Domain Adaptation for Translation Models in Statistical Machine Translation

Thesis
Presented to the Faculty of Arts and Social Sciences
of the University of Zurich
for the degree of Doctor of Philosophy

by
Rico Sennrich

Accepted in the Autumn Term 2013
on the Recommendation of the Doctoral Committee:
Prof. Dr. Martin Volk (main advisor)
Prof. Dr. Holger Schwenk
Prof. Dr. Michael Hess

Zurich, 2013

Abstract

We investigate methods to adapt translation models in SMT to a specific target domain. We discuss two major problems, unknown words because of data sparseness in the (in-domain) training data, and ambiguities arising from out-of-domain parallel texts with different domain-specific translations. We propose novel solutions to both problems.

The main contributions of this thesis are as follows:

- We present a novel translation model architecture that supports domain adaptation at decoding time from a vector of component models. The combination is implemented through instance weighting, and all statistics necessary for the computation of translation probabilities are stored in the models.
- We present an architecture to combine multiple MT systems, using techniques and ideas from domain adaptation. The hypotheses by external MT systems are treated as out-of-domain knowledge, and combined with in-domain data through instance weighting.
- We introduce a sentence alignment algorithm that is able to robustly align even noisy parallel texts. We found that higher-quality sentence alignment of the in-domain parallel text has a significant effect on translation quality in our target domain.
- We propose new translation model features that express how flexible, or general, translation units are, in order to prevent translations that only occur in the context of multiword expressions from being overgeneralised.

Wir untersuchen Methoden zur Anpassung von Übersetzungsmodellen in SMÜ an eine bestimmte Zieldomäne. Wir diskutieren zwei Hauptprobleme: spärliche Daten in den Trainingsdaten der Zieldomäne führen zu unbekannten Wörtern, und der Herbeizug von Daten aus Fremddomänen verursacht Mehrdeutigkeiten. Für beide Probleme präsentieren wir neue Lösungsansätze.

Die Hauptbeiträge dieser Dissertation sind folgende:

- Wir präsentieren eine Architektur für Übersetzungsmodelle, welche aus einem Vektor von Teilmodellen besteht und Domänenadaption während der Übersetzung selbst erlaubt. Die Kombination der Teilmodelle wird über eine Gewichtung von Vorkommenshäufigkeiten vollzogen.

- Wir stellen eine Architektur zur Kombination verschiedener Übersetzungssysteme mittels Techniken aus der Domänenadaption vor. Die Hypothesen externer Übersetzungssysteme werden dabei wie Wissen aus einer Fremddomäne behandelt, und mit Daten aus der Zieldomäne kombiniert.

- Wir präsentieren ein Satzalignierungsverfahren, welches auch verrauschte parallele Texte robust auf Satzebene alignieren kann. Durch die Erhöhung der Satzalignierungsqualität erreichen wir eine signifikant bessere Übersetzungsqualität.

- Wir schlagen neue Merkmale für Übersetzungsmodelle vor, welche die Flexibilität von Übersetzungseinheiten ausdrücken, und verhindern, dass inflexible Übersetzungen, welche nur innerhalb eines Mehrwortausdrucks vorkommen, übergeneralisiert werden.

Acknowledgments

First and foremost, I would like to thank my supervisor Martin Volk for giving me the opportunity to pursue a PhD, and for his valuable guidance and unwavering support during the project.

I've had the pleasure to work with and enjoy the company of many colleagues at the institute, and would like to extend my thanks to everybody for the supportive work environment and the fun times at the office, during lunch and coffee breaks. My special thanks go to Sarah Ebling and Annette Rios for providing valuable feedback on drafts of this dissertation, and to the others who were always available for technical discussions and gave feedback on papers that ended up in the dissertation in one way or another, especially my fellow MT researchers Magdalena Plamada, Mark Fishel and Anne Göhring.

I would like to thank Simon Clematide, Michi Amsler and Matthias Fluor for keeping the institute's servers up and running, and bearing with me when I made this harder. My thanks also go to the technical and administrative staff who ensured that there were few things to distract me from doing research.

I want to thank the SMT group in Le Mans, foremost Holger Schwenk, for hosting me for a 3-month research visit, and for the friendly reception and fruitful collaboration during my stay.

Last but not least, I thank my family and friends, simply for being there.

This thesis project was primarily funded by the Swiss National Science Foundation under grant 105215_126999.

Contents

List of Figures

List of Tables

List of Abbreviations

IN	In-domain [training data]
LM	Language model
MBA	Microsoft Bilingual Aligner
MERT	Minimum Error Rate Training
MLE	Maximum Likelihood Estimation
MT	Machine Translation
OCR	Optical character recognition
OOV	Out-of-vocabulary
OUT	out-of-domain [training data]
PRO	Pairwise Ranking Optimization
SMT	Statistical Machine Translation
TM	Translation model (not translation memory)
WMT	Workshop on Statistical Machine Translation

1 Introduction

Statistical Machine Translation (SMT) is data-driven, and SMT systems learn from millions and billions of words of human-translated texts, and texts in the target language. The quality of SMT systems depends heavily on the data that we use for training, not only its quality and amount, but also on how relevant it is for the texts that we wish to translate. While the mantra that "more data is better data" is often true, we can find exceptions to this rule if we also consider the relevance of training data. A corpus of texts from the same domain as the texts that we wish to translate helps us to learn surprisingly good translations, often much better than an equal (or larger) amount of less relevant training data. This thesis proposes novel ways to improve translation quality for a given target domain, focusing on adapting the translation model.

1.1 Problem: Domain-specific Statistical Machine Translation

There are multiple reasons why tailoring Statistical Machine Translation to a specific target domain is important. Every domain comes with its own domain-specific vocabulary, and because languages are very much alive and subject to change, we can never hope to obtain a system with full vocabulary coverage.

Even more common, and a harder problem for general-domain systems, are homonyms and polysemies, words with multiple meanings. In computing, a *mouse* is a mechanical, not an organic entity, and no spiders go hungry when people remove *bugs* from the *web*. While some ambiguities may coincide between languages – German *Maus* also refers to both the animal and the pointing device –, it is a potential cause for mistranslations if they do not. Take the English *bug* for example, which refers to insects, programming errors, or covert listening devices, among others. German *Käfer* refers to insects, but programming errors are simply called *Fehler* (Engl. *error*), or the English term may even be used as a loan word. Listening devices could be referred to as *Abhörgerät* or *Wanze* in German. Choosing the wrong translation variant can result in incomprehension, or even worse, miscomprehension. One example for the latter is *stomach bug*, which is a *Magen-Darm-Entzündung* in German. A literal translation into *Bauchkäfer* could be very misleading: *Bauchkäfer*, or *Käfer im Bauch*, is commonly used as a nickname for unborn babies.

SMT is data-driven and requires human-translated texts to learn translations. These are costly to obtain, considering that typical SMT systems build on millions of sentences of manually translated texts. Luckily, large amounts of texts are already translated for daily communication needs, and can be re-used to train SMT systems.

While training data from the target domain, i.e. from the same domain as the texts which we want to translate, is beneficial for SMT systems, in-domain data is usually relatively scarce, and we have access to a much larger amount of out-of-domain resources. When we want to optimize our system for a specific domain, i.e. perform domain adaptation, a major challenge is how to best combine in-domain and out-of-domain resources, so that additional data complements, but does not dilute the domain-specific translations that we can learn from in-domain data. In this thesis, we address this problem with new methods that not only focus on improving translation quality, but also on reducing the cost of domain adaptation – both the creation and maintenance of adapted systems come at a significant cost in terms of computational and human resources –, and applying it to new problems, i.e. settings in which there is no clear distinction of in-domain and out-of-domain texts, or in which general-domain MT systems take the place of out-of-domain training data. We propose ways to perform domain adaptation quickly, and to support multiple domains in a single translation system.

1.2 Thesis Contributions

- We present a novel translation model architecture with a vector of component models that supports domain adaptation at decoding time. The combination is implemented through instance weighting, and all statistics necessary for the computation of translation probabilities are stored in the models. With this architecture, one translation system can support multiple domains, and we also introduce a method to adapt instance weights to new domains within minutes. We perform an extensive evaluation with other domain adaptation methods to show that instance weighting is a low-maintenance and low-risk form of domain adaptation, and is thus suitable for general deployment.

- We expand the discussion of domain adaptation to settings in which there is no binary split into in-domain and out-of-domain training data. We investigate how different methods scale to 4–10 component models, and propose a method to perform domain adaptation even if there is only a single, heterogeneous training text, using unsupervised clustering of training data and domain adaptation methods.

- We demonstrate that we can conceptualize the combination of translation systems as a domain adaptation problem. Using an in-domain translation system as backbone, and rule-based and online translation systems to generate additional translation hypotheses for the text that we want to translate, we present combination methods that exploit the relative strengths and weaknesses of these systems. The in-domain translation system is most specific, and should be relied on for well-evidenced source words and phrases, but rule-based systems and large-scale online systems are a suitable way to fill in any lexical gaps that arise from data sparseness. We show that such a combination can markedly outperform the best individual system.

- We introduce a sentence alignment algorithm that is able to robustly align even noisy parallel texts. Given the relative impact of in-domain parallel data on translation quality, it is insufficient to rely on already aligned (or easily alignable) parallel texts. We found that higher-quality sentence alignment of the in-domain parallel text has a significant effect on translation quality in our target domain.

- We propose new translation model features that express how flexible, or general, translation units are, in order to prevent translations that only occur in the context of multiword expressions from being overgeneralised. We argue that many multiword expressions are domain-specific, and that these features can improve translation quality if training and test domain do not match.

1.3 Outline

We review the theoretical background of SMT in chapter 2, with a focus on translation modelling and the state of the art of domain adaptation in SMT. In chapter 3, we discuss differences between two text collections, and perform a qualitative analysis to show why data from the same domain is more valuable for SMT training than out-of-domain data, and motivate our experiments in domain adaptation.

In chapter 4, we describe our experimental data and methods, and offer a quantitative analysis of SMT performance with in-domain and out-of-domain data. In chapter 5, we present an MT-based sentence alignment algorithm which is suitable even for noisy parallel texts, and allows us to increase the amount of extractable high-quality in-domain training data. We then proceed to a discussion of how to best mix data from multiple domains into a single model (chapter 6), evaluating existing methods and presenting a novel architecture. In chapter 7, we consider rule-based and other machine translation systems as additional

knowledge sources, and propose methods of integrating these into a domain-specific translation system. We discuss multiword expressions as a source of ambiguities in SMT in chapter 8, and present an algorithm to prevent overgeneralisations of translations learned from (often domain-specific) multiword expressions. We conclude and offer an outlook for future research in chapter 9.

2 Statistical Machine Translation

This chapter gives some background on Statistical Machine Translation. The goal is not exhaustiveness; this would be beyond the scope of this thesis, and good introductions can be found in (Cancedda et al., 2008; Koehn, 2010). Rather, we will focus on algorithms and techniques which are most relevant for our experiments.

2.1 Statistical Models for Machine Translation

The basic idea in Statistical Machine Translation is that we can learn to translate from a corpus of translated text (subsequently called a parallel corpus) by looking at translation frequencies. If a word or sentence in one language is consistently paired with the same word or sentence in the other language, this indicates that the two are good translations of each other. We formalize this expectation that frequent translations are good translations through probabilistic models.

2.1.1 Word-based SMT

The seminal paper by Brown et al. (1993) marks the (re-)birth of Statistical Machine Translation, after the original proposal by Weaver (1949/1955) to use computers to learn translations did not come to fruition in the following decades.

Brown et al. (1993) base their model on Bayes' theorem to estimate the probability that a sentence in the target language T is the translation of a source sentence S (equation 1). The translation problem is the search for the most probable translation $\hat{T}$ for a given S (equation 2). $P(S|T)$ is estimated in a translation model; $P(T)$ can be estimated monolingually by a target-side language model (LM), and expresses the probability that the translation is a sentence in the target language, regardless of the source. $P(S)$ from equation 1 can be ignored, since the source sentence S is given, and its probability is thus constant for all possible translations T and does not affect the search for the best translation $\hat{T}$.

$$P(T|S) = \frac{P(S|T)P(T)}{P(S)} \tag{1}$$

$$\hat{T} = \arg\max_T P(S|T)P(T) \tag{2}$$

The advantage of this decomposition is that we can make simplifying independence assumptions when learning each model. The probability of T is modelled as a Markov chain, specifically an N-gram language model, i.e. assuming that the probability of each word only depends on the $N-1$ preceding words. Brown et al. (1993) model translation probabilities on a word level, the individual word translations being independent from each other. For such a model, the main challenge is to estimate translation probabilities from parallel text without supervision, and the larger part of (Brown et al., 1993) is devoted to the so-called IBM models which word-align parallel text through expectation maximization algorithms. From the aligned parallel text, probability estimates are estimated through Maximum Likelihood Estimation (MLE), which in the unrestricted case corresponds to relative frequency estimation.

2.1.2 Log-Linear Models

A more general version of equation 2 does not simply multiply the model probabilities, but assigns an exponential weight to each (equation 3). In the logarithmic space, we get equation 4.

$$\hat{T} = \arg\max_T P(S|T)^{\lambda_1} \cdot P(T)^{\lambda_2} \tag{3}$$

$$= \arg\max_T \exp(\lambda_1 \log(P(S|T)) + \lambda_2 \log(P(T)))$$

$$= \arg\max_T \lambda_1 \log(P(S|T)) + \lambda_2 \log(P(T)) \tag{4}$$

Equation 4 is a special case of log-linear models (equation 5), with the logarithms of $P(S|T)$ and $P(T)$ as features h_i.

$$\hat{T} = \arg\max_T \sum_{i=1}^{n} \lambda_i h_i(S,T) \tag{5}$$

The log-linear model allows for an inclusion of arbitrary additional features, and given a method of finding the optimal weights λ_i, the model is able to distinguish between useful

and redundant features. The problem of optimizing the weights λ_i is discussed in section 2.3.

State-of-the-art SMT systems no longer only consist of a translation model probability $P(S|T)$ and a language model probability $P(T)$, but use a number of additional features that have been proven useful, such as smoothed and unsmoothed bidirectional translation model probabilities $P(S|T)$ and $P(T|S)$, lexical reordering models (Koehn et al., 2005), and many more (Och et al., 2004; Chiang et al., 2009).

2.2 Phrase-based Translation Models

Word-based translation models make a strong independence assumption, namely that translation probabilities can be estimated on a word-level, ignoring the context that a word occurs in. This assumption does not hold true in natural languages. The translation of a word may depend on its context for morphosyntactic reasons (e.g. agreement within noun phrases), or because it is part of an idiomatic expression that cannot be translated compositionally. Also, some (but not all) translation ambiguities can be disambiguated in a larger context.

Phrase-based translation models improve performance by estimating translation probabilities not only for single words, but also for word sequences.[1] Because our experiments mostly modify phrase-based translation models, subsequently referred to as *phrase tables*, we will give some background on how they are learned, and what additional features they typically contain.

2.2.1 Learning Phrase Translations

Koehn et al. (2003) discuss methods to extract phrase translation pairs from a parallel corpus, and to estimate phrase translation probabilities and other features. They base phrase pair extraction on the symmetrized results of the IBM word alignment algorithms (Brown et al., 1993), and then extract all phrase pairs that are consistent with word alignment (Och et al., 1999), i.e. so that no word in the source phrase is aligned to a word outside the target phrase, and vice versa. The phrase translation probabilities $\phi(\bar{s}|\bar{t})$ are estimated by relative frequency.

