes from 2 to 16. Whereas, in Figure 8.3(b) the number of nodes is fixed to 2, while the number of cores goes from 2 to 8. In both subfigure, the first bar (labeled “1”) corresponds to Moses time. These two figures confirm our expectations, the bottleneck here is the IO limitations rather than the CPU. The number of CPUs has less effect when compared to the number of nodes. In fact, augmenting the number of nodes means more disks which can be written/read in parallel.

8.6 Conclusions

In this chapter, we presented two methods which make the phrase scoring manageable for extra large corpora. This was achieved by exploiting multiple processing units and parallel disk IOs using the STXXL platform for external memories. The first implementation can be run on a single machine. Whereas the other can be executed in a multinode environment (typically on a cluster of nodes). All these tools depend on the STXXL and OpenMP libraries. In addition to that, the hybrid version assumes the existence of an MPI implementation and the DEMSort extension for the STXXL library.

Given that the bottleneck in this process is the slow disk speeds compared to internal memory, the amount of improvement strongly depends on the number of parallel disks. This could be shown by the experiment in Section 8.5 and in Figure 8.3, where the hybrid version performed much better than the other version. Although its development and maintenance are more complex, the hybrid approach presents two interesting advantages over the shared memory approach. First, by distributing the data over many machines, more memory becomes available for the calculations, since each machine will use its own memory. More importantly, the second advantage is that more parallelism could be achieved on the IO level, because each machine uses independent disk(s).

The main limitation of our methods is the disk space consumption. This is essentially due to the fact that our basic data structure uses a fixed size character string for the keys of our STXXL vectors. As a result, some very long pairs cannot be taken into account and shorter ones have to be filled with blank characters. This implies that a considerable amount of the space allocated for keys is wasted. A possible solution to this would be to use suffix arrays to index the phrases and use only the ID's in the STXXL vector keys.

9

Conclusions and Outlook

The natural language data dispersed in the plethora of documents exposed in the Internet has been and will always remain an invaluable resource for the field of Natural Language Processing (NLP). Its high degree of language and domain diversity, its large quantity, and its small cost are altogether unbeatable features which no other source can offer. However, because its generation is not fully controlled and its collection is automated, some undesirable pieces of data may sneak into the models to inhibit their benefits. A distinguished example of such data is a great part of the training material distributed with the international competitions in Statistical Machine Translation (SMT). The work presented in this thesis focuses on two limiting aspects of the Internet data in the context of SMT. On the one hand, it cuts down the impact of the undesirable pieces on the final models; and on the other hand it boosts the efficiency of extracting these models. In consequence, this work helps translation systems to achieve higher performance in less training time. The approaches developed in the course of this thesis were demonstrated on datasets from WMT and IWSLT evaluation campaigns for French and English languages, where the translation was carried out in both directions.

The work was first concerned with reducing the effect of noise. This was realized through two operations: filtering and weighting. The filtering was studied in Chapters 5 and 6; whereas the weighting was undertaken in Chapter 7. Both approaches judge the cleanliness of a data instance based on an attributed score by a precomputed model. The fundamental difference, though, lies in this model being from another trusted dataset in the filtering scenario, while it is computed from the same training data in weighting. In addition, the filtering discards parts of the data and therefore ends up with a reduced training set, while the weighting does not change the size of the latter.

As for the filtering, we examined two independent types. The first one was covered in Chapter 5 and consists of removing noisy sentence pairs from a parallel corpus. This approach builds on top of, and extends, the work by Munteanu and Marcu (2005). In essence, it is a binary classifier which is trained on a small clean corpus and a bilingual lexicon, in order to distinguish between correct and incorrect pairs. We saw how the quality of the classifier largely depends on the lexicon. We defined a method to build this lexicon from previously aligned data. Estimating the lexicon probabilities from smoothed association measure scores, in lieu of the raw cooccurrences, led to improvements over the maximum likelihood lexicon. It was also perceived that the feature design depends weakly on the identity of the language pair. For roughly similar sized lexicons and similar noise to clean ratio in the clean corpus, the features ended up having similar weights. Therefore, the classifiers of two independent language pairs were interchangeable without significant drop in the performance. This implies that it is no longer necessary for the small clean corpus to be from the same language pair. This is very useful when no clean corpus is available for the language pair under consideration. In terms of BLEU scores, improvements ranging from 0.2 to 0.7 points were obtained by removing 20 to 25% of the parallel pairs.

The other kind of filtering we studied was investigated in Chapter 6 and consisted of in-domain data selection. Data selection removes, from the training data, the fragments which are irrelevant to the task at hand. Such fragments, usually, tend to hinder the model's performance in the corresponding task and thus were considered as noise. The work performed in this direction enhances Moore and Lewis (2010)'s method for data selection. Basically, the method chooses sentences from a large corpus according to their similarity to a given small in-domain corpus. The similarity here is interpreted using language model's cross-entropy. The essential accomplishment here was to show that substantial gains can be attained when the in-domain vocabulary is extended with the synonyms of its corresponding words. This vocabulary extension implicitly expands the coverage of the small in-domain data. This helps to better match the large web data volumes to a specific domain and to discard the unnecessary material. What makes this approach more interesting, however, is that the word synonyms are automatically extracted from available corpora. We extensively studied the subject of extracting semantic associations from corpora and detached it in Chapter 3. The data selection approaches yield an improvement of around 0.5 BLEU points, over the state of the art due to Moore and Lewis (2010).