Phrase-based models can treat a sequence of words as a single translation unit, but increasing the length of the unit results in data scarcity. Long phrases tend to be less frequent, and many are only observed a few times during training. For such low frequencies, relative frequencies are unreliable probability estimates. Thus, Koehn et al. (2003) propose lexical

[1] The term *phrase* is here used for arbitrary word sequences, and is not linguistically motivated.

weights as an additional feature for phrase pairs. Lexical weights are estimated from the IBM word alignment probabilities, and are less prone to data sparseness than the directly estimated phrase translation probabilities. Foster et al. (2006) introduce more smoothing methods for phrase tables, all aimed at penalizing probability distributions that overfit the training data because of data sparseness.

Johnson et al. (2007) show another way of dealing with spurious phrase pairs. For each phrase pair, they perform the Fisher exact test to measure whether phrase pairs co-occur more often than one would expect by chance to a statistically significant degree. Phrase table size can be reduced by an order of magnitude without losing translation performance – translation quality can even increase – by discarding phrase pairs that do not pass a given significance threshold.

2.3 Discriminative Training

Log-linear models are a powerful framework for SMT because they allow for the inclusion of various useful features in the model. However, they require a method to obtain optimal log-linear weights.

While Och and Ney (2002) used the maximum entropy criterion and generalised iterative scaling, Och (2003) proposes to directly optimize translation quality by minimizing automatic SMT metrics such as word error rate, or maximizing BLEU[2]. The algorithm, minimum error rate training (MERT), approximates the search space by storing the n-best candidate translations given a set of parameters, then iteratively uses the parameters that are optimal for the accumulated n-best candidates to compute a new n-best list and repeating the optimization, until convergence. MERT is a wide-spread and important component of SMT systems, but also a major cause of instability in terms of quality, since it suffers from both local optimum and overfitting effects. Research has been performed to increase the stability of MERT (e.g. Cer et al., 2008; Foster and Kuhn, 2009) or account for its instability during evaluations (Clark et al., 2011).

Alternative discriminative training algorithms for SMT have been proposed in recent years, such as using a margin infused relaxed algorithm (MIRA) (Watanabe et al., 2007) and pairwise ranking optimization (PRO) (Hopkins and May, 2011). Apart from the instability of MERT, these algorithms address the fact that MERT scales poorly to a high number of features.[3] PRO still uses n-best lists for its search space, but instead of Och's (2003) line search, generates ranked pairs of translation hypotheses, and trains a linear classifier to

[2] see section 2.4.

[3] See (Hopkins and May, 2011) for a demonstration in a synthetic experiment.

separate the winners of the pairwise comparisons from the losers. The hyperplane of the classifier is then used as weight vector in the log-linear model. Hopkins and May (2011) have demonstrated that this approach scales better to a high number of features than MERT.

2.4 SMT Evaluation

Any claims as to the effectiveness of SMT and experimental algorithms need to be supported by evidence. A quality metric should ideally fulfill several requirements, and the ISO standard on external metrics (ISO/IEC, 2003) lists several desirable properties for metrics. Ideally, a metric should be reliable (free of random error), objective, give repeatable results, and, most importantly, produce results that are meaningful in regards to quality characteristics that are evaluated.

For this thesis, we use two automatic metrics, BLEU and METEOR, and we will discuss potential problems and how we address them.

2.4.1 BLEU and METEOR

BLEU (Papineni et al., 2002) is the classic automatic metric for machine translation. Its core idea is to use a test set that has been translated by humans, and to consider an automatic translation to be good if it is similar to the human reference translation(s).

For every N-gram in the translation hypothesis (with N typically from 1 to 4), we check if it occurs in one or more of the reference translations. This yields N-gram precision scores, which are averaged out over different N-gram levels by taking the geometric mean. For N-grams occurring multiple times in the translation hypothesis, the number of matches is clipped to its frequency in the reference. BLEU does not consider N-gram recall, but introduces a sentence brevity penalty to penalize translation hypotheses that are shorter than the reference translation.

There are various criticisms that have been levelled at BLEU, for instance its lack of recall or its use of a geometric average, which makes BLEU score 0 if there is no match at the highest N-gram level (Banerjee and Lavie, 2005; Callison-Burch et al., 2006).

METEOR (Banerjee and Lavie, 2005) is a metric that addresses these criticisms by considering both precision and recall, and by explicitly counting crossing alignments between the hypothesis and the reference, rather than using higher-order N-grams to measure fluency. METEOR 1.3 (Denkowski and Lavie, 2011) also supports various non-exact matches between hypothesis and reference: matches on the level of stems, Wordnet synonyms, and with a paraphrase table.

system	sentence
source	We know all too well that the present treaties are inadequate [...]
reference	Uns ist sehr wohl bewusst, daß die geltenden Verträge unzulänglich sind [...]
hypothesis	Wir wissen nur zu gut, dass die gegenwärtigen Verträge nicht ausreichen [...]

Table 2.1: High-quality translation hypothesis with minimal overlap to human reference translation.

However, reference-based metrics in general suffer from the fact that the underlying assumption, namely that the similarity of a translation to a human reference allows conclusions about its quality, need not always be true. There are usually multiple good ways of translating a sentence, which can be very dissimilar on the surface level, as the example in table 2.1 demonstrates.

To evaluate how meaningful automatic translation metrics are, they can be correlated to human evaluations. This is done regularly in the Workshops on Statistical Machine Translation (WMT), where a variety of MT systems are ranked by human judgements (Callison-Burch et al., 2009, 2010, 2011, 2012). In the 2012 Workshop, most automatic metrics achieved a relatively low segment-level Kendall's tau correlation with human judgements (METEOR: 0.25 into English; 0.20 out of English). Note, however, that even the inter- and intra-annotator agreement of the manual rankings is relatively low, with inter-annotator agreement ranging from 0.176 to 0.336 for the different translation directions. This may be due to the fact that ranking translations by quality becomes harder if the quality gap between systems is small, as Callison-Burch et al. (2011) point out. Lopez (2012) considers the problem of producing reliable and repeatable human rankings of machine translation systems unsolved.

On a system level, BLEU obtained a Spearman correlation with human judgements of 0.81, whereas METEOR-1.3 achieved a correlation of 0.83, for translations into English. This indicates that at least on the system level, automatic metrics are indeed meaningful in that a higher score correlates with a higher rank. There are a few famous counter-examples, for instance the fact highlighted by Callison-Burch et al. (2006) that rule-based systems tend to do worse in an automatic evaluation than in a human one. This may be one reason why correlation with human judgements was found to be low for the translation pair EN–DE, where 3 out of the best 4 systems in the 2012 Workshop were rule-based systems. BLEU's correlation with human judgements was found to be 0.22, that of METEOR 0.18. Despite this sobering meta-evaluation of MT metrics, we will use them in our evaluations for lack of better alternatives. Since we mostly discuss systems that are technically similar, we hope that there is no systematic bias in our results that would be contrary to human judgement.

2.4.2 Randomness and Statistical Significance

Clark et al. (2011) discuss three extraneous variables in SMT systems which cause randomness in the results and need to be controlled when evaluating the quality difference between two systems.

One variable is the test set selection. While a test set should be representative of the domain for which we want to evaluate SMT performance, different test sets (or subsets thereof) typically yield different scores due to idiosyncrasies such as sentence length or the number of unknown words. To estimate whether the score difference between two systems can be explained by this random variance, or is indeed caused by the underlying systems, statistical significance tests have been proposed, based on bootstrap resampling (Koehn, 2004) or approximate randomization (Riezler and Maxwell, 2005).

Two other variables that Clark et al. (2011) discuss are related to the optimization of the weights in the log-linear model (see section 2.3). Since the optimization problem is non-convex, and approximations are made during the search, optimizers may find different local optima. Also, different local optima may be overfitted to the development set to different degrees, which results in randomness in the test set score. If we want to make claims about the relative performance of the underlying systems, we want to control these variables. Clark et al. (2011) propose multiple optimizer runs and bootstrap resampling to account for both optimizer and test set instability.

In the experiments, we use MultEval, an implementation of the approach by Clark et al. (2011), for statistical significance testing. Note that a low p-value indicates that the difference between two systems is unlikely to be caused by either test set selection or optimizer instability. A larger test set and a higher number of optimizer runs typically help to reject the null hypothesis, i.e. that a random process causes the difference between two systems. However, passing a statistical significance test does not guarantee that the results are meaningful; it remains true that automatic metrics do not correlate perfectly with human judgements.

2.5 Alternative Translation Models

In the experiments, we will focus mostly on phrase-based translation models. In this section, we will discuss some alternative translation models for SMT.

Some of the experiments that we present in this thesis could be applied to these alternative models. For instance, methods to improve the extraction of training data (chapter 5) and to apply weights to different training data sets (chapter 6) are relatively model-independent. Other aspects, such as a decoding-time recomputation of translation probabilities (sec-

tion 6.6) are less generic, and are unlikely to be applicable to continuous space translation models or conditional random field models, which do not use simple maximum-likelihood estimates for translation probability estimation, but iterative optimization.

2.5.1 Hierarchical and Syntax-based Translation Models

Hierarchical phrase-based models (Chiang, 2005) have been proposed as a way to model discontinuous phrase pairs and reorderings in the translation model instead of requiring a separate distortion model. The model allows *hierarchical phrases* that consist of words (terminals) and subphrases (nonterminals), as in this German-to-English example:

- $X \rightarrow \langle$ hat X_1 gesehen, has seen $X_1 \rangle$

The model is thus a weighted synchronous context-free grammar (CFG), and decoding is performed by CYK parsing. Chiang's hierarchical phrase-based model does not use any linguistically motivated rules, and the hierarchical phrases are learned using similar phrase extraction heuristics as in phrase-based models. However, the formalism can easily be applied to rules learned from a syntactic parser, and Chiang (2010) gives an overview of approaches that use syntactic information on the source or target side, or both.

Hierarchical models outperform phrase-based models in some settings, but not in others. Birch et al. (2009) compare performance of phrase-based and hierarchical models and conclude that their respective performance depends on the type of reorderings necessary for the language pair. Specifically, "phrase-based models account for short-range reorderings better than hierarchical models do, [...] by contrast, hierarchical models clearly outperform phrase-based models when there is significant medium-range reordering, and [...] none of these systems adequately deal with longer range reordering." (Birch et al., 2009)

Excepting phrase-based models, hierarchical models are the only type of translation model with which we experiment in this thesis, namely in chapter 8. While phrase-based, hierarchical and syntax-based models use different types of translation units, model estimation is mathematically similar, and our domain adaptation experiments with phrase-based models (chapter 6) should be easily transferable to hierarchical and syntactic models.

2.5.2 N-Gram Translation Models

N-gram translation models (Mariño et al., 2006) emerged from finite-state modelling. Similar to n-gram language models (which are simply weighted finite-state automata), the probabilities in n-gram translation models are not conditioned on the respective units in the

other language ($p(s|t)$), but on their n-gram history ($p((t,s)_k|(t,s)_{k-1})$). N-gram translation models can thus be seen as "a language model of a sort of 'bilanguage' composed of bilingual units [tuples]" (Mariño et al., 2006, 528). Apart from this difference in probability estimation, the main difference from phrase-based translation models are the extraction heuristics of the translation units, called phrase pairs in phrase-based SMT, tuples in n-gram based SMT (see Crego et al., 2004). One difference (and a shortcoming of n-gram translation models) is that tuple extraction is monotonic, i.e. that the extracted segments must have the same order in both languages, and that reordering phenomena are thus not modelled well. The problem can be alleviated by learning linguistically motivated reordering patterns during training to extend the monotonic search (Crego and Mariño, 2006).

Lavergne et al. (2011) formulate a discriminative variant of n-gram based models, with the tuple probabilities computed through a conditional random field (CRF). Their system did not reach the level of performance of phrase-based or n-gram based systems, and Lavergne et al. (2011) cite their poor reordering model and a sensibility to alignment errors as possible reasons.

2.5.3 Continuous Space Translation Models

In the models discussed so far, translation probabilities are estimated for a discrete set of translations units, using MLE, and possibly some smoothing algorithm. For language modelling, continuous space models have proven a strong alternative (Schwenk, 2007). Recently, several continuous space translation models have been introduced (Le et al., 2012; Schwenk, 2012). The models map the translation units to a continuous representation in a multi-layer neural network, and perform probability estimation in this continuous space. A theoretical motivation for this continuous space representation is its generalisation power to translation units that have not been seen during training.

The continuous space model by Le et al. (2012) is conceptually more similar to n-gram translation models, the one by Schwenk (2012) to phrase-based models. Since the estimation of translation probabilities through neural networks comes at a significant computational cost, the authors employ simplifying modelling assumptions, and/or a two-pass decoding approach, in which a conventional translation model is used to generate an n-best list of translations, which are then reranked through the continuous space model.

2.6 Domain Adaptation in SMT

In chapter 3, we will conduct a qualitative analysis to show some features of domain-specific language. For this overview of related work on domain adaptation techniques in SMT, we will follow a loose definition of domain adaptation, subsuming all methods that try to make better use of the part of the training data that is more similar, and thus more relevant, to the text that is being translated. Some methods are fully unsupervised (e.g. Zhao et al., 2004; Yamamoto and Sumita, 2007), and thus have little control over whether they really adapt their system to different domains, or to some other factor that distinguishes parts of the text. Finch and Sumita (2008) explicitly target one of these factors, namely the distinction between *questions* and *declaratives*, showing that for some language pairs, and with a conversation corpus in which both classes of sentences are frequent, the use of class-specific SMT models improves performance.

2.6.1 Language Model Adaptation

The focus of this thesis is domain adaptation for translation models, but we will shortly discuss some language model adaptation techniques. Language model adaptation is relevant to our work for two reasons. Firstly, we will also apply language model adaptation in our experiments to observe how it interacts with translation model adaptation. Specifically, both translation model adaptation and language model adaptation aim at moving the probability distribution towards favouring in-domain translations in case of ambiguity. Earlier studies reported that adapting both the language and the translation model does not have a cumulative effect (Foster et al., 2010). A second reason is that some language model adaptation techniques can be (and have been) applied for translation model adaptation.

Zhao et al. (2004) use information retrieval techniques to retrieve sentences from a monolingual text collection that are relevant for the current translation hypothesis. A language model is built from these sentences, which is then interpolated with a general background language model.

Foster and Kuhn (2007) investigate mixture-modelling for language models, with component models trained on the individual corpora, then mixed through a linear or log-linear combination. They tested several distance metrics on which to base the optimization of model weights, and propose an expectation–maximization algorithm that minimizes the perplexity of an in-domain development set. Linear interpolation with perplexity minimization has become common practice in SMT (in the WMT 2008 shared task, e.g. Déchelotte et al., 2008; Schwenk et al., 2008; Blackwood et al., 2008). As an alternative to linear interpolation,

the component models can also be added as separate features to the global log-linear SMT model, with weights optimized on BLEU or other MT metrics through MERT/PRO/MIRA (Koehn and Schroeder, 2007).

Moore and Lewis (2010) propose a perplexity-based data selection for out-of-domain monolingual corpora. Given an in-domain and an out-of-domain corpus, their idea is to select sentences from the out-of-domain corpus based on cross-entropy difference between an in-domain and an out-of-domain LM, thus selecting sentences that are more similar to the in-domain corpus than to the out-of-domain corpus.

2.6.2 Translation Model Adaptation

This is a short overview of domain adaptation approaches in SMT that target the translation model. Selected approaches will be discussed in more detail and evaluated in chapter 6. Hildebrand et al. (2005) extend the information retrieval approach by Zhao et al. (2004) to sentence pairs, which are used to train a domain-specific translation model. Yamamoto and Sumita (2007) propose another unsupervised method, which first clusters parallel training data through monolingual measures, and then generates cluster-specific models for each cluster – interpolated with a general model to combat data sparseness. At decoding time, the closest cluster is predicted for each source sentence, and the sentence is translated with the cluster-specific system. We will discuss this method more fully in section 6.5.

Foster and Kuhn (2007), in addition to applying mixture-modelling to language models, do the same with translation models, using interpolation weights optimized on language models. They find that "both TM and LM adaptation are effective", but that "combined LM and TM adaptation is not better than LM adaptation on its own". While this finding has been revised since, with results showing that adapting both models can outperform the adaptation of only one model (Foster et al., 2010), it is conceivable that the two methods are not fully additive, since both aim at moving the probability distribution towards favouring in-domain translations in case of ambiguity.

Bertoldi and Federico (2009) build an adapted translation model from monolingual in-domain data in the target language. They do this by translating the data into the source language, thus creating a synthetic parallel in-domain corpus. This approach is potentially useful if a large amount of monolingual in-domain data is available, but no parallel data. However, the authors also find that a human-translated in-domain corpus is a vastly better training resource than a synthetic one.

Matsoukas et al. (2009) and Foster et al. (2010) propose instance-weighting based on a classifier, assigning weights on the level of sentences or phrase pairs. In Foster et al.

(2010), the instance-weighted out-of-domain corpus is linearly interpolated with an in-domain model.

Some authors investigate data selection for parallel corpora (Yasuda et al., 2008; Axelrod et al., 2011). Parts of the out-of-domain corpus that are similar to the in-domain corpus are selected for training, and the rest is discarded. Note that for parallel corpora, discarding training data may help to learn a probability distribution that better matches the target domain, but also brings the risk of aggravating data sparseness. Banerjee et al. (2012) describe data selection techniques that are tailored to reduce the translation model's out-of-vocabulary (OOV) rate.

Another common method is the extension of the translation model with additional features that indicate whether a phrase pair was learned from in-domain or out-of-domain data (e.g. Niehues and Waibel, 2010; Nakov and Ng, 2009; Daumé III and Jagarlamudi, 2011). Niehues and Waibel (2010) use factored translation models to enrich the translation output with information as to which corpus the phrase pair was learned from. Additional features in the log-linear model then use this information for scoring. Recently, Chen et al. (2013) introduced a feature that expresses the similarity between a phrase pair and an in-domain deveopment set in a vector space model.

Nakov and Ng (2009) perform a deterministic merge of phrase tables, creating a phrase table that consists of all phrase pairs of the original one, plus all phrase pairs from an out-of-domain phrase table that do not occur in the original one. The phrase table is extended by additional features to indicate each phrase pair's origin. Such a deterministic merge was also shown to be effective in (Bisazza et al., 2011; Haddow and Koehn, 2012).

We can also add multiple translation models to the global log-linear SMT model, setting the weights through MERT. Koehn and Schroeder (2007) investigate such a method, but do not score a phrase pair with both translation models. If a phrase pair occurs only in one model, and we define the probability of non-occurring phrase pairs as 0, this would limit the system's phrase table to the intersection of the two components, exacerbating data sparseness. Rather, they exploit a framework by Birch et al. (2007) with alternative decoding paths. During decoding, phrase pairs from each model (with their associated scores) are considered alternative translation options. If a phrase pair occurs in multiple phrase tables, this results in multiple translation options being generated, each using only the features from one model, rather than features from both.

A more in-depth discussion of some domain adaptation techniques, specifically mixture-modelling (linear, log-linear or through instance weighting), phrase table merging, and data selection, will follow in chapter 6. We contribute techniques for weight optimization for instance weighting, unsupervised methods to deal with heterogeneous training or test data,

and novel ways to use mixture-modelling for multi-engine MT (chapter 7). First, we will discuss why domain adaptation is important, and what problems it addresses, on the basis of two sample text collections.

3 Domain-specific Language

It is a well-known fact that the performance of language technology tools depends on the type of text they are applied to. Domain adaptation research aims at improving performance for different types of texts, but often without discussing how these texts differ. We can empirically show that performance depends on whether training data is from similar (in-domain) texts or not, which often is motivation enough. We will do such an empirical evaluation in section 4.2.

In this chapter, we will offer a qualitative analysis of differences between two text collections, the mountaineering texts from the Text+Berg corpus, and Parliamentary Proceedings from the Europarl corpus, with a focus on the translation pair German–French. While we focus on just two example domains, this analysis will nevertheless give some insight into the differences between domains, and the consequences for translation.

We will here follow the terminology by Lee (2001), and use **domain** to refer to the subject matter of the respective texts, but we acknowledge that they also differ in other dimensions, such as the situational context and function of the texts, i.e. their **genre**. As mentioned in section 2.6, most techniques in domain adaptation research do not make such linguistic distinctions, and we will also use the term domain vaguely in the subsequent chapters.

3.1 The Text+Berg Corpus

The Text+Berg corpus is a collection of the yearbooks of the Swiss Alpine Club since 1864 (Volk et al., 2010a). The yearbooks have been published in both German and French since 1957, which provides us with a sizeable amount of parallel data for Statistical Machine Translation. The majority of the texts are originally German and translated into French, but in rare cases, the original is in French (or a third language).

The corpus is thematically homogeneous. The topic of most articles are the mountains, or is closely related to mountains. However, there are various sources of heterogeneity. The diachronic nature of the corpus has an effect on the orthographical and lexical level, since the spelling of words has changed since the 19th century, and new words have been introduced since. Also, the goals of the Swiss Alpine Club have changed over time, which had an effect on the types of texts being published. While the (scientific) exploration of

the Alps was at the foreground in the early years, mountaineering has since become a sport (Bubenhofer and Schröter, 2012).

But even within a single year, there is a wide array of topics. Some examples from the 1911 yearbook illustrate the diversity. There are the typical reports on mountain expeditions: "Klettereien in der Gruppe der Engelhörner" (English: Climbing in the Engelhörner group) or "Aus den Hochregionen des Kaukasus" (English: From the high regions of the Caucasus). But the 1911 book also contains scientific articles on the development of caves ("Über die Entstehung der Beaten- und Balmfluhhöhlen") and on the periodic variations of the Swiss glaciers ("Les variations périodiques des glaciers des Alpes suisses").

3.2 Europarl

Europarl is a collection of Parliamentary Proceedings of the European Parliament (Koehn, 2005). It is well-known in SMT research because the proceedings are currently available in 21 European languages, and parallel data sets can thus be extracted for 210 translation directions.

A peculiarity of the Europarl corpus is that it is actually a transcription of (formal) speech. The corpus is annotated with metatextual information, such as the speaker and original language of the individual speech segments. Also, the high number of original languages entails that, no matter the language pair which we want to translate, the majority of text is originally written in a third language. For the language pair DE–FR, only about 15% of the speech segments are originally German, another 15% originally French, with 70% originally being in a third language. Although the original language and translation direction do influence the text and affect SMT (see Lembersky et al., 2012), we will ignore such effects in our discussion.

3.3 Linguistic Differences between Text+Berg and Europarl

In order to find linguistic differences between the two corpora, we use three search methods:

- find the most frequent words in one corpus that do not occur in the other.
- find words whose probability (relative frequency) is non-zero in both corpora, but markedly different.

German		French	
word	freq.	word	freq.
Mitgliedstaaten (*member states*)	31 300	Mesdames (*Ladies*)	7800
Maßnahmen (*measures*)	16 900	monétaire (*monetary*)	3500
Änderungsantrag (*amendment*)	9400	OMC (*WTO*)	2900
Menschenrechte (*human rights*)	6700	budgets (*budgets*)	2200
schließlich (*eventually*)	5900	troisièmement (*thirdly*)	2000
Ausschuß (*committee*)	5000	Prodi (*Prodi*)	1700
Berichterstatterin (*rapporteur*)	4200	commissaires (*commissaries*)	1700
Binnenmarkt (*internal market*)	3200	codécision (*co-decision*)	1300
Arbeitnehmer (*employee*)	3100	fiscale (*fiscal*)	1300
Rechtsgrundlage (*legal basis*)	1900	ACP (*ACP*)	1300
types (exclusive):	180 000	types (exclusive):	55 000
types (total):	235 000	types (total):	100 000

Table 3.1: Selection of most frequent terms in Europarl corpus that do not occur in Text+Berg.

- find words that occur in both corpora, but that have different translations, measured by the most probable translation in one being improbable (or non-existent) in the other, and vice-versa.

We then qualitatively analyse the top-ranked hits by manually classifying the cause of the lexical divergences, and discussing their consequences. The search operates on a purely lexical level and may not find linguistic differences on other levels. However, we can infer some patterns on the basis of lexical frequencies. For instance, the high frequency of *daß* in Europarl (as compared to Text+Berg) can be traced back to regional spelling differences, namely the fact that in Switzerland, *ss* is used instead of *ß*.

Tables 3.1 and 3.2 show words that are typical of Europarl, those in table 3.1 not occurring in Text+Berg, those in table 3.2 only rarely so. Tables 3.3 and 3.4 do the same for words typical of Text+Berg.[1] Firstly, we can observe that the number of types that both corpora share is relatively low, between 25% (German) and 45% (French). Both corpora have around the same number of German types; for French, the Text+Berg corpus even has 50% more (150 000 as opposed to 100 000). This is especially surprising considering that the parallel part of Text+Berg is around 5 times smaller than Europarl. The main reason for the high number of types in Text+Berg is the fact that the corpus was digitised using OCR software, and many types that are only observed a few times are the result of *OCR errors*. These

[1] All statistics on Text+Berg were computed on the part that is parallel DE–FR.

German		French	
term	ratio	term	ratio
Änderungsanträge (*amendments*)	1880	amendement (*amendment*)	2390
Kommissar (*commissary*)	1470	directive (*directive*)	1460
daß (*that*)	1390	rapporteur (*rapporteur*)	870
muß (*must*)	1080	terrorisme (*terrorism*)	760
Präsidentschaft (*presidency*)	1010	parlement (*parliament*)	680
Parlament (*parliament*)	800	communautaires (*of the community*)	680
Fraktion (*[parliamentary] group*)	720	souhaiterais (*[I] would like*)	550
Terrorismus (*terrorism*)	660	Kosovo (*Kosovo*)	460
Vebraucher (*consumer*)	490	euro (*euro*)	290
Kosovo (*Kosovo*)	430	voté (*voted*)	250

Table 3.2: Selection of terms typical of Europarl. Ratio: (relative frequency in Europarl / relative frequency in Text+Berg).

erroneous types tend to be infrequent, although we do see evidence of them in table 3.4 in the form of **meme* (instead of *même*).

There are a few characteristics that types typical of Europarl and those typical of Text+Berg have in common. These highlight reasons for the fact that a natural language corpus will never achieve full vocabulary coverage. **Compounding** in German is a productive word formation process, and covers the whole frequency spectrum, from the frequent *Menschenrechte* and *Änderungsantrag* in Europarl, *Bergführer* und *Steigeisen* in Text+Berg, to compounds such as *Bleistiftstriche* (Engl: *pencil lines*) and *Gleitschirmtypen* (Engl: *types of paragliders*), which only appear once in Text+Berg. The same is true for other word formation processes such as affixation (***co****décision*, *Berichterstatter****in***) or abbreviations (*OMC*, *SAC*).

Some of the types that are rare or unseen in one corpus demonstrate the **inflectional morphology** of both German and French. French *souhaiterais* (Engl: *[I] would like*) occurs 3000 times in Europarl, but only once in Text+Berg; other types such as *commissaires* and *budgets* do not occur in Text+Berg, whereas their base forms *commissaire* and *budget* do. Other reasons for types being exclusive to one corpus are **regional spelling differences**, e.g. the fact that *ß* is not used in Switzerland, and **named entities** which are only mentioned in one corpus, such as *Prodi*, *Bern* and *Cervin*.

This categorisation highlights *how* new words are formed, but not *why* the vocabulary is so different in the two corpora. Following the terminology by Lee (2001), some of the differences can be attributed to the situational context and function of a text, or its **genre**. Europarl consists of political speeches, which are persuasive in nature and directed to an au-

German		French	
term	freq.	term	freq.
SAC (*S[wiss] A[lpine] C[lub]*)	2500	bivouac (*bivouac*)	980
grosse (*great*)	2410	cordée (*rope team*)	830
Fuss (*foot*)	1610	Piz (*peak*)	820
Bern (*Berne*)	1250	crampons (*climbing irons*)	690
Bergführer (*mountain guide*)	750	ascensions (*ascents*)	670
Piz (*peak*)	740	névé (*névé*)	670
Alpinismus (*alpinisme*)	670	éboulis (*scree*)	630
Steigeisen (*climbing iron[s]*)	620	Cervin (*Matterhorn*)	520
Biwak (*bivouac*)	550	sherpa (*Sherpa*)	460
Sherpa (*sherpa*)	510	bernois (*Bernese*)	440
types (exclusive):	185 000	types (exclusive):	105 000
types (total):	240 000	types (total):	150 000

Table 3.3: Selection of most frequent terms in Text+Berg corpus that do not occur in Europarl.

dience. The phrase *Mesdames et messieurs* (Engl: *Ladies and gentlemen*) occurs in one out of 150 sentences, or 7000 times in total. The phrase does not occur, however, in Text+Berg, which contains texts from other genres, such as travel writing and book reviews, in which a direct address of the audience is atypical.[2] Other types such as *souhaiterais* and *troisièmement* are also rhetorical devices typical of speeches. Most of the types that we found to be very typical of one of the corpora, however, are directly related to the subject matter, or **domain** of the respective texts. Europarl is political in nature, and its topics are *budgets*, legal *amendments*, economic policies affecting the *internal market*, *employees* and *consumers*, and crises that were current at the time of production (*Kosovo* and *terrorism*). The texts in Text+Berg are mostly concerned with mountaineering or issues related to mountain regions in general, such as reports on the culture and natural phenomena in various mountain regions. Among the vocabulary that is specific to or typical of this mountaineering domain, we find mountaineering equipment such as *bivouac*, *Steigeisen* (Engl: *climbing irons*) or *rucksacks*, geological terms such as *falaises* (Engl: *cliffs*), *éboulis* (Engl: *scree*) and *glacier*, and various other terms such as *Bergführer* (Engl: *mountain guide*) and *cordée* (Engl: *rope team*).

For corpus-based machine translation, of which SMT is a subtype, domain-specific vocabulary is one reason why training data is especially relevant when it comes from the same

[2] If it does occur, it is through phrasing that is specific to written texts, such as *chers lecteurs* (Engl: *dear readers*).

German		French	
term	ratio	term	ratio
Hütte (*cabin*)	5470	arête (*ridge*)	20 330
Basislager (*base camp*)	4830	glacier (*glacier*)	17 000
Fels (*rock*)	4240	alpinistes (*alpinists*)	8870
unsern (*our*)	3140	cabane (*cabin*)	6930
Nordwand (*North Face*)	2990	Valais (*Valais*)	3350
Rucksack (*Rucksack*)	2990	gravi (*climbed*)	2920
Füssen (*feet*)	1970	rochers (*rocks*)	2910
senkrecht (*vertical*)	1250	skieur (*skier*)	1300
Eiger (*Eiger*)	1020	*meme (*even*)	1280
Lawinengefahr (*avalanche risk*)	950	falaises (*cliffs*)	1160

Table 3.4: Selection of terms typical of Text+Berg. Ratio: (relative frequency in Text+Berg / relative frequency in Europarl).

domain as the text that is to be translated. Section 4.2 presents out-of-vocabulary rates with in-domain and out-of-domain training data, and shows that they can be reduced faster with in-domain training data than with the same amount of out-of-domain data. However, this alone would not make domain adaptation challenging: simply using all available parallel training data minimizes the unknown word problem.[3]

Tables 3.5 and 3.6 illustrate a second important difference between different domains, namely that even words which are shared by both domains can have different translations in each. The lists have been compiled by searching for words whose most probable translation in one domain-specific translation model is very improbable (or non-existing) in the other model, and vice-versa. We made a selection to highlight domain-specific translations, but there are other reasons why translations may diverge between two corpora. For instance, German *Frau* is translated to *femme* (Engl: *woman*) in Text+Berg, but is predominantly used as a honorific in Europarl (*madame*, Engl: *Mrs.*) Translations may also differ because of spelling differences in the two corpora (*Kenya* vs. *Kenia* and *weiss* vs. *weiß*). Such spelling differences are of little importance for comprehension, and could be normalized away.