Our work on the semantic associations in Chapter 3 was developed as a tool for our data selection approaches. We discussed how these semantic relations can be extracted from bilingual and monolingual corpora. While the first resources deliver higher quality associations, the second ensure larger coverage. In the bilingual case, the associations are based on the idea of pivoting to the source language through the target. In the monolingual case, word vectors are exploited to draw the semantic similarities. We proposed many improvements over the state-of-the-art methods which led to around 60% increase in the correlation with manually established associations. Most importantly, we provided an approach to fuse the associations from bilingual corpora into the process of extracting the word vectors from monolingual corpora.

The second approach we used to reduce the noise effect was to weight data instances based on how certain we are about their cleanliness. The operation was inspected in Chapter 7. This operation does not require a clean corpus as input. Instead, it assumes that most of the corpus is clean and the bad data instances are outliers. Then association measures are used to detect these outliers and diminish their contribution to the model accordingly. In language modeling, the weighting was performed on the n-gram and on the sentence levels. In the translation models it was performed on the phrase and on the sentence pair levels. A consequence of this weighting was to formulate a new smoothing technique which supports fractional counts. However, we unveiled this technique earlier in Chapter 4, so that it can be used in all the subsequent other chapters. We used our proposed smoothing also in tasks which could be handled by the traditional smoothing, since it proved beneficial for integral counts as well. A particular novelty about this smoothing is that it explicitly considers different unseen estimators in its definition. In addition, it can be easily cast into Kneser-Ney or Witten-Bell approaches. When interpreted as Kneser-Ney, it gave consistent fair improvements over the commonly used Kneser-Ney smoothing, ranging from 0.8 to 5% reduction in perplexity. Much more important improvements were seen in the Witten-Bell context, the perplexity reduction ranges between 20 and 30%. The weighting itself improves the quality of the models by around 0.5 BLEU points.

The second main concern of the work was training efficiency for large corpora. This was tackled by parallelizing the process in Chapter 8. The parallelization is ensured on the processing unit level and on the IO level. While the former type of parallelization is held using OpenMP and MPI interfaces, the latter builds on top of the STXXL library, developed by Dementiev and Kettner (2005). It turned out that, for SMT training, the severe bottleneck is the IO rather than the processing units. Indeed, the data is usually too large to fit into the main memory, and therefore computing the models leads to massive number of swappings between the memory and the disk. As a result, the speed up obtained in the distributed scenario was significantly more important, since every machine possesses its own physical disk. The increase in speed is mostly dependent on the number of disks which can be written to simultaneously. During our experiments we have seen up to 16 times speed up over the serial implementation. In the evaluation campaign situation, faster training would allow for more extensive explorations of the parameter space. In a real world application, spending shorter time in training is necessary to provide responses for specific requirements in due time.

The weighting approach can be viewed as a generalization of the filtering. Indeed, using binary weights of values 0 for the bad instances and 1 for the good ones, the weighting turns into a filtering. However, the two operations are not exclusive, as they serve different goals. The filtering cleans by comparing to a given reference, while the weighting cleans a corpus by contrasting it with itself. Therefore, the former should be prioritized when applicable as it has stronger connection to the targeted task. A further advantage of the filtering over the weighting is that the former leaves us with less data, requiring less computational resources. On top of this, probably the most attractive feature of the filtering is that it is carried out totally as a preprocessing, and is fully detached from the model estimation; Unlike the weighting, in which the weights have to be taken into account while modeling. This makes the filtering more appropriate to be used in other tasks, such as Neural Machine Translation. On the other hand, having all data instances, in the model, with an adequate weight looks a better alternative for phrase-based SMT. The system might need to resort to those very bad phrases in case it did not find any better alternative. As these will be combined into full translation hypotheses, the decoder will be still able to choose the best amongst them.

The efforts spent in the semantic associations and the smoothing represent an added value to the thesis. They improve the state of the art by a non trivial margin. Moreover, both are very general and susceptible to find wide range of additional applications in the field of NLP, such as information retrieval or word sense disambiguation.

Looking back at the SMT problems caused by the noise and brought up in the Introduction (Chapter 1), we are confident to claim that our methods succeed to reduce their effects considerably. We think our methods are very useful and should be always applied before using the data for training. In particular, the filtering succeeds in removing non-useful parts of the data, reducing thus its size without harming the system's performance. However, one justified argument about this work is that the evaluations are held on relatively clean data. In fact, our testing scenarios use the data distributed in the evaluation campaigns. Such data has already received cleaning efforts and would not be able to show the real strength of our methods. A more realistic choice is to use unprocessed, raw, Internet data. Although this raw data would give more credits to our work, it slightly diverges from our main application scenario which focuses on improving the evaluation campaign systems.

While working on this thesis, some complementary questions arise, which open up interesting directions for future research: First, most of the proposed algorithms were developed and tested in an SMT environment. However, they are general enough to be applicable to other NLP tasks, which would be interesting to investigate.