Other examples show that the same word may have different meanings in different domains. German *Pass* typically refers to a mountain pass (French: *col*) in Text+Berg, but to a *passport* in Europarl. In other domains, it has additional meanings, such as the pass of the ball in sports (French: *passe*). Words such as *montée* and *appel* are used in the more physical, literal sense of *climb* and *shout* in Text+Berg, and in the metaphorical sense of

[3]The problem can be further reduced by modelling translations on a different level, using lemmatisation, decompounding, or character-based models (Nakov and Tiedemann, 2012).

German	Text+Berg	Europarl
Haushalt	ménage (*household*)	budget (*budget*)
Führer	guide (*guide*)	dirigeant (*leader*)
Ziel	but (*destination*)	objectif (*objective*)
Höhe	altitude (*altitude*)	niveau (*level*)
Pass	col (*mountain pass*)	passeport (*passport*)
Ebene	plaine (*plain*)	niveau (*level*)
Vertrag	contrat (*contract*)	traité (*treaty*)
steigen	monter (*increase/climb*)	augmenter (*increase*)
Sitten	Sion	moeurs (*manners*)
Steuer	volant (*wheel*)	taxe (*tax*)

Table 3.5: Selection of German words for which the most probable translation in Text+Berg has a low probability in Europarl, and vice-versa.

French	Text+Berg	Europarl
temps	Wetter (*weather*)	Zeit (*time*)
langue	Zunge (*tongue*)	Sprache (*language*)
régime	Diät (*diet*)	Regime (*regime*)
montée	Aufstieg (*climb*)	Zunahme (*increase*)
appel	Ruf (*shout*)	Appell (*appeal*)
faces	Wände (*[mountain] faces*)	Seiten (*sides*)
Kenya	Kenya (*Kenya*)	Kenia (*Kenya*)
branches	Äste (*branches [of a tree]*)	Teile (*branches [of an organization]*)
héroïne	Heldin (*heroine*)	Heroin (*heroin*)
sais	weiss (*[I] know*)	weiß (*[I] know*)

Table 3.6: Selection of French words for which the most probable translation in Text+Berg has a low probability in Europarl, and vice-versa.

increase and *appeal* in Europarl. A stark contrast are homonyms such as *Steuer*, which can mean *wheel* and *tax*, the former translation being predominant in Text+Berg, the latter in Europarl.

These ambiguities add a new problem for SMT. For the same source word, we may learn starkly different probability distributions from different corpora. We expect that the probability distribution that we learn from in-domain data has a higher fitness[4] for the domain, except for rare phrases, for which probability estimates are usually poor. This problem, which we will refer to as the *ambiguity problem*, requires different strategies than the aforementioned *unknown word problem*. While adding more training data is a good strategy

[4]We borrow this term from early evolutionary biology to emphasize that the question in domain adaptation is not how "good" or "bad" data or a model is, but how well-adapted it is to the task at hand.

to reduce data sparseness and the unknown word problem, adding out-of-domain data can actually make the ambiguity problem worse, i.e. result in a model that resolves fewer ambiguities successfully.

These conflicting effects have a number of consequences. Firstly, they make it hard to predict the utility of out-of-domain parallel training data for a given task. Out-of-domain data may reduce data sparseness and thus improve translation performance, but also brings the risk of aggravating the ambiguity problem and hurting the system. Depending on the extent of each problem, the net effect of adding more out-of-domain data may be positive or negative. Secondly, a simple concatenation of training data gives little control over which translation is preferred in case of ambiguity, and we want to exploit out-of-domain data to reduce data sparseness, but prefer the in-domain probability distribution for known words.

Section 4.2 provides more empirical evidence of the unknown word problem and the ambiguity problem. These two problems are the main motivation for the experiments in this thesis.

4 Building a Domain-specific SMT system

4.1 Experimental Data and Model Configurations

Most of the experiments in the following chapters were conducted within the Swiss National Science Foundation research project "Domain-specific Statistical Machine Translation", which uses the Alpine domain (and thus the Text+Berg corpus) as target domain. However, various out-of-domain data sets were used in different experiments. We give a short description of commonly used data sets here, and will later refer to the data sets and/or models by the names given here, rather than re-introducing them for every experiment.

4.1.1 Corpora

Table 4.1 shows a list of corpora used, along with a short description and their respective size.[1] Corpus size has been measured after tokenization and clean-up.

For experiments in the Text+Berg domain, we typically use *SAC_145_dev* (1424 sentences) as development set for MERT, PRO and other optimizers, and *SAC_test* (991 sentences) as evaluation set. Both are held-out from training, and the latter has been extracted from the Smultron 3.0 treebank (Volk et al., 2010b). The news-test20** data sets, which are used in some experiments as development and test sets, are from the WMT shared tasks.

[1] OpenSubtitles is based on documents from `http://www.opensubtitles.org`

name	description	language(s)	sentences	tokens (L1)	tokens (L2)
Europarl	version 5 (de–fr) / 6 of Europarl corpus (Koehn, 2005)	de–fr	1 495 000	37 850 000	44 230 000
		de–en	1 707 000	43 740 000	46 030 000
		fr–en	1 785 000	52 740 000	47 560 000
		de	1 986 000	48 720 000	
		en	2 032 000	54 810 000	
		fr	2 002 000	59 630 000	
JRC-Acquis	EU law texts (Steinberger et al., 2006)	de–en	1 202 000	23 560 000	25 807 000
		de–fr	1 166 000	20 700 000	23 980 000
OpenSubtitles	film subtitle corpus (Tiedemann, 2009)	de–en	4 649 000	33 270 000	35 380 000
		de–fr	2 334 000	18 400 000	18 330 000
News	News Crawl corpus, part of WMT 2011 training data	de	51 670 000	915 000 000	
		en	113 100 000	2 655 000 000	
		fr	24 980 000	616 500 000	
News-commentary	version 6 of news commentary corpus, part of WMT 2011 training data	de–en	135 600	3 375 000	3 289 000
		fr–en	114 600	3 303 000	2 823 000
		de	162 100	3 875 000	
		en	180 900	4 323 000	
		fr	147 500	4 174 000	
SAC_145	release 145 of Text+Berg corpus (collected publications of **S**wiss **A**lpine **C**lub) described in chapter 3.1	de–fr	219 200	4 179 000	4 749 000
		de	1 149 000	21 610 000	
		fr	661 900	13 190 000	
UN–Giga	UN and 10^9 corpus from WMT 2011	fr–en	33 180 000	1 016 000 000	850 300 000

Table 4.1: Parallel and monolingual training data used for SMT experiments.

4.1.2 Tools and Models

Generally, we use openly available tools and commonly used settings to create the experimental SMT systems. Specifically, we follow the instructions of the 2011 Workshop on Statistical Machine Translation (WMT 2011)[2] unless specified otherwise. We will give a short summary of the steps here.

Data Preparation

Training data and translation sets are tokenized and lowercased. To produce cased output, we would train and apply a recaser, but we left out this step for the experiments, reporting case-insensitive scores instead. For translation model training, we perform length-based filtering, discarding sentence pairs which contain empty sentences, sentences longer than 80 tokens, and pairs with a large divergence in length (by a factor of 9 or more).

Language Models

We build all language models with SRILM (Stolcke, 2002), using interpolated modified Kneser-Ney smoothing (Chen and Goodman, 1998) and n-gram length 5. We use closed-vocabulary models, or, where this is not possible (e.g. when using KenLM (Heafield, 2011) for decoding), a negative log probability of -100 for unknown words.[3]

To avoid redundancy, we describe a few language models that are used in multiple experiments:

- SAC_WMT11_interpolate_[de,fr]: linear interpolation of language models trained on *SAC_145*, *Europarl*, *News-Commentary* and *News*. Interpolation coefficients set through cross-entropy minimization on *SAC_145_dev* with SRILM.
- WMT11_interpolate_[de,en]: linear interpolation of language models trained on *Europarl*, *News-Commentary* and *News*. Interpolation coefficients set through cross-entropy minimization on *news-test2008* with SRILM.

Translation Models

We train 7-gram phrase-based translation models with Moses (Koehn et al., 2007), using MGIZA++ (Gao and Vogel, 2008) for word alignment, which implements the IBM models.

[2] `http://www.statmt.org/wmt11/baseline.html`

[3] This is not relevant if the language model is trained on a superset of the translation model training data; this was not the case in all experiments, however.

Phrase pairs are extracted with the heuristic *grow-diag-final-and*. Additionally, we prune phrase tables based on statistical significance tests (Johnson et al., 2007). The idea is to collect the phrase co-occurrence statistics from the parallel corpus, and, through the Fisher exact test, determine whether the co-occurrence frequency is to be expected by chance or not. All phrase pairs that fall below a pre-defined threshold are discarded, the threshold being set so that all singleton phrase pairs also fall below it.

The parameters of the global log-linear model are set through Minimum Error Rate Training (MERT) on a development set (Och, 2003).

Evaluation

As discussed in section 2.4, we report two translation measures: BLEU (Papineni et al., 2002) and METEOR 1.3 (Denkowski and Lavie, 2011). All results are lowercased and tokenized, measured with five independent runs of MERT (Och and Ney, 2003) and MultEval (Clark et al., 2011) for resampling and significance testing.

If we report differences in score to be significant, we refer to the significance tests by Clark et al. (2011), which aim at controlling for optimizer instability. However, differences which are significant under their criterion, i.e. which are unlikely to be caused by a random process, may still not be meaningful. We typically evaluate approaches on multiple data sets, and point out if the results are inconsistent.

4.2 SMT Learning Curves: How Important is In-domain Data?

On the data side, there are two simple ways to improve translation quality: get more data, and get data that is more relevant for the translation task (e.g. in-domain rather than out-of-domain data). In isolation, both effects are well-described. Koehn (2002) shows the learning curve of SMT systems on Europarl data to be logarithmic, i.e. that every doubling of the amount of training data led to a (roughly) constant increase in translation quality. There has been more work on learning curves in SMT (e.g. Turchi et al., 2009; Kolachina et al., 2012), and while Kolachina et al. (2012) find functions that fit the learning curve better than a logarithmic curve, it is a reasonable estimate.

However, adding training data does not guarantee an increase in performance. Koehn (2002, 15) has remarked that small in-domain corpora perform better when translating material from the same domain than larger, out-of-domain corpora. He found that adding out-of-domain data to the in-domain system even decreased system performance. Other studies made similar findings (Banerjee et al., 2010; Ceauşu et al., 2011)

There are questions pertaining to the creation of SMT systems for which there is conflicting evidence. On the one hand, our knowledge about the log-linear learning curve of SMT systems suggests that adding a mere 1000 sentences will hardly improve an SMT system trained on 1 000 000 sentences. On the other hand, we expect different results when combining a small amount of in-domain data with a large amount of out-of-domain system than when the amounts are reversed. To shed light on the effects of in-domain and out-of-domain training data, we present some learning curves that illustrate the respective contribution of in-domain and out-of-domain training texts.

We have created the data points by training SMT systems with varying amounts of parallel training data. The language model is the same for all systems, *SAC_WMT11_interpolate*.[4] The training set consists of a simple concatenation of 2^x sentences from *SAC_145* and 2^y sentences from *Europarl*, with $10 \leq x \leq 17 \vee x = 0$ and $10 \leq y \leq 20 \vee y = 0$. We use a test set from the Text+Berg corpus (*SAC_test*), so *SAC_145* is considered in-domain (or IN), *Europarl* out-of-domain (OUT) for this experiment. For each data point, MERT is performed on *SAC_145_dev*. x and y are two independent variables, but we illustrate the results as a two-dimensional plot: BLEU scores as a function of the number of training set sentences ($2^x + 2^y$). To which extent a system is trained on in-domain data is expressed through a third value $\frac{x}{x+y}$, which is represented by the shade of each data point. It ranges from 0 (white), which denotes an out-of-domain system, to 1 (black) for an in-domain system.

Figure 4.1 shows the learning curve in the translation direction DE–FR in a logarithmic scale, with BLEU score as a function of training set size. In the appendix, figure 10.1 (p. 147) shows the same for the translation direction FR–DE. We will focus the discussion on the results from figure 4.1, which are shown numerically in table 4.2. We observe that systems trained on purely in-domain data form the upper bound of the learning curve, systems trained on purely out-of-domain data the lower bound; systems trained on a combination of both fall in between.[5] With this training and test data, adding out-of-domain data does not lead to a decline in translation quality. However, the more in-domain data is used for training, the flatter the learning curve for out-of-domain data becomes. Adding 2^{20} sentences of out-of-domain training data improves BLEU scores by 10.1 points if there is no in-domain data; by 1.8 points with 2^{15} sentences in-domain data; and by 1 point with 2^{17} in-domain sentences. We can expect this downward trend to continue. That negative learning curves are possible

[4] Since the collection of monolingual training data is easier by far, using a constant, large language model is a realistic scenario. Experiments with an out-of-domain language model yielded a similar learning curve, although on a lower level of performance. See figures 10.2 (p. 148) and 10.3 (p. 148).

[5] This is true when considering BLEU as a function of training set size. Since the amount of in-domain data is limited, the globally best system may well use a mix of in-domain and out-of-domain data.

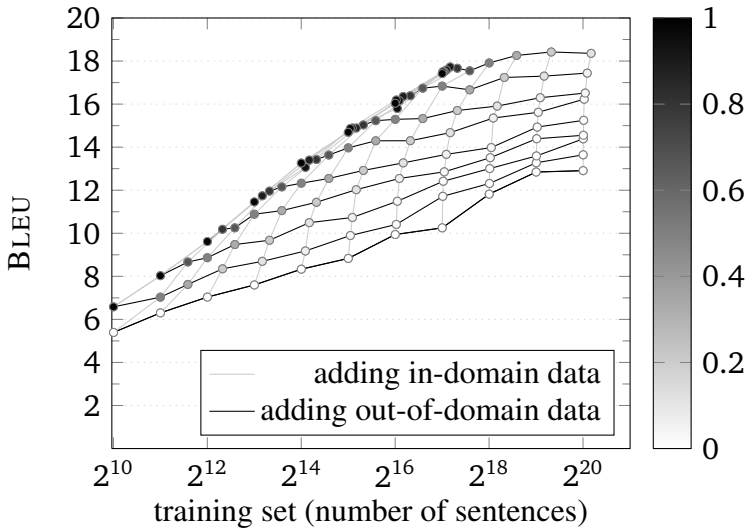

Figure 4.1: BLEU score as a function of training set size, illustrating the relative effect of adding in-domain (IN) and out-of-domain (OUT) training data. Data point colour denotes the relative amount of IN data. Translation direction DE–FR.

has been described in earlier studies (see Koehn, 2002), and indeed, we observe a negative learning curve in different settings (e.g. figure 10.3, p. 148).

A pleasant surprise is the steep performance improvement possible with in-domain data. Even the smallest number of sentences we tested (2^{10}) has a significant effect on translation performance – for all tested amounts of out-of-domain data. Specifically, the performance delta is 3.8 BLEU points with no out-of-domain-data, 1.1 BLEU points with 2^{15} out-of-domain sentences, and 0.7 with 2^{20} out-of-domain sentences. In other words, even for a system trained on more than a million sentence pairs, adding 1000 sentences (or 0.1%) of data can indeed be sufficient to markedly improve translation quality.

One explanation of the unequal contribution of in-domain and out-of-domain data is data sparseness. In section 3, we have seen that the two corpora have different vocabularies, and we can expect that, with equal amounts of training data, the in-domain system will reach a better lexical coverage of the test set than the out-of-domain system. We use the OOV rate on the test set as an indicator of data sparseness. The OOV rate does not capture all data sparseness problems, since it does not consider multi-word phrases and the frequency of phrases.[6] Still, we consider the OOV rate to be a useful indicator of data sparseness.

[6]For rare words, translation performance suffers from spurious alignment and probability estimation (see Koehn and Knight, 2001).

IN	OUT											
	2^0	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}	2^{20}
2^0	2.8	5.4	6.3	7.0	7.6	8.3	8.8	9.9	10.3	11.8	12.8	12.9
2^{10}	6.6	7.0	7.6	8.3	8.7	9.2	9.9	10.4	11.7	12.3	13.3	13.6
2^{11}	8.0	8.7	8.9	9.5	9.7	10.5	10.7	11.5	12.4	13.0	13.6	14.4
2^{12}	9.6	10.2	10.3	10.9	11.1	11.4	12.0	12.5	12.9	13.5	14.4	14.6
2^{13}	11.5	11.7	12.0	12.2	12.3	12.5	12.9	13.3	13.7	14.0	14.9	15.2
2^{14}	13.3	13.1	13.4	13.4	13.6	14.0	14.3	14.3	14.7	15.4	15.6	16.2
2^{15}	14.7	14.9	14.9	14.9	15.0	15.2	15.3	15.3	15.7	15.9	16.3	16.5
2^{16}	16.0	16.2	15.8	16.2	16.4	16.4	16.7	16.8	16.7	17.2	17.3	17.4
2^{17}	17.4	17.5	17.5	17.5	17.6	17.7	17.7	17.5	17.9	18.3	18.4	18.4

Table 4.2: BLEU score for different combinations of training data. Translation direction DE–FR.

Table 4.3 shows the OOV rate of each system. The table confirms that the in-domain systems have a higher lexical coverage than the out-of-domain systems. With 2^{17} sentences of training data, the OOV rate is 9.8% for the in-domain system and 22.8% for the out-of-domain system. Another way to look at it is that we require 32 times fewer sentences (2^{15} rather than 2^{20}) to achieve the same OOV rate with an in-domain system than with an out-of-domain system. This is in line with the fact that the two training corpora have starkly different vocabularies, with a vocabulary overlap of only 25–55% (see tables 3.1, p. 37 and 3.3, p. 39).

Generally, OOV rate and BLEU score are inversely proportional. We measure a highly negative Spearman rank correlation of -0.989 between OOV rate and BLEU score.[7] This in itself is unremarkable, since we expect that adding training data both lowers the out-of-vocabulary rate and increases BLEU scores. However, the data also shows that, even in this controlled setting, the steeper learning curve of in-domain systems cannot be fully explained by their faster decrease in OOV rate. A system trained on 2^{15} in-domain sentences performs 5.9 BLEU points better than a system trained on the same number of out-of-domain sentences. If we instead compare it to an out-of-domain system with a similar OOV rate, which is a system trained on 2^{20} sentences, IN still outperforms OUT by 1.8 BLEU points.

Figure 4.2, in which translation performance is shown as a function of the OOV rate, illustrates the phenomenon that IN systems achieve higher BLEU scores than OUT systems with comparable OOV rates.[8] This gap must be explained by other factors than the

[7] Of course, the correlation is specific for this test setting, and we do not imply that it will hold true in different settings, for instance when comparing different language pairs or modelling techniques.

[8] In high-sparsity settings, the two types of systems converge, with the extremum being 100% OOV rate and a BLEU score of 2.8. The floor of BLEU score is not 0 because some word sequences (names, numbers, punctuation marks) are identical in both languages.

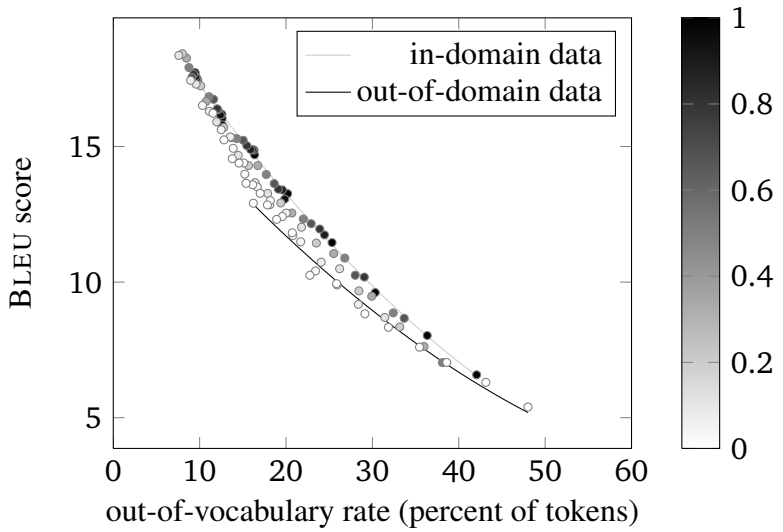

Figure 4.2: BLEU score as a function of out-of-vocabulary rate (tokens). Data point colour denotes the relative amount of in-domain data. Curves are estimated using least-squares polynomial fitting (second degree). Translation direction DE–FR.

IN	OUT											
	2^{0}	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}	2^{20}
2^{0}	100.0	48.0	43.1	38.6	35.5	31.9	29.1	25.8	22.8	20.7	17.8	16.2
2^{10}	42.1	38.1	36.0	33.2	31.4	28.4	25.9	23.4	20.8	18.9	17.0	15.4
2^{11}	36.3	33.7	32.4	29.9	28.4	26.2	24.1	21.7	19.6	18.2	16.1	14.6
2^{12}	30.3	29.1	28.0	26.8	25.5	23.5	21.8	20.0	18.1	16.7	15.1	13.8
2^{13}	25.3	24.5	23.9	22.9	22.0	20.7	19.4	17.9	16.4	15.2	13.9	12.8
2^{14}	20.2	19.9	19.6	19.1	18.6	17.7	16.7	15.7	14.5	13.5	12.5	11.5
2^{15}	16.4	16.3	16.1	15.9	15.5	15.0	14.3	13.7	12.8	12.0	11.1	10.3
2^{16}	12.6	12.6	12.6	12.4	12.1	12.0	11.6	11.1	10.8	10.1	9.6	9.0
2^{17}	9.8	9.7	9.7	9.7	9.6	9.5	9.3	9.1	8.8	8.5	8.0	7.6

Table 4.3: OOV rate (tokens) for different combinations of training data. Translation direction DE–FR.

unknown-word problem, whose extent is expressed by the OOV rate. One issue in domain adaptation that we identified in section 3 is the ambiguity problem. Even if a word is known by both the in-domain and the out-of-domain system, we expect that the in-domain system is more likely to produce a translation that is appropriate for the domain (see tables 3.5 (p. 41) and 3.6 (p. 41)).

The interaction between the unknown-word problem and the ambiguity problem explains why learning curves for a mix of in-domain and out-of-domain data can be so unpredictable, and even negative. Out-of-domain data helps to mitigate the unknown-word problem, which improves translation quality. At the same time, out-of-domain data exacerbates the ambiguity problem. Depending on how serious data sparseness and the ambiguity problem are at a given point, adding out-of-domain data may either improve performance (if the benefit of reducing data sparseness outweighs the negative effect of introducing translations that are unfit for the domain), or decrease it.[9] Adding in-domain data has no such drawback: it both mitigates the unknown word problem and the ambiguity problem.

In summary, the most important findings of this analysis of SMT learning curves are the following:

- In-domain data allows for relatively steep learning curves, even when adding it to a system trained on large amounts of out-of-domain data.
- Data sparseness is an important reason why in-domain data is more valuable than out-of-domain data. With the same training set size, in-domain data sets result in better lexical coverage than out-of-domain ones.
- Data sparseness is not the only reason why in-domain data is more valuable than out-of-domain data. The probability distribution learned from in-domain data is better fitted to the domain than the one learned from out-of-domain data.
- Whether adding out-of-domain data can improve SMT performance is unpredictable, and depends on the relative effects of the reduction in data sparseness and the increase in ambiguity. Adding out-of-domain data to systems trained on large amounts of in-domain data may lead to very flat training curves (figure 4.1) or even negative ones (figure 10.3, p. 148).

One conclusion that we can draw from these findings is that in-domain data is very valuable, and efforts to better exploit in-domain corpora can be worthwhile even if the potential

[9] Haddow and Koehn (2012) report experimental evidence that support this distinction. They found that adding out-of-domain phrase pairs that are unknown in the in-domain translation model improves SMT performance, while re-scoring phrase pairs with additional out-of-domain data is harmful.

amount of in-domain data looks small compared to existing out-of-domain corpora. We will discuss such efforts in section 5. Also, the advantages and disadvantages of out-of-domain data have implications for the question of how to best integrate out-of-domain data, which we will discuss in section 6. Without starting this discussion now, we want to emphasize that a simple concatenation of data sets, as performed in this chapter, is an unsatisfactory technique. Since the effect of adding out-of-domain data is unpredictable and may even be negative, the creator of an SMT system either has to accept this risk of performance degradation, or test different combinations of data sets and use the best one. This is a time-consuming process from which we learn little – if adding a data set decreases performance, we cannot conclude that it is bad; it simply has a lower fitness for the tested target domain than other data sets. Ideally, a domain adaptation technique should make such a trial-and-error combination of data sets superfluous.

4.3 Summary

After introducing experimental tools and methods, we perform first experiments with the concatenation of in-domain and out-of-domain data. We show empirical evidence for the importance of in-domain training data, and that tiny amounts of in-domain data (compared to the amount of out-of-domain training data) can significantly improve translation quality. Conversely, we show that out-of-domain training data yields much smaller improvements, and sometimes even results in quality degradation. We discuss the relationship between OOV rate and translation quality, and show that higher vocabulary coverage is an important, but not the only reason for the superiority of in-domain training data. The importance of in-domain data, and the unpredictable benefits from adding out-of-domain data, partially motivate our experimental research in the following chapters.

5 Improving Data Collection: Sentence Alignment

The previous chapter has shown how sensitive SMT systems are to the domain of the (parallel) training data, and how important it is to use in-domain training data. This means that large, freely available parallel corpora such as Europarl and the UN corpus, which cover only specific domains, are unsatisfactory for many SMT purposes. Collecting parallel resources that are tailored to the translation task at hand, and training a domain-specific system on these resources, may be the most effective way of improving SMT performance relative to a general-purpose system. As a consequence, it is important to have robust tools for the preparation of new parallel corpora.

Sentence alignment, the task of identifying which sentences correspond to each other in a text and its translation, is a relatively easy task in well-structured text collections such as Europarl. Europarl is marked-up with structural information such as paragraph and speaker information, and paragraph and speaker changes serve as anchors in sentence alignment. Sentence alignments are not allowed to cross these boundaries, which means that a sentence alignment algorithm only has to process small units of texts, and any errors do not propagate across paragraph boundaries. Koehn (2005) reports that a paragraph in Europarl typically consists of only 2–5 sentences, and that this leads to a "very high" alignment quality. Figure 5.1 shows an excerpt of Europarl, annotated with manual alignments, structural anchors, and showing the beginning and length (in words) of each sentence. Note that most alignments in Europarl are trivial. Manning and Schütze (1999) report that typically, approximately 90% of sentence alignments are 1-to-1, with the remaining 10% being cases in which the translators merge, split, insert or delete sentences during translation.

For other text collections, sentence alignment is significantly harder. In the Text+Berg corpus, there are various complicating factors for sentence alignment. Since it is a collection of journal articles, there are fewer (reliable) structural anchors available to guide the alignment process. The only reliable anchor point are article boundaries, which are identified with the help of manually corrected tables of contents. With 300 000 sentences and 2500 articles, the average length of an article is 120 sentences. Paragraph boundaries are not reliable enough to serve as anchor points, partially because of stylistic differences between the original authors and the translators, partially because of errors in the automatic digitisation process.

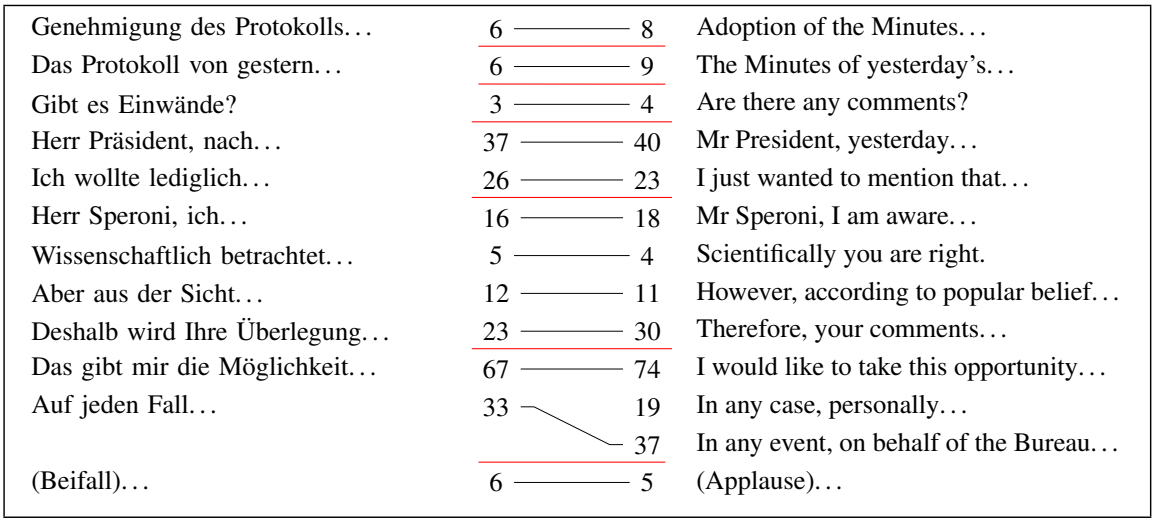

Figure 5.1: Sample sentence alignment in Europarl. Numbers indicate sentence length (number of tokens). Red lines denote structural anchors (speaker/paragraph boundaries).

A large distance between anchors is a problem because alignment errors can propagate, since a misalignment prevents other sentences from being correctly aligned. This increases the likelihood that they will be misaligned as well. In the worst case, this causes a chain reaction of misalignments up to the next anchor. There are additional error sources related to the digitised nature of the Text+Berg corpus. Image captions, which physically appear in different places in the two language versions, may be misrecognized as part of the running text. This means that an algorithm should ideally be robust towards such blocks of deleted or inserted sentences.

Additionally, the number of easy 1-to-1 sentence alignments (*beads*) is markedly lower in Text+Berg than the 90% reported as being typical for other corpora (Manning and Schütze, 1999). In a manually-aligned set of 1000 sentences, spanning 7 articles, we observe only 74% 1-to-1 beads. 16% are 1-to-2 or 2-to-1 beads, 6.3% 1-to-0 or 0-to-1 beads, and the remaining beads are m-to-n alignments of higher order. Since alignment error rates are typically higher for non-1-to-1 beads (see Gale and Church, 1993), this has consequences on the expected performance of sentence alignment tools.