Second, It would be also interesting to see whether the bootstrapping helps our noise reduction techniques. In fact, the seed data in the filtering can be enlarged using the result of a former filtering. In the weighting, the weighted data can be re-weighted by considering the weighted cooccurrences as input instead of the raw cooccurrences.

Third, In the smoothing, we only considered three simple forms of the discounter, accepting either Kneser-Ney or Witten-Bell arguments. A more complex discounter form combining the two arguments sounds appealing and could yield further improvements.

Fourth, Our implementation of the scoring software naively represents the keys as character strings, which sometimes results in wasted memory space. This can be improved by adopting data structures which are more suitable for string representations such as Tries.

This work was packed in a software framework which we hope to see adopted by other researchers to help them make use of the big amounts of available data on the Web in the best possible way, both for SMT-related tasks and other NLP applications.

Appendices

Appendix A

Derivations of Formulae

A.1 Count re-estimation from clusters

Equation (3.6) can be derived in a similar way to Equation (3.1). First, we start by considering the joint probabilities:

$$\begin{aligned}
\Pr(w, c) &= \sum_{cw} \Pr(w, c \mid cw) \Pr(cw) \\
&\approx \sum_{cw} \Pr(w \mid cw) \Pr(c \mid cw) \Pr(cw) \\
&= \sum_{cw} \Pr(w \mid cw) \Pr(c, cw) \\
&= \sum_{cw} \Pr(w \mid cw) \sum_{cc} \Pr(c, cw \mid cc) \Pr(cc) \\
&\approx \sum_{cw} \Pr(w \mid cw) \sum_{cc} \Pr(c \mid cc) \Pr(cw \mid cc) \Pr(cc) \\
&= \sum_{cw} \Pr(w \mid cw) \sum_{cc} \Pr(c \mid cc) \Pr(cw, cc) \\
&= \sum_{cw} \sum_{cc} \Pr(w \mid cw) \Pr(c \mid cc) \Pr(cw, cc)
\end{aligned} \tag{A.1}$$

The lines including an approximation sign assume conditional independence between the events. The last line switches the position of the inner summation since the term $\Pr(w \mid cw)$ is a constant for this summation. Then the joint probabilities can be cast into association values because the two sides of the equation have the same normalizer, which leads to (3.6).

Appendix B

Examples from Datasets

B.1 Semantic Similarity Datasets

Word1	Word2	Similarity
football	soccer	9.03
type	kind	8.97
street	avenue	8.88
admission	ticket	7.69
tiger	carnivore	7.08
territory	kilometer	5.28
death	row	5.25
seven	series	3.56
president	medal	3
media	gain	2.88

(a) WS-353

Word1	Word2	Similarity
quick	rapid	9.7
cow	cattle	9.52
task	job	8.87
delightful	wonderful	8.65
book	text	6.35
sugar	honey	5.13
day	morning	4.87
ball	costume	2.32
create	destroy	0.63
shrink	grow	0.23

(b) SimLex-999

Word1	Word2	Similarity
beach	sand	48
frozen	ice	47
blossom	buds	46
dinner	eat	41
costumes	outfit	36
green	yellow	34
forest	frog	31
button	flowers	13
cocktail	written	4
bible	misty	3
angel	gasoline	1

(c) MEN

Table B.1: Word pair semantic similarity examples from different datasets

B.2 Word Analogy Datasets

Word1	Word2	Word3	Word4
Nassau	Bahamas	Valletta	Malta
Brussels	Belgium	Doha	Qatar
Irvine	California	Portland	Oregon
Argentina	Argentinean	Greece	Greek
rational	irrational	convenient	inconvenient
code	coding	walk	walking
free	freely	precise	precisely
smart	smarter	quick	quicker

(a) Google analogy dataset

Word1	Word2	Word3	Word4
milder	mildest	trickier	trickiest
richest	rich	steadiest	steady
park's	park	team's	team
days	day	citizens	citizen
raised	raises	saved	saves
keep	kept	have	had

(b) Microsoft analogy dataset

Table B.2: Question examples from different analogy datasets

B.3 Syntactic Similarity Dataset

Word1	Word2	Similarity
codes	books	1
internationally	barely	1
exceeded	shaved	0.88966
operations	generators	0.875047
differ	finance	0.283609
holy	estimated	0.128147
alley	religious	0.0821403
was	concentration	6.36355e-09

Table B.3: Syntactic similarity examples

Appendix C

Example Outputs

C.1 Examples of Semantic Associations

Word	Assoc.
do	1092.71
be	564.493
take	468.471
to	398.912
making	305.91
are	222.02
done	204.74
provide	193.92
ensure	159.40
that	157.09

(a) Raw cooccurrence associations

Word	Assoc.
do	15835.70
take	4067.90
making	3306.80
be	2376.42
done	2172.23
are	1748.79
doing	1505.02
provide	1420.62
bring	1274.00
give	1030.01

(b) LLR associations

Word	Assoc.
do	0.0536
take	0.0213
making	0.0202
doing	0.0129
render	0.0114
bring	0.0113
done	0.0105
formulate	0.0090
give	0.0081
provide	0.0077

(c) Jaccard associations

Word	Assoc.
do	0.0978
render	0.0608
making	0.0503
take	0.0477
formulate	0.0377
doing	0.0345
bring	0.0325
realise	0.0294
provide	0.0261
will	0.0260

(d) GMean associations

Table C.1: Top 10 associated words with "*make*" using different association measures

Appendix D

Additional Results

D.1 Additional Smoothing Results

		English		French	
Unseen	Discounter	WMT-Dev	IWSLT-Dev	WMT-Dev	IWSLT-Dev
TED Corpus					
LOO	KN	379.197	110.476	242.35	222.631
	Hyp	373.896	110.087	237.331	222.605
	Pow	379.301	109.709	240.431	223.054
	Exp	379.783	109.626	240.831	223.203
Chao	Hyp	370.439	109.766	236.163	221.386
	Pow	375.997	109.496	239.198	221.785
	Exp	379.752	109.827	241.299	222.926
Good-Turing	Hyp	364.952	110.398	234.199	221.401
	Pow	368.876	108.915	235.658	219.974
	Exp	372.114	109.014	237.314	220.7
Lind	Hyp	365.504	110.641	234.124	221.997
	Pow	369.08	108.87	235.495	220.24
	Exp	372.269	108.937	237.145	220.953
GIGA Corpus					
LOO	KN	207.39	178.395	150.014	372.999
	Hyp	206.283	177.073	147.049	366.729
	Pow	204.824	175.828	148.099	370.898
	Exp	204.495	175.698	148.272	371.709
Chao	Hyp	205.904	176.739	146.746	364.216
	Pow	204.77	175.852	147.714	368.387
	Exp	205.215	176.361	148.634	371.536
Good-Turing	Hyp	207.333	178.034	146.843	362.298
	Pow	203.933	175.189	146.592	364.538
	Exp	203.987	175.356	147.246	367.139
Lind	Hyp	208.333	178.946	147	362.504
	Pow	203.78	175.061	146.346	364.145
	Exp	203.675	175.083	146.935	366.649

Table D.1: Perplexities of devsets using models computed usin different Kneser-Ney-based smoothing techniques on the TED and GIGA corpora

- Unseen estimators: LOO (leave-one-out); Chao; Good-Turing; Lind (nonparametric-Lindley)
- Discounters: Hyp (hyperbolic); Pow (power); Exp (exponential)

		English		French	
Unseen	Discounter	WMT-Dev	IWSLT-Dev	WMT-Dev	IWSLT-Dev
TED Corpus					
LOO	WB	610.139	154.009	376.078	299.296
	Hyp	433.052	119.244	272.032	235.97
	Pow	421.705	118.976	265.44	234.518
	Exp	417.882	120.043	263.107	235.696
Chao	Hyp	445.897	120.294	279.086	240.169
	Pow	442.073	119.667	276.904	239.159
	Exp	437.369	119.326	274.091	238.528
Good-Turing	Hyp	433.052	119.244	272.032	235.97
	Pow	429.237	118.498	269.765	235.103
	Exp	426.093	118.372	267.811	235.082
Lind	Hyp	435.047	119.484	272.922	237.21
	Pow	431.304	118.669	270.694	236.247
	Exp	427.916	118.5	268.622	236.131
GIGA Corpus					
LOO	WB	293.37	248.396	212.872	531.485
	Hyp	219.374	186.752	159.099	399.237
	Pow	219.626	186.988	157.039	393.409
	Exp	222.095	188.907	157.228	393.323
Chao	Hyp	219.985	187.17	161.016	405.684
	Pow	218.553	186.013	160.284	403.987
	Exp	218.074	185.713	159.735	402.65
Good-Turing	Hyp	219.374	186.752	159.099	399.237
	Pow	217.602	185.23	158.349	397.912
	Exp	217.262	184.987	158.037	397.388
Lind	Hyp	219.53	186.939	158.582	397.718
	Pow	217.498	185.14	157.808	396.536
	Exp	217.179	184.886	157.573	396.278

Table D.2: Perplexities of devsets using models computed usin different Witten-Bell-based smoothing techniques on the TED and GIGA corpora

- Unseen estimators: LOO (leave-one-out); Chao; Good-Turing; Lind (nonparametric-Lindley)
- Discounters: Hyp (hyperbolic); Pow (power); Exp (exponential)

References

Beckmann Andreas, Meyer Ulrich, Sanders Peter, and Singler Johannes. Energy-efficient sorting using solid state disks. *Sustainable Computing: Informatics and Systems*, 1 (2):151–163, 2011.

Lars Arge, Octavian Procopiuc, and Jeffrey Scott Vitter. Implementing I/O-Efficient Data Structures Using TPIE. In *In Proc. European Symposium on Algorithms*. 2002.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011.

Sudipto Banerjee and Anindya Roy. *Linear algebra and matrix analysis for statistics.* Boca Raton, FL: CRC Press, 2014. ISBN 978-1-4200-9538-8.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. Tailoring Continuous Word Representations for Dependency Parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*, 2014. URL `http://aclweb.org/anthology/P/P14/P14-2131.pdf`.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland, June 2014. URL `http://www.aclweb.org/anthology/P/P14/P14-1023`.

Andreas Beckmann, Ulrich Meyer, Peter Sanders Johannes Singler, and Peter Sanders Johannes Singler. Energy-Efficient Fast Sorting 2011, 2012. URL http://sortbenchmark.org/demsort_2011.pdf.