Figure 5.2 shows an excerpt of the Text+Berg corpus with manual alignments. The example illustrates some reasons why the number of 1-to-1 beads is unusually low. Partially, it is due to differences in the use of punctuation, which affects automatic sentence splitting. The German book reference *Albert Eggler: Gipfel über den Wolken. Bern: Hallwag.* is split into

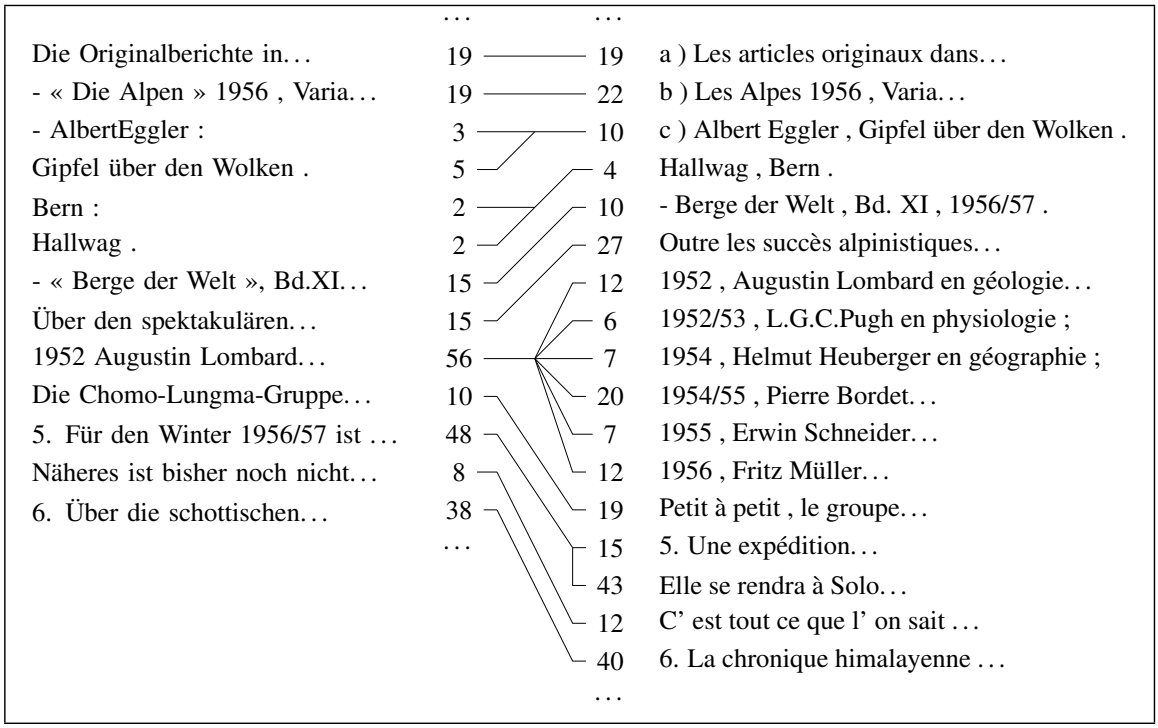

Figure 5.2: Sample sentence alignment in Text+Berg. Numbers indicate sentence length.

four sentences, its French translation into two. Also, the French translator chooses to split multiple comma-separated enumerations in the German original into separate sentences.

These examples illustrate that it is harder to perform sentence alignment for the Text+Berg corpus than for well-structured text collections such as Europarl. Still, the importance of in-domain data for SMT suggests that we should not choose training data that is easy to align, but training data from the domain which we want to translate.

5.1 Related Work

The first sentence alignment algorithms by Brown et al. (1991) and Gale and Church (1993) were based on a length-comparison between source and target text and work without language-specific information. A second strand of sentence alignment algorithms work with lexical information. Lexical information is integrated through correspondence rules (Simard et al., 1993), through dictionaries (Varga et al., 2005; Ma, 2006), or using a translation model trained on the parallel text itself (Moore, 2002; Varga et al., 2005; Braune and Fraser, 2010).

The approaches by Moore (2002), Varga et al. (2005) and Braune and Fraser (2010) are related in that they all perform a length-based alignment of the parallel text in a first pass, build a translation model from it, and use this for a second alignment pass. The differences are in the choice of the alignment model, search space pruning, and strategy to identify 1-to-many beads. Braune and Fraser (2010) note that the Microsoft Bilingual Aligner (Moore, 2002) only returns 1-to-1 beads[1], and proposes a two-step clustering approach to improve recall for 1-to-many beads. In a first step, the dynamic programming search identifies 1-to-1 beads, and in a second step, these beads (and possible unaligned sentences) are merged if a merge results in a model-optimal alignment.

Different degrees of textual and meta-textual structure open different possibilities for sentence alignment. Tiedemann (2007) shows that movie subtitles can be aligned on the basis of time stamps. Sentence alignment is also performed on comparable corpora for which the monotonicity assumption, i.e. the assumption that the sentences in the two versions of the corpus are in the same order, does not hold (Fung and Cheung, 2004; Adafre and de Rijke, 2006; Yasuda and Sumita, 2008).

5.2 MT-based Sentence Alignment

We describe an MT-based sentence alignment algorithm, introduced in (Sennrich and Volk, 2010).[2] The basic idea of MT-based sentence alignment is to not align source and target text directly, but to first perform an automatic translation of the source text into the target language, and then aligning this translation with the target text. Each potential sentence pair is assigned a weight by a modified version of BLEU, and a dynamic programming search is then performed to find the best path of alignments.

MT-based sentence alignment requires an MT system for the language pair. It is thus more demanding than length-based approaches, which require no further resources. Compared to approaches that use external dictionaries, availability of MT systems may be better or worse.

MT-based sentence alignment can also be bootstrapped, and learn the required resources from the to-be-aligned corpus itself. This bootstrapped approach consists of performing a first alignment with a length-based (or any other) sentence alignment algorithm, building an SMT system from the resulting corpus, and then using this SMT system to translate the

[1] Internally, it also computes 1-to-2 and 2-to-1 beads, which can be printed with a small modification to the implementation.

[2] We have released an implementation of the algorithm as free software on `https://github.com/rsennrich/Bleualign`

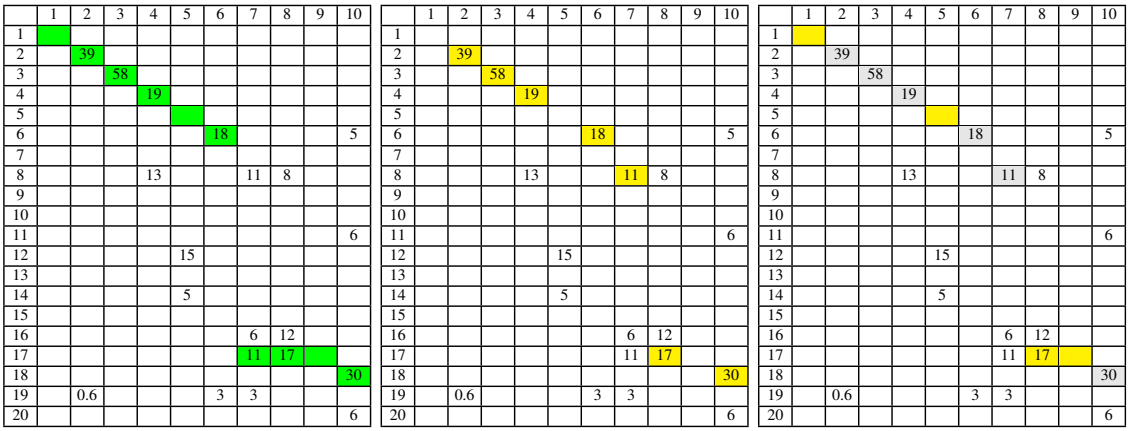

Figure 5.3: 20 × 10 alignment matrix ($T' \times T$) illustrating different steps of alignment algorithm. True alignment is shown in green for weight computation step. Beads added or updated in a step are shown in yellow; those from previous steps in gray.

corpus for a second, MT-based alignment (Sennrich and Volk, 2011). This is conceptually similar to the hybrid approach by Moore (2002) and its variants.

5.3 Bleualign: Algorithm

Given two texts $S = s_1, s_2 ... s_n$ and $T = t_1, t_2 ... t_m$, each being a sequence of sentences, and an automatic translation of S in the target language $T' = t'_1, t'_2 ... t'_n$, the task is to find sentence alignments between S and T. The algorithm achieves this by first computing an alignment between T' and T, and then replacing each t'_i with s'_i as a final step. The motivation is that it is easier to automatically decide whether any pair (t'_i, t_j) is a potential bead than it is for (s'_i, t_j). T' and T form an $n \times m$ matrix, and we will illustrate the algorithm with an example (figure 5.3). The true alignment is highlighted in the leftmost matrix. Note the 1-to-3 bead $(t'_{17}, t_{7,8,9})$, and the unaligned sentences $t'_{7,...,16}$, which makes this an example of a bitext for which length-based algorithms perform poorly.

The sentence alignment algorithm consists of three steps, illustrated in figure 5.3:

1. A weight is computed for every sentence pair in the matrix.
2. A dynamic programming search finds the best path of 1-to-1 beads in the matrix.
3. Gaps in the alignment matrix are filled using two procedures: a higher-order alignment search and a length-based alignment.

5.3.1 Weighting Sentence Pairs

In a first step, a weight is assigned to every sentence pair in the alignment matrix, using a similarity score. In principle, any similarity score can be used, and it can be adjusted to the language pair.[3]

Bleualign uses 2-gram BLEU as similarity measure (Papineni et al., 2002). BLEU has been criticised as a measure of translation quality (see section 2.4), and is not considered reliable on a sentence level (Callison-Burch et al., 2006), but the decision whether two sentences correspond to each other or not is markedly easier than comparing the quality of two systems.

The choice of 2-gram BLEU over the more conventional 4-gram BLEU is motivated by the fact that the final score uses the geometric mean of the different n-gram orders, resulting in a score of 0 for sentence pairs without matching 4-gram. Thus, 2-gram BLEU remains useful for sentences with no 4-gram matches.[4] Also, the algorithm measures BLEU in both directions, once with t'_i as hypothesis and t_j as reference, once vice versa, to obtain a symmetrical score, since the original BLEU score is an asymmetrical precision score.

The complexity of this step is $O(n \times m)$, with n being the number of sentences in the source text, m the number of sentences in the target text. The search space is not pruned because the algorithm is conceived to work even if the alignment does not fall along the main diagonal. The computation of BLEU is relatively inexpensive, especially for the large portion of sentence pairs for which their sets of bigrams are disjoint, and thus have a BLEU score of 0.[5] Note that n and m are not the size of the whole corpus, but of the text between two anchor points, since each such text can be aligned independently.

5.3.2 Dynamic Programming Search

As a next step, the algorithm finds a path of 1-to-1 beads that maximizes the sum of weights along the path, while retaining a monotonic order of the alignments. The search is restricted to 1-to-1 beads for computational complexity reasons; 1-to-many or many-to-1 beads are extracted in a later step.

The search can be conceptualised as a shortest-path search through a weighted acyclic directed graph, with a complexity of O($|V|+|E|$), $|V|$ being the number of vertices, $|E|$ the number of edges. We define all cells in the alignment matrix as vertices, with each cell being

[3] Yildiz and Tantuğ (2012) report that a lexicon-based sentence alignment for English–Turkish performs better when stemming is applied, and the same may be true for MT-based alignment for agglutinative languages.

[4] An alternative solution would be to smooth the number of n-gram matches (Lin and Och, 2004).

[5] Also, note that we need to perform costly operations such as normalization and n-gram extraction only once per sentence.

connected to its neighbours through (directed) edges.[6] The algorithm has a complexity of $O(n \times m)$, and is a close variant of the longest common subsequence algorithm, with the only difference being that each cell represents a sentence pair instead of a character pair, and that the weight of each cell is real-valued rather than binary.

5.3.3 Additional Alignment Procedures

The dynamic programming search has two main shortcomings. Firstly, it only finds 1-to-1 beads. Secondly, sentence pairs that have a BLEU score of 0 are never extracted, which impairs recall. Thus, two further procedures are applied to find more alignments.

In a first heuristic, any 1-to-1 beads that precede or follow unaligned sentences (either in T' or T) are concatenated with those unaligned sentences. This results in 1-to-n and/or n-to-1 candidate beads, with n ranging from 2 to a configurable maximum. A 1-to-n or n-to-1 candidate bead is preferred over the original 1-to-1 bead if it obtains a higher weight.[7] In our example in figure 5.3, the sentences t_9, and $t'_9 ... t'_{16}$ are unaligned after the dynamic programming search, among others. The procedure thus forms a number of new candidate beads from the existing bead (t'_{17}, t_8), such as $(t'_{16,17}, t_8)$ and $(t'_{17}, t_{8,9})$. Since $(t'_{17}, t_{8,9})$ scores higher than both the original bead and all other candidates formed from the original bead, it is added to the best path of beads.

In a final step, unaligned sentences are aligned with the length-based Gale and Church algorithm, using the beads from the previous step as anchor points. Length-based alignment is performed for each gap, i.e. each block of sentences between two anchor points, unless its size is both asymmetrical by a factor larger than two and larger than three sentences. This adds the beads (t'_1, t_1) and (t'_5, t_5) to the matrix in figure 5.3. Note that if the dynamic programming search has a low precision, for instance because no translation is provided, this procedure will also perform poorly, since it relies on the beads of the dynamic programming search as anchors. Even if the anchor points are reliable, we expect this length-based alignment to have a lower precision than the other procedures. Nevertheless, we consider it important to not only extract sentence pairs that are already well-translated, i.e. for which t'_i and t_j are similar and obtain high BLEU scores. If the automatic translation of a sentence is poor because of unknown words, this means that its human translation is particularly valuable for training.

[6] Note that this algorithm is slightly more efficient than the one we describe in Sennrich and Volk (2010), which required pruning to keep complexity to $O(n \times m)$.

[7] For BLEU, the algorithm requires an increase in the number of n-gram matches, to make sure that the BLEU score not only improves because of a change in brevity penalty.

system	strict			lax		
	P	R	F_1	P	R	F_1
Gale and Church	0.67	0.68	0.68	0.79	0.80	0.80
MBA	0.89	0.67	0.76	0.96	0.72	0.83
Gargantua	0.79	0.85	0.82	0.88	0.93	0.90
Bleualign	0.87	0.84	0.85	0.99	0.95	0.97

Table 5.1: Sentence alignment quality on manually-aligned test set.

5.4 Evaluation of Sentence Alignment

For an evaluation of sentence alignment algorithms, we use the same test procedure and manually aligned test set as in (Sennrich and Volk, 2010), but with a larger release of the Text+Berg corpus.

A hand-aligned set of 1000 sentences, or 7 articles, from the Text+Berg corpus, serve as gold standard. The evaluation measures precision, recall and F_1 with two truth conditions. Under the *strict* condition, an alignment hypothesis is considered true if it exactly matches the gold alignment. Under the *lax* condition, a partial match on both language sides is sufficient.

The evaluation covers four systems: an implementation of (Gale and Church, 1993), the Microsoft Bilingual Aligner (MBA) (Moore, 2002), Gargantua (Braune and Fraser, 2010), and Bleualign (Sennrich and Volk, 2010). For Bleualign, we use the approach described in (Sennrich and Volk, 2011), namely a bootstrapped approach that requires no existing MT system, but uses an SMT system trained on the corpus itself, based on an alignment with (Gale and Church, 1993).