Adam Berger and John Lafferty. Information Retrieval As Statistical Translation. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, New York, NY, USA, 1999. URL http://doi.acm.org/10.1145/312624.312681.

Maximilian Bisani and Hermann Ney. Joint-sequence Models for Grapheme-to-phoneme Conversion. *Speech Commun.*, 50(5):434–451, May 2008. URL http://dx.doi.org/10.1016/j.specom.2008.01.002.

G. Bouma. Normalized (pointwise) mutual information in collocation extraction. In *From Form to Meaning: Processing Texts Automatically, Proceedings of the Biennial GSCL Conference 2009*, volume Normalized, Tübingen, 2009.

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. A Statistical Approach to Machine Translation. *Comput. Linguist.*, 16(2):79–85, june 1990. URL http://dl.acm.org/citation.cfm?id=92858.92860.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Meredith J. Goldsmith, Jan Hajic, Robert L. Mercer, and Surya Mohanty. But Dictionaries Are Data Too. In *Proceedings of the Workshop on Human Language Technology*, HLT '93, Stroudsburg, PA, USA, 1993a. URL http://dx.doi.org/10.3115/1075671.1075716.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Comput. Linguist.*, 19(2):263–311, June 1993b. URL http://dl.acm.org/citation.cfm?id=972470.972474.

Elia Bruni, Nam Khanh Tran, and Marco Baroni. Multimodal Distributional Semantics. *J. Artif. Int. Res.*, 49(1):1–47, January 2014. URL http://dl.acm.org/citation.cfm?id=2655713.2655714.

J. Bunge and M. Fitzpatrick. Estimating the Number of Species: A Review. *Journal of the American Statistical Association*, 88(421):364–73, March 1993. URL http://dx.doi.org/10.2307/2290733.

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 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*, HLT-NAACL '06, Stroudsburg, PA, USA, 2006. URL http://dx.doi.org/10.3115/1220835.1220838.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, Stroudsburg, PA, USA, 2009. URL http://dl.acm.org/citation.cfm?id=1626431.1626433.

Guihong Cao, Jian-Yun Nie, and Jing Bai. Integrating Word Relationships into Language Models. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, New York, NY, USA, 2005. URL http://doi.acm.org/10.1145/1076034.1076086.

M. Carpuat, H. Daumé III, A. Fraser, C. Quirk, F. Braune, A. Clifton, A. Irvine, J. Jagarlamudi, J. Morgan, M. Razmara, A. Tamchyna, K. Henry, and R. Rudinger. *Johns Hopkins Summer Workshop*, chapter Domain Adaptation in Machine Translation: Final Report. 2012.

Anne Chao. Nonparametric Estimation of the Number of Classes in a Population. *Scandinavian Journal of Statistics*, 11(4), 1984. URL http://dx.doi.org/10.2307/4615964.

Barbara Chapman, Gabriele Jost, and Ruud van der Pas. *Using OpenMP: Portable Shared Memory Parallel Programming (Scientific and Engineering Computation)*. The MIT Press, 2007.

Stanley F. Chen and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, Stroudsburg, PA, USA, 1996. URL http://dx.doi.org/10.3115/981863.981904.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999. URL `http://dx.doi.org/10.1006/csla.1999.0128`.

Trevor F. Cox and M.A.A. Cox. *Multidimensional Scaling, Second Edition*. Chapman and Hall/CRC, 2 edition, 2000. ISBN 1584880945. URL `http://www.amazon.com/Multidimensional-Scaling-Second-Trevor-Cox/dp/1584880945`.

Andreas Crauser and Kurt Mehlhorn. LEDA-SM Extending LEDA to Secondary Memory. In *Proceedings of the 3rd International Workshop on Algorithm Engineering*, WAE '99, London, UK, UK, 1999.

James Richard Curran. *From distributional to semantic similarity*. Phd thesis, University of Edinburgh. College of Science and Engineering. School of Informatics., 2004.

Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. Similarity-Based Models of Word Cooccurrence Probabilities. *Machine Learning*, 34(1-3):43–69, 1999. URL `http://dx.doi.org/10.1023/A:1007537716579`.

Hal Daumé. Notes on CG and LM-BFGS Optimization of Logistic Regression. Paper available at `http://pub.hal3.name#daume04cg-bfgs`, implementation available at `http://hal3.name/megam/`, 2004.

III Hal Daumé and Daniel Marcu. Domain Adaptation for Statistical Classifiers. *J. Artif. Int. Res.*, 26(1):101–126, May 2006. URL `http://dl.acm.org/citation.cfm?id=1622559.1622562`.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407, 1990.

R. Dementiev and L. Kettner. STXXL: Standard template library for XXL data sets. In *In: Proc. of ESA 2005. Volume 3669 of LNCS*. 2005.

R. Dementiev, L. Kettner, and P. Sanders. STXXL: standard template library for XXL data sets. *Softw. Pract. Exper.*, 38(6):589–637, May 2008.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011. URL http://dl.acm.org/citation.cfm?id=1953048.2021068.

Ted Dunning. Accurate Methods for the Statistics of Surprise and Coincidence. *Comput. Linguist.*, 19(1):61–74, 1993. URL http://dl.acm.org/citation.cfm?id=972450.972454.

Ted Dunning. *Finding Structure in Text, Genome and Other Symbolic Sequences*. PhD thesis, University of Sheffield, 1998. URL http://arxiv.org/abs/1207.1847.

Nadir Durrani, Barry Haddow, Kenneth Heafield, and Philipp Koehn. Edinburgh's machine translation systems for European language pairs. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, 2013.

Pavlos S. Efraimidis and Paul G. Spirakis. Weighted random sampling with a reservoir. *Information Processing Letters*, 97(5):181– 185, 2006. URL http://www.sciencedirect.com/science/article/pii/S002001900500298X.

Stefan Evert. A Simple LNRE Model for Random Character Sequences. In *Proceedings of the 7èmes Journées Internationales d'Analyse Statistique des Données Textuelles*, 2004a.

Stefan Evert. *The statistics of word cooccurrences.* PhD thesis, University of Stuttgart, 2004b.

Stefan Evert. Corpora and collocations. *Corpus Linguistics. An International Handbook*, 2:223–233, 2008.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing Search in Context: The Concept Revisited. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, New York, NY, USA, 2001. URL http://doi.acm.org/10.1145/371920.372094.

G. Foster, R. Kuhn, and H. Johnson. Phrasetable Smoothing for Statistical Machine Translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, Sydney, Australia, 2006a.

George F. Foster, Roland Kuhn, and Howard Johnson. Phrasetable Smoothing for Statistical Machine Translation. In *EMNLP*, 2006b.

William A. Gale and Kenneth W. Church. A Program for Aligning Sentences in Bilingual Corpora. In *Proceedings of the 29th Annual Meeting on Association for Computational Linguistics*, ACL '91, Stroudsburg, PA, USA, 1991. URL `http://dx.doi.org/10.3115/981344.981367`.

Alberto Gandolfi and C. C. A. Sastri. Nonparametric Estimations about Species Not Observed in a Random Sample. *Milan Journal of Mathematics*, 72(1):81–105, 2004. URL `http://dx.doi.org/10.1007/s00032-004-0031-8`.

Jianfeng Gao, Joshua Goodman, Mingjing Li, and Kai-Fu Lee. Toward a unified approach to statistical language modeling for Chinese. *ACM Transactions on Asian Language Information Processing (TALIP)*, 1(1):3–33, 2002.

Qin Gao and Stephan Vogel. Parallel Implementations of Word Alignment Tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP '08, Stroudsburg, PA, USA, 2008. URL `http://dl.acm.org/citation.cfm?id=1622110.1622119`.

Qin Gao and Stephan Vogel. Training Phrase-Based Machine Translation Models on the CloudOpen Source Machine Translation Toolkit Chaski. *Prague Bull. Math. Linguistics*, 93:37–46, 2010.

M. E. Ghitany and M.E. Al-Awadhi. A Unified Approach to Some Mixed Poisson Distributions. *Tamsui Oxford Journal of Mathematical Sciences*, (17):147–161, 2001.

Irving John Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3 and 4):237–264, 1953.

J.T. Goodman. Putting it all together: language model combination. *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, 3:1647–1650, 2000.

Thanh-Le Ha, Teresa Herrmann, Jan Niehues, Mohammed Mediani, Eunah Cho, Yuqi Zhang, Isabel Slawik, and Alex Waibel. The KIT translation systems for IWSLT 2013. In *Proceedings of IWSLT*, 2013.

Christian Hardmeier. Fast and Extensible Phrase Scoring for Statistical Machine Translation. *Prague Bull. Math. Linguistics*, 93:87–96, 2010.

Felix Hill, Roi Reichart, and Anna Korhonen. SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. *CoRR*, abs/1408.3456, 2014. URL `http://arxiv.org/abs/1408.3456`.

Sabine Hunsicker, Radu Ion, and Dan Stefanescu. Hybrid Parallel Sentence Mining from Comparable Corpora. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation.* 2012.

Nancy Ide, Christiane Fellbaum, Collin Baker, and Rebecca Passonneau. The Manually Annotated Sub-corpus: A Community Resource for and by the People. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, Stroudsburg, PA, USA, 2010. URL `http://dl.acm.org/citation.cfm?id=1858842.1858855`.

J Howard Johnson and Joel Martin. Improving translation quality by discarding most of the phrasetable. In *In Proceedings of EMNLP-CoNLL'07*, 2007.

John G. Kemeny. Mathematics without Numbers. *Daedalus*, 88(4), 1959. URL `http://dx.doi.org/10.2307/20026529`.

Dietrich Klakow. Selecting articles from the language model training corpus. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, volume 3. IEEE, 2000.

R. Kneser and H. Ney. Improved Backing-Off for m-gram Language Modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '95)*, volume I, Detroit, Michigan, USA, 1995.

P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings The Tenth Machine Translation Summit (MT Summit X)*, volume 5, Phuket, Thailand, 2005.

P. Koehn and J. Schroeder. Experiments in Domain Adaptation for Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation (WMT 2007)*, Prague, Czech Republic, 2007.