Table 5.1 shows alignment quality measured on the manually-aligned test set. More variants, for instance Bleualign with different translation systems, were evaluated in (Sennrich and Volk, 2010). Gale and Church serves as a baseline with an F_1 of 0.68 (strict) and 0.80 (lax). MBA beats the length-based algorithm in terms of precision, but offers slightly lower recall values. The F_1 score is 0.76 (strict) and 0.83 (lax). Gargantua obtains better recall values than MBA, at the cost of precision. In terms of F_1, it beats MBA by 0.06 (strict) and 0.07 (lax) points. Bleualign performs best, with an F_1 of 0.85 (strict) and 0.97 (lax). Compared to the results in (Sennrich and Volk, 2010), where Bleualign was applied with out-of-domain translation systems, there is an improvement in F_1 of 0.04 (strict) through the bootstrapped approach described in (Sennrich and Volk, 2011).

Table 5.2 reports the size of the phrase table that is learned from the different corpora, both before and after pruning according to statistical significance tests (Johnson et al., 2007).

system	phrase table size	
	unfiltered	filtered
Gale and Church	**20 000 000**	365 000
MBA	8 140 000	344 000
Gargantua	15 400 000	543 000
Bleualign	14 400 000	**592 000**

Table 5.2: Phrase table size (number of phrase pairs) with different sentence alignment algorithms, before and after significance filtering (Johnson et al., 2007).

system	DE–FR		FR–DE	
	BLEU	METEOR	BLEU	METEOR
Gale and Church	17.1	35.8	12.8	32.6
MBA	17.0	35.8	13.2	32.7
Gargantua	**17.8**	**36.8**	**13.4**	**33.2**
Bleualign	**17.8**	**36.9**	**13.4**	**33.2**

Table 5.3: SMT performance. Best systems (no other system being significantly better) marked in bold.

The two sizes are an indication of how many good sentence pairs are found. Without significance filtering, the Gale and Church system has the largest phrase table by a wide margin. However, the high number of phrase pairs is actually a sign of randomness. When filtering out phrase pairs that do not meet the required significance threshold[8], it becomes markedly smaller than the filtered tables based on Gargantua and Bleualign. The phrase table sizes indicate that the system based on Bleualign has the highest number of non-random phrase pairs, whereas the Gale and Church system has the highest number of random phrase pairs.

Table 5.3 shows SMT performance with different sentence alignment algorithms being used to align the parallel training corpus. Gargantua and Bleualign are both winners in that they significantly outperform Gale and Church and the Microsoft Bilingual Aligner, by a margin of 0.2–1.1 BLEU/METEOR points. The difference between Bleualign and Gargantua is not statistically significant.

As to the choice of a sentence alignment tool, both Bleualign and Gargantua show comparable performance, and outperform MBA and Gale and Church's algorithm.[9] Gargantua has the advantage of having no external dependency, while Bleualign either requires an existing MT system or a toolkit to train an SMT system from the to-be-aligned corpus. However, the

[8] defined as a p-value lower than the statistical significance of 1-1-1 phrase pairs, i.e. phrase pairs for which the frequency of the source phrase, the target phrase, and the phrase pair are all 1.

[9] In an evaluation on two different data sets, Bleualign performed best (Abdul-Rauf et al., 2012).

possibility to use any MT system turns into an advantage for Bleualign if the bitext that one wants to align is relatively small. In this case, too little data may be available for reliably estimating the IBM-1 model that Braune and Fraser (2010) use for the second alignment pass in Gargantua.

5.5 On the Relation Between Sentence Alignment Quality and SMT Performance

We cannot directly cross-reference tables 5.1 and 5.3. The results in table 5.1 are based on 7 articles, and the difference in alignment quality on the full *SAC_145* corpus with approximately 2500 articles may be more extreme. With a different test set, consisting of a single article of 500 sentences, we observed MBA and Gale and Church's algorithm to completely fail (Sennrich and Volk, 2011). However, there are other means to investigate the relationship between sentence alignment quality and SMT performance.

Considering the alignment metrics precision and recall, we will first discuss how relevant precision is for SMT performance. Since we have no data on sentence alignment precision on the whole *SAC_145* data set, this will be done with artificially induced noise. The basis for this experiment is the corpus aligned with Bleualign. Randomly aligned sentence pairs from the monolingual parts of *SAC_145* are added until the size of the original corpus is 87.5%, 75%, 50%, 25% or 12.5% of the full corpus.[10] Since the number of correctly aligned sentence pairs is not changed (although it is lower than 100% to an unknown degree), this corresponds to corpora with differing precision, but identical recall. This experiment allows us to estimate how robust SMT systems are against misaligned sentence pairs, and conversely, how important sentence alignment precision is.

Table 5.4 shows SMT performance for these systems with artificial misalignments. With 12.5% and 25% misalignments, performance is significantly worse, with a drop of 0.3 BLEU points, and 0.3–0.6 METEOR points relative to the baseline. With 50% noise, performance drops by 1.1 BLEU and 1.8 METEOR points. With 87.5% noise, the system is 5.6 BLEU and 8.6 METEOR points worse than the original system.

The main reason why misaligned sentence pairs hurt SMT is that in randomly aligned sentence pairs, word alignment is similarly random. This hampers word alignment with the IBM models, even for the correctly aligned sentence pairs. Wrong word alignments cause two problems for the phrase table. Firstly, additional, random phrase pairs are added to the

[10] Sentences occurring in the parallel corpus have been removed, so that only new sentences are added to the parallel corpus.

alignment precision	unfiltered phrase table BLEU	METEOR	filtered phrase table BLEU	METEOR
1	17.8	37.0	17.8	36.9
0.875	**17.5**	**36.6**	17.7	36.8
0.75	**17.5**	**36.3**	17.8	**36.7**
0.5	**16.7**	**35.2**	**17.3**	**36.5**
0.25	**14.3**	**31.7**	**16.1**	**34.8**
0.125	**12.2**	**28.4**	**14.5**	**32.8**

Table 5.4: SMT performance DE–FR with artificially induced misalignments. The misalignments are added to the original corpus (aligned with Bleualign), so the number of correctly aligned sentence pairs is constant.

phrase table. Secondly, the translation probability estimates are based on noisier data, and thus become more uniform. Significance tests can partially neutralize the first type of noise, i.e. random phrase pairs, and the results in table 5.4 confirm that performance degrades more slowly when the phrase table is filtered with a significance test, the difference between the filtered and the unfiltered system being 2.3 BLEU and 4.4 METEOR points for the most extreme setting with 87.5% misalignments.

One conclusion from the results is that, if we apply significance filtering, SMT systems are relatively robust towards misaligned sentence pairs. While a low sentence alignment precision harms SMT performance, we only see those effects at values of below 75% alignment precision. If sentence aligners perform above this level, there is little potential in trying to filter out noisy sentence pairs. In (Sennrich and Volk, 2010), we report on attempts to improve sentence alignment precision by performing MT-based sentence alignment in both translation directions, and then intersecting the results. While this procedure did improve sentence alignment precision, it lowered recall, and did not improve SMT performance.

A causal link between the recall of sentence alignment algorithms and SMT performance is easy to establish. By definition, algorithms with higher recall extract more good sentence pairs from a parallel corpus, which reduces data sparseness and leads to an improvement in SMT performance as seen in section 4.2. We can predict score gains for different training set sizes based on the learning curves in table 4.2 (p. 49), and cross-reference those with the actual performance observed in table 5.3. For a comparison of MBA and Bleualign, we assume that we can fully attribute the difference in SMT performance to differences in recall, since the difference in precision is small. The training corpus obtained with Bleualign is 60% larger than the one obtained with MBA, so we assume that this number reflects the relative improvement in recall by Bleualign.

The logarithmic learning curves for *SAC_test* with systems trained on *SAC_145* suggest that for German–French, performance improves by 1.5 BLEU points per doubling of the amount of parallel training data.[11] From an increase in recall (training set size) by 60%, we would expect a gain in BLEU score by $log_2(1.6) \cdot 1.5 \approx 1$. For French–German, performance increases by about 1.3 per doubling, so the expected score difference would be $log_2(1.6) \cdot 1.3 \approx 0.9$. The actually observed differences in table 5.3 are 0.8 BLEU points, which is slightly below, but close to these expectations.

This causal link between sentence alignment recall and SMT performance may be useful to estimate the potential effect of choosing a different sentence aligner, and explains why it will not improve performance for all corpora. If the corpus is relatively easy to align, and a length-based algorithm achieves a recall of above 90%, only a small improvement in recall (and thus SMT quality) is possible. Conversely, an increase in recall by 60% can result in a significant performance improvement.[12]

5.6 Summary

We discuss why sentence alignment is harder for a relatively unstructured and noisy corpus such as Text+Berg, compared to well-structured text collections such as Europarl. We propose a novel sentence alignment algorithm that is based on the machine translation of the source text. The main alignment is performed between the automatic translation of the source text and the target text, based on surface similarity of potential sentence pairs. The best path of alignments, found in a dynamic programming search, is then complemented through multiple additional alignment procedures. We find that Bleualign, an implementation of this algorithm, outperforms other sentence alignment tools on a hand-aligned test set. By applying different sentence alignment tools on an SMT training corpus, we also observe significant improvements in translation quality from the better-performing alignment tools.

We conclude that our novel sentence alignment algorithm successfully reduces the data sparseness problem in our target domain. To further reduce data sparseness, out-of-domain data can be used, and the next chapter is dedicated to the combination of in-domain and out-of-domain parallel data.

[11] At least in the range that we are concerned with. We cannot extrapolate these numbers to very low amounts of data.

[12] The logarithmic nature of the learning curve suggests that this will hold true regardless of the absolute amount of training data. Indeed, the benefit of better sentence alignment has not decreased in this thesis, although the evaluation is based on twice the amount of training data compared to (Sennrich and Volk, 2010).

6 Translation Model Combination: Tackling the Ambiguity Problem

In section 4.2, we operated with the most basic method of combining parallel training data, its concatenation. This section is devoted to alternative methods, which all have in common that in-domain data is given preference in some ways in order to mitigate the ambiguity problem. We both give an overview of existing approaches, and propose new methods.[1]

The settings in which one may want to perform domain adaptation vary. A setting with one in-domain and one out-of-domain corpus, as used in section 4.2 and other domain adaptation research, is an idealization. Typically, we have a larger number of parallel corpora available, and merging them into one in-domain and one out-of-domain corpus is not necessarily easy or useful. It is also possible that a system needs to support multiple target domains (Banerjee et al., 2010), or that we have one multi-domain training corpus with which we want to perform unsupervised adaptation (Yamamoto and Sumita, 2007). We will operate under the assumption that the target domain is known for any given task, but we envision a system that can be quickly adapted to new domains, and which can switch between decoding for multiple domains online. We describe an architecture that supports both online adaptation and multiple target domains in section 6.6.

We further assume that multiple parallel corpora are available for training, but that we cannot make any further assumption about their relative fitness.[2] A development set of about 1000 sentences will serve as the only information about the target domain. Ideally, a domain adaptation method should perform well even if we cannot make a binary distinction into in-domain and out-of-domain corpora, and should perform no worse than the baseline, a concatenation of all data, if several (or all) corpora are comparably fit, meaning that none is preferable for the target domain. A method that fulfills these requirements may not perform better in terms of BLEU under ideal circumstances, but reduces both the effort and risk of building a domain-specific SMT system.

[1] This chapter builds on our work in (Sennrich, 2012b,a; Sennrich et al., 2013).

[2] As discussed in chapter 3, we want to emphasize that the question in domain adaptation is not how "good" or "bad" data or a model is, but how well-adapted it is to the task at hand. In other words, the same model has a different fitness for different tasks, and adapting a model to one domain may reduce its fitness for others.

With risk, we mean the fact that performance gains or losses can be unpredictable. We have seen that adding out-of-domain training data can both improve or decrease SMT performance, depending on the relative effects of the unknown word problem and the ambiguity problem (see figures 4.1 (p. 48) and 10.3 (p. 148)). Building a domain-specific system could thus entail the tedious process of ascertaining that none of the training corpora that has been selected harms performance, at worst through trial-and-error. Thus, the risk of performance degradation requires an additional effort on the part of the creator of the SMT system. The same argument can be made for any adaptation technique.

We will begin with a discussion of different translation model combination techniques, and then proceed to an empirical evaluation. Our main contributions are techniques for perplexity optimization with instance weighting (section 6.2.3), an approach which combines training data clustering and domain adaptation (section 6.5), and a framework for decoding-time domain adaptation (section 6.6).

6.1 Discussion of Domain Adaptation Techniques

6.1.1 Log-linear Interpolation

A log-linear combination of translation models is a good fit into the general log-linear architecture of SMT systems. A major advantage is that the model weights can be optimized directly through MERT (or any other discriminative optimizer for log-linear SMT). Each translation model typically consists of multiple features, and a log-linear combination adds all of them to the global log-linear model (equation 5, p. 22).

One problem is that, without smoothing, observations with probability 0 in one model will also have an interpolated probability of 0.[3] This means that the interpolated translation model is the intersection of the component models, which runs counter to our goal of minimizing the unknown word problem. In the evaluation, we will apply uniform smoothing, i.e. a constant probability of 0.001 (for all features except for the phrase penalty, which is 2.718), for unknown phrase pairs.

A second method to integrate additional translation models into the log-linear model is using them as alternative decoding paths (Birch et al., 2007), an idea which Koehn and Schroeder (2007) first used for domain adaptation. The basic difference is that this method does not use all translation models to score a phrase pair, but considers phrase pairs from different models as alternative hypotheses. Each translation model comes with its own set of weights (5 in the standard Moses model), which can be discriminatively optimized. If a

[3] Defining $\log(0)$ as $-\inf$.

phrase pair occurs in multiple phrase tables, this results in multiple translation options being generated, each using only the features from one model, rather than features from both.

Theoretically, mixing several translation models log-linearly is guaranteed to lead to equal or better performance compared to using only one of the models. In the worst case, the additional models have a detrimental effect, but we can always set the model weights so that they are fully ignored. In practice, each translation model adds 5 more dimensions to the weight space, and with many models, discriminative optimization will either take far longer, search a smaller region of the weight space, or both. Thus, scaling this method to a high number of models is problematic with MERT, although PRO or MIRA promise to scale more gracefully.

A suboptimal choice of weights is not the only weakness of alternative paths, however. Let us assume that all models have the same weights. Note that, if a phrase pair occurs in several models, combining models through alternative paths means that the decoder selects the one with the highest probability, whereas with linear interpolation, the probability of the phrase pair would be the average of all models. Selecting the highest-scoring phrase pair is the less robust decision, and prone to data noise and data sparseness. If one model overestimates the probability of a phrase pair, e.g. because the estimation is based on a low number of observations, this phrase pair is likely to be selected, which may negatively affect the translation.

As a side note, we want to emphasize that different combination algorithms need not be mutually exclusive. Regardless of the domain adaptation technique that we apply, we use a log-linear SMT model as the global model. Even for translation model adaptation, hybrid approaches can be successful. In (Sennrich, 2011a), we combined the architecture with alternative paths with a second method of rescoring an out-of-domain model.[4]

6.1.2 Linear Interpolation

A well-established approach in language modelling is the linear interpolation of several models, i.e. computing the weighted average of all probabilities. It is defined as follows:

$$p(x|y;\lambda) = \sum_{i=1}^{n} \lambda_i p_i(x|y) \tag{1}$$

with λ_i being the interpolation weight of each model i, and with $(\sum_{i=1}^{n} \lambda_i) = 1$. Note that this weight vector λ is independent from the weight vector of the global log-linear model.