Philipp Koehn. *Statistical Machine Translation.* Cambridge University Press, New York, NY, USA, 1st edition, 2010. ISBN 0521874157, 9780521874151.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical Phrase-based Translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, Stroudsburg, PA, USA, 2003. URL http://dx.doi.org/10.3115/1073445.1073462.

Philipp Koehn, Amittai Axelrod, Alexandra Birch, Chris Callison-Burch, Miles Osborne, David Talbot, and Michael White. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In *Proceedings of International Workshop on Spoken Language Translation*, 2005. URL http://cis.upenn.edu/~ccb/publications/iwslt05-report.pdf.

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. 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, Stroudsburg, PA, USA, 2007.

Wessel Kraaij and Martijn Spitters. Language Models for Topic Tracking. , *Language Models for Information Retrieval.* Kluwer Academic Publishers, 2003.

John Lafferty and Chengxiang Zhai. Document Language Models, Query Models, and Risk Minimization for Information Retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, New York, NY, USA, 2001. URL http://doi.acm.org/10.1145/383952.383970.

Thomas K Landauer and Susan T. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *PSYCHOLOGICAL REVIEW*, 104(2):211–240, 1997.

Omer Levy, Yoav Goldberg, and Ido Dagan. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *TACL*, 3:211–225, 2015. URL `https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/570`.

Percy Liang, Ben Taskar, and Dan Klein. Alignment by Agreement. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, Stroudsburg, PA, USA, 2006. URL `http://dx.doi.org/10.3115/1220835.1220849`.

J. Lin. Divergence Measures Based on the Shannon Entropy. *IEEE Trans. Inf. Theor.*, 37(1):145–151, September 2006. URL `http://dx.doi.org/10.1109/18.61115`.

Sung-Chien Lin, Chi-Lung Tsai, Lee-Feng Chien, Ker-Jiann Chen, and Lin-Shan Lee. Chinese language model adaptation based on document classification and multiple domain-specific language models. In *Fifth European Conference on Speech Communication and Technology*, 1997.

D. V. Lindley. Fiducial Distributions and Bayes' Theorem. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20(1):102–107, 1958. URL `http://dx.doi.org/10.2307/2983909`.

Wang Ling, Chris Dyer, Alan W. Black, and Isabel Trancoso. Two/Too Simple Adaptations of Word2Vec for Syntax Problems. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, 2015. URL `http://aclweb.org/anthology/N/N15/N15-1142.pdf`.

K. Lund and C. Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods Instruments and Computers*, 28(2): 203–208, 1996. URL `http://scholar.google.de/scholar.bib?q=info:BfG544ylGnkJ:scholar.google.com/&output=citation&hl=de&as_sdt=2000&as_vis=1&ct=citation&cd=0`.

Saab Mansour, Joern Wuebker, and Hermann Ney. Combining translation and language model scoring for domain-specific data filtering. In *Proceedings of IWSLT*, 2011.

Mohammed Mediani, Joshua Winebarger, and Alexander Waibel. Improving In-Domain Data Selection for Small In-Domain Sets. 2014.

I. Dan Melamed. Models of Translational Equivalence Among Words. *Comput. Linguist.*, 26(2):221–249, June 2000. URL `http://dx.doi.org/10.1162/089120100561683`.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013a. URL `http://arxiv.org/abs/1301.3781`.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. , *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013b. URL `http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionali pdf`.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia, June 2013c.

Robert C. Moore. Fast and Accurate Sentence Alignment of Bilingual Corpora. In *Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users*, AMTA '02, London, UK, UK, 2002. URL `http://dl.acm.org/citation.cfm?id=648181.749407`.

Robert C. Moore. Improving IBM Word-alignment Model 1. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA, 2004. URL `http://dx.doi.org/10.3115/1218955.1219021`.

Robert C. Moore and William D. Lewis. Intelligent Selection of Language Model Training Data. In *ACL (Short Papers)*, 2010.

Dragos S. Munteanu and Daniel Marcu. Extracting Parallel Sub-Sentential Fragments from Non-Parallel Corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, 2006. URL `http://www.aclweb.org/anthology/P/P06/P06-1011`.

Dragos Stefan Munteanu. *Exploiting Comparable Corpora*. PhD thesis, Los Angeles, CA, USA, 2006. AAI3257825.

Dragos Stefan Munteanu and Daniel Marcu. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31:477–504, 2005.

Christiane Fellbaum Charles Fillmore Nancy Ide, Collin Baker and Rebecca Passonneau. MASC: the Manually Annotated Sub-Corpus of American English. , *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may 2008. http://www.lrec-conf.org/proceedings/lrec2008/.

H. Ney, U. Essen, and R. Kneser. On Structuring Probabilistic Dependences in Stochastic Language Modelling. *Computer Speech and Language*, 8:1–38, 1994.

Jan Niehues and Stephan Vogel. Discriminative Word Alignment via Alignment Matrix Modeling. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, Stroudsburg, PA, USA, 2008. URL `http://dl.acm.org/citation.cfm?id=1626394.1626397`.

Franz-Josef Och. Maximum-Likelihood-Schätzung von Wortkategorien mit Verfahren der kombinatorischen Optimierung. Bachelor's thesis (Studienarbeit), Friedrich-Alexander-Universität Erlangen-Nürnburg, Germany, 1995.

Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003a.

Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Comput. Linguist.*, 29(1):19–51, 2003b. URL `http://dx.doi.org/10.1162/089120103321337421`.

Tsuyoshi Okita, Yvette Graham, and Andy Way. Gap Between Theory and Practice: Noise Sensitive Word Alignment in Machine Translation. , *WAPA*, volume 11 of *JMLR Proceedings*. 2010.

Michael A. Olson, Keith Bostic, and Margo Seltzer. Berkeley DB. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '99, Berkeley, CA, USA, 1999.

Peter S. Pacheco. *Parallel programming with MPI.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996. ISBN 1-55860-339-5.

K. Papineni, S. Roukos, T. Ward, and W.-jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, Philadelphia, Pennsylvania, 2002.

Pavel Pecina. *Lexical Association Measures: Collocation Extraction*, volume 4 of *Studies in Computational and Theoretical Linguistics.* ÚFAL, Praha, Czechia, 2009. ISBN 978-80-904175-5-7.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, October 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, January 1964. URL `http://dx.doi.org/10.1093/comjnl/7.2.155`.

Mirko Rahn, Peter S, Johannes Singler, and Tim Kieritz. DEMSort-Distributed External Memory Sort, 2009. URL `http://sortbenchmark.org/demsort.pdf`.

Mirko Rahn, Peter Sanders, and Johannes Singler. Scalable Distributed-Memory External Sorting. , *26th IEEE International Conference on Data Engineering, March 1-6, 2010, Long Beach, California, USA*. März 2010.

Philip Resnik and Noah A. Smith. The Web As a Parallel Corpus. *Comput. Linguist.*, 29(3):349–380, September 2003. URL `http://dx.doi.org/10.1162/089120103322711578`.

Xin Rong. word2vec Parameter Learning Explained. *CoRR*, abs/1411.2738, 2014. URL `http://arxiv.org/abs/1411.2738`.

K. Rottmann and S. Vogel. Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-07)*, Skövde, Sweden, 2007.

Peter Sanders and Roman Dementiev. Asynchronous parallel disk sorting. Research Report MPI-I-2003-1-001, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany, February 2003.

Frank Smadja, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. Translating Collocations for Bilingual Lexicons: A Statistical Approach. *Comput. Linguist.*, 22 (1):1–38, March 1996. URL `http://dl.acm.org/citation.cfm?id=234285.234287`.

Dan Stefanescu and Radu Ion. Parallel-Wiki: A Collection of Parallel Sentences Extracted from Wikipedia. *Research in Computing Science*, 70:145–156, 2013.

A. Stolcke. SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference of Spoken Language Processing (ICSLP 2002)*, Denver, Colorado, USA, 2002.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. On the Estimation of Discount Parameters for Language Model Smoothing. In *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy, August 27-31, 2011*, 2011. URL `http://www.isca-speech.org/archive/interspeech_2011/i11_1433.html`.

Yik-Cheung Tam and Tanja Schultz. Correlated Bigram LSA for Unsupervised Language Model Adaptation. In *Advances in Neural Information Processing*

Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008, 2008. URL `http://papers.nips.cc/paper/3564-correlated-bigram-lsa-for-unsupervised-language-model-adaptation`.

Nadi Tomeh, Alexandre Allauzen, and François Yvon. Maximum-entropy word alignment and posterior-based phrase extraction for machine translation. *Machine Translation*, pages 1–38, 2013. URL `http://dx.doi.org/10.1007/s10590-013-9146-4`.

Dániel Varga, Péter Halácsy, András Kornai, Viktor Nagy, László Németh, and Viktor Trón. Parallel corpora for medium density languages. *AMSTERDAM STUDIES IN THE THEORY AND HISTORY OF LINGUISTIC SCIENCE SERIES 4*, 292:247, 2007.

A. Venugopal, A. Zollman, and A. Waibel. Training and Evaluation Error Minimization Rules for Statistical Machine Translation. In *Proceedings of the Workshop on Data-drive Machine Translation and Beyond (WPT-05)*, Ann Arbor, Michigan, USA, 2005.

Jeffrey Scott Vitter. *Algorithms and Data Structures for External Memory*. Now Publishers Inc., Hanover, MA, USA, 2008. ISBN 1601981066, 9781601981066.

S. Vogel. SMT Decoder Dissected: Word Reordering. In *Proceedings of the IEEE International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE 2003)*, Beijing, China, 2003.

I.H. Witten and T.C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4), 1991.

Joern Wuebker, Matthias Huck, Saab Mansour, Markus Freitag, Minwei Feng, Stephan Peitz, Christoph Schmidt, and Hermann Ney. The RWTH Aachen machine translation system for IWSLT 2011. In *Proceedings of IWSLT*, 2011.

Peyton Young. Optimal Voting Rules, 1995.

Hui Zhang and David Chiang. Kneser-Ney Smoothing on Expected Counts. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Baltimore, Maryland, June 2014. URL `http://www.aclweb.org/anthology/P14-1072`.

Yuqi Zhang, Evgeny Matusov, and Hermann Ney. Are Unaligned Words Important for Machine Translation ? In *Conference of the European Association for Machine Translation*, Barcelona, Spain, May 2009. URL `http://www.mt-archive.info/EAMT-2009-Zhang.pdf`.