[4]More specifically, a model trained on translation hypotheses of other MT systems. See chapter 7.

For SMT, linear interpolation of translation models has been used in numerous systems. The approaches diverge in how they set the interpolation weights. Some authors use uniform weights (Cohn and Lapata, 2007), others empirically test different interpolation coefficients (Finch and Sumita, 2008; Yasuda et al., 2008; Nakov and Ng, 2009; Axelrod et al., 2011), others apply monolingual metrics to set the weights for translation model interpolation (Foster and Kuhn, 2007; Koehn et al., 2010).

There are reasons against all these approaches. Uniform weights are easy to implement, but give little control. Empirically, it has been shown that they often do not perform optimally (Finch and Sumita, 2008; Yasuda et al., 2008). An optimization of BLEU scores on a development set is promising, but slow and impractical, especially if the translation model is rebuilt between optimization runs. There is no easy way to integrate linear interpolation into log-linear SMT frameworks and perform discriminative optimization, since linear interpolation does not behave linearly in the log-space in which log-linear models are optimized.[5] Monolingual optimization objectives such as language model perplexity have the advantage of being well-known and readily available, but their relation to the ambiguity problem is indirect at best. Foster et al. (2010) optimize the interpolation weights so as to minimize the translation model perplexity on a development set of phrase pairs, which has the attractive property of being directly related to the ambiguity problem. A lower perplexity means that more probability mass is given to the correct phrase pairs (according to a reference set), which also means that the model will be more likely to predict these phrase pairs during decoding.

Linear interpolation is seemingly well-defined in equation 1. Still, there are a few implementation details worth pointing out. If we directly interpolate each feature in the translation model, and define the feature values of non-occurring phrase pairs as 0, this disregards the meaning of each feature. If we estimate $p(x|y)$ via MLE (see equation 3), and $p(y) = 0$, then $p(x|y)$ is strictly speaking undefined. A naive algorithm treats unknown phrase pairs as having a probability of 0, which results in a deficient probability distribution. Alternatively, we propose and implement the following algorithm. For each value pair (x, y) for which we compute $p(x|y)$, we replace λ_i with 0 for all models i with $p(y) = 0$, and then re-normalize the weight vector λ to 1. We do this for $p(\overline{t}|\overline{s})$ and $lex(\overline{t}|\overline{s})$, but not for $p(\overline{s}|\overline{t})$ and $lex(\overline{s}|\overline{t})$, our reasoning being the consequences for perplexity minimization (see section 6.2.3). Namely, we do not want to penalize a small in-domain model for having a high out-of-vocabulary rate on the source side, but we do want to penalize models that know the source phrase, but not its correct translation.

[5] Recently Haddow (2013) has extended PRO to also optimize a linear mixture of translation models.

A second modification pertains to the lexical weights $lex(\bar{s}|\bar{t})$ and $lex(\bar{t}|\bar{s})$, which form no true probability distribution, but are derived from the individual word translation probabilities of a phrase pair (see Koehn et al., 2003). The lexical weights are calculated as follows, using a set of word alignments a between $\bar{s}$ and $\bar{t}$:[6]

$$lex(\bar{s}|\bar{t},a) = \prod_{i=1}^{n} \frac{1}{|\{j|(i,j) \in a\}|} \sum_{\forall(i,j)\in a} w(s_i|t_j) \tag{2}$$

A special NULL token is added to $\bar{t}$ and aligned to each unaligned word in $\bar{s}$. $w(s_i|t_j)$ is calculated through MLE, as in equation 3, but based on the word (pair) frequencies.

We propose to not interpolate the lexical weights $lex(\bar{s}|\bar{t})$ and $lex(\bar{t}|\bar{s})$ directly, but the word translation probabilities w which are the basis of the lexical weight computation. The reason for this is that word pairs are less sparse than phrase pairs, so that we can even compute lexical weights for phrase pairs which are unknown in a model.[7] We will evaluate these proposed modifications of linear interpolation, along with a naive variant, which defines $p(x|y) = 0$ if $p(y) = 0$, and directly interpolates the lexical weights.

6.1.3 Instance Weighting

Weighting of different corpora can also be implemented through a modified Maximum Likelihood Estimation. The traditional equation for MLE is:

$$p(x|y) = \frac{c(x,y)}{c(y)} = \frac{c(x,y)}{\sum_{x'} c(x',y)} \tag{3}$$

where c denotes the count of an observation, and p the model probability. If we generalise the formula to compute a probability from n corpora, and assign a weight λ_i to each, we get:[8]

$$p(x|y;\lambda) = \frac{\sum_{i=1}^{n} \lambda_i c_i(x,y)}{\sum_{i=1}^{n} \sum_{x'} \lambda_i c_i(x',y)} \tag{4}$$

We also use this weighted variant of MLE for the word translation probabilities $w(s_i|t_j)$ in equation 2 to introduce instance weighting for lexical weights.

[6] The equation shows $lex(\bar{s}|\bar{t})$; $lex(\bar{t}|\bar{s})$ is computed analogously.

[7] For instance if the word pairs (the,der) and (man,Mann) are known, but the phrase pair (the man, der Mann) is not.

[8] Unlike equation 1, equation 4 does not require that $(\sum_{i=1}^{n} \lambda_i) = 1$.

The main difference to linear interpolation is that this equation takes into account how well-evidenced a phrase pair is. This includes the distinction between lack of evidence and negative evidence, which is missing in a naive implementation of linear interpolation.[9]

Translation models trained with weighted training instances have been discussed before, and have been shown to outperform uniformly weighted models in some settings. However, researchers who demonstrated this fact did so with arbitrary weights (e.g. Koehn, 2002), by empirically testing different weights (e.g. Nakov and Ng, 2009), or by using language model perplexity (Shah et al., 2012). Matsoukas et al. (2009) optimize instance weights discriminatively, but since their weights do not fit into the log-linear framework, they do so by optimizing translation error rate (TER) on n-best lists of hypotheses. We propose an optimization of the weights in equation 4 by minimizing translation model perplexity, which allows for a quick and automatic optimization of instance weights.

6.1.4 Data Selection

Data selection for translation models has been presented in (Yasuda et al., 2008; Axelrod et al., 2011). Both approaches have in common that they use an in-domain translation model, combined with sentence pairs from out-of-domain corpora that are similar to the in-domain text. Yasuda et al. (2008) use language model perplexity, while Axelrod et al. (2011) use the cross-entropy difference between an in-domain and a general model as proposed by (Moore and Lewis, 2010).

We argue that these perplexity-based data selection methods do not adequately consider the unknown word problem. Even if discarding parallel training data reduces perplexity, which would indicate that ambiguities are better resolved in the smaller model, discarding data will also increase data sparseness, and can thus lead to worse performance than with all available training data.

Alternatively, one can mine out-of-domain resources for unseen words, as e.g. Daumé III and Jagarlamudi (2011) and Banerjee et al. (2012) suggest. Such approaches specifically aim to reduce the unknown word problem, but they do not address the ambiguity problem. Banerjee et al. (2012) show an improvement over a purely in-domain system, but do not compare their system to one that uses all available training data.

[9] Smoothing or significance filtering can downgrade or remove phrase tables with little evidence, so the problem may not be as pressing in practice.

6.1.5 Priority Merge

Nakov (2008) perform a priority merge of phrase tables, creating a phrase table that consists of all phrase pairs of the original one, plus all phrase pairs from an *extra* phrase table that do not occur in the original one. The phrase table is extended by additional features to indicate each phrase pair's origin. This adaptation method is also known as "phrase-table fill-up" (Bisazza et al., 2011).

Haddow and Koehn (2012) evaluate the same merging procedure, without adding any features, and show that it outperforms a concatenation of training data. The motivation behind a priority merge is consistent with the unknown word problem and ambiguity problem: new phrase pairs are introduced to reduce the unknown word problem, but for known phrase pairs, the out-of-domain model is disregarded, so that it does not exacerbate the ambiguity problem. The algorithm can be regarded as a special case of linear interpolation, with the original model receiving a weight near 1, the extra model a weight near 0.[10]

While a priority merge can yield the desired result, it has several limitations. Considering that it can be conceived as an extreme form of linear interpolation, there are situations for which we expect less extreme weights to be optimal. For instance, if multiple models have a comparable fitness, we expect that considering evidence from all of them is better than just selecting one probability estimate.

It is unclear how to automatically prioritize models in a setting with a higher number of models, or when the target domain is not known beforehand, but specified through a development set. One could engineer ways to automatically prioritize the models, for instance using language model perplexity for ranking. In our evaluation, we do not perform such experiments, but only report on the results for the binary setting with one in-domain, one out-of-domain model.

6.1.6 Origin Features

A common method is the extension of the translation model through additional features that indicate the origin of the phrase pair. There are multiple ways to implement this idea, and different features have been proposed (e.g. Nakov, 2008; Niehues and Waibel, 2010; Daumé III and Jagarlamudi, 2011).

An implementation that includes phrase pairs from multiple corpora and only differentiates them through this origin feature is conceptually very similar to the alternative paths idea presented in the log-linear interpolation section, with the difference that phrase pairs

[10] How well linear interpolation simulates this algorithm depends on an implementation detail discussed in section 6.1.2, namely how to distribute the probability mass if a phrase pair does not occur in all models.

from different models share the log-linear weights. The main disadvantage is the same as for alternative weights decoding: the phrase pairs from different models compete during decoding, and a max (or min) function determines the most likely (or least costly) hypothesis as winner, but the evidence from different corpora is not consolidated, and the method is sensible to outliers. Of course, this may not be true for all implementations, and there are systems that combine multiple methods. We will evaluate an implementation of a priority merge that includes origin features, but we will not claim any generalisable insight into the use of origin features because of the many ways to integrate them.

6.2 Perplexity

We will now introduce some measures from information theory and discuss how they can be applied to translation models. This serves two purposes. Firstly, they can serve as an indirect, but quickly optimizable, objective function for setting optimal model weights of various combination methods such as linear interpolation or instance weighting. Secondly, they can serve as an analytic tool to investigate the extent of the ambiguity problem for a given translation model and a reference set of translations.

6.2.1 Theoretical Background

In information theory, entropy is a measure of "information, choice and uncertainty" (Shannon, 1948). How well we can predict the outcome of an event depends on the number of possible outcomes and on how probable each one is. The entropy is highest if the outcome is fully random, with a uniform probability $\frac{1}{n}$ for each of n outcomes, and zero if one outcome is certain and all others impossible. The notion that texts are not fully random lies at the heart of language modelling. On the basis of an entropy analysis, Shannon (1951) has shown that we can better predict symbols (letters or words) if we consider the preceding symbols, which is the basis for n-gram models. In applications such as speech recognition and MT, we use this predictive power of language models to deal with noisy channels, e.g. to predict which translation of an ambiguous source phrase is most probable. A "perfect" language model would be able to predict all symbols with certainty, eliminating any ambiguity from the noisy signal.[11]

[11] This, of course, would only be possible if a language were fully redundant, and thus quite boring.

The classical definition of the entropy H is:

$$H = -\sum_{x} p(x) \log_2 p(x) \tag{5}$$

Cross-entropy does not measure the uncertainty inherent in a text, but compares the model probability distribution q to a "true" one p, which is estimated through a reference set with N events.

$$H(p,q) = -\sum_{x} p(x) \log_2 q(x) \tag{6}$$

$$H(p,q) \approx -\sum_{i=1}^{N} \frac{1}{N} \log_2 q(x_i) \tag{7}$$

A derived measure is perplexity, defined as $2^{H(p,q)}$. For optimization, the difference between cross entropy and perplexity is purely cosmetic.

A low perplexity means that the model gives a high probability to the events in the reference set, which entails that it is more likely to predict these events in a generative setting than a model with high perplexity. Thus, a low perplexity is a desirable property for language models, and lower perplexity has been shown to correlate with better performance in various applications such as speech recognition (Chen et al., 1998; Chen and Goodman, 1998) and MT (Brants et al., 2007). The degree of correlation depends on the exact nature of the models being compared. Chen and Goodman (1998) find that "this correlation [between perplexity and performance] is especially strong when considering only n-gram models that differ in the smoothing technique used". There are limitations to the usefulness of perplexity, for instance when comparing models with different vocabularies. Brants et al. (2007) note that "perplexities tend to be higher with larger vocabularies", which means that if we compare models trained on different data, the one with the higher perplexity (and larger vocabulary) may actually be the better-performing one. This may also account for the weak correlation between perplexity and translation performance observed by (Eck et al., 2004).

6.2.2 Translation Model Perplexity

On a purely theoretical basis, monolingual models tell us nothing about how ambiguous the translations in a parallel corpus are, and how close the translations are to the ones we wish for. Still, language model perplexity has successfully been used to optimize translation model weights (Foster and Kuhn, 2007; Koehn et al., 2010), and for translation model data selection (Yasuda et al., 2008). This may work in practice because a corpus that is from

the same domain as the reference set is typically both more similar monolingually and more similar in terms of how ambiguous words are translated. Nevertheless, we aim to directly capture the perplexity of the translation model.

For the feature $p(\bar{t}|\bar{s})$, we calculate the cross-entropy as follows:

$$H = -\sum_{\bar{s},\bar{t}} \frac{1}{N} \log_2 p(\bar{t}|\bar{s}) \tag{8}$$

with N being the number of phrase pairs $(\bar{s}, \bar{t})$ in an in-domain reference set. The definitions for the other translation model features are analogous. The algorithm ignores phrase pairs in the test set that are missing from the translation model, and strictly speaking would be assigned a probability of 0.[12]

Translation model perplexity is an integral part of expectation–maximization word alignment (Brown et al., 1993), and has been used to analyse the performance of SMT systems (Al-Onaizan et al., 1999). It has also been used to optimize a linear interpolation of translation models (Foster et al., 2010).

The main obstacle to computing translation model perplexity is that it requires an alignment between the source and the target text on the word/phrase level[13] One possibility would be a manual alignment, which exists for *SAC_test* as part of the Smultron treebank (Volk et al., 2010b). However, we will focus on using an automatic alignment in order to demonstrate that the method remains practical in settings without manual alignment. We automatically align development sets with the same procedure that we use for training translation models (i.e. IBM word alignment and heuristic phrase pair extraction), and use the resulting phrase pairs as reference set. This reference set is undoubtedly noisy, but the noise is reduced by the fact that phrase pairs that do not occur in any of the translation models are ignored.

Operating with translation model perplexity has several advantages over monolingual measures if we want to optimize translation models. Firstly, there is a clear connection between translation model perplexity and the resolution of translation ambiguities. A lower perplexity indicates that more probability mass is assigned to correct translations, and less to incorrect ones.[14] Conversely, a higher perplexity indicates that the ambiguity problem is

[12] For this reason, the measure is unsuitable to compare translation models with different vocabularies. During optimization, the vocabulary is kept constant.

[13] In contrast, language model perplexity requires all n-grams in a reference text, which are trivial to extract.

[14] Assuming that the feature being optimized is a true probability distribution. Otherwise, a model might assign a pseudo-probability of 1 to all observations, which would look good in a perplexity computation, but have little discriminative value. In hindsight, it is always easy, but not always believable, to say "I could have seen that coming".

more severe, and that the model gives lower probabilities to the correct phrase pairs. Thus, minimizing perplexity maximizes the probability of translations that are correct according to the reference set, and thus increases the probability that they will be produced during decoding.

Secondly, there is not just one translation model perplexity, but one per feature.[15] When optimizing model weights in domain adaptation, different features may have different optima, which we cannot find through monolingual measures.

6.2.3 Perplexity Minimization

Discriminative optimization, although having a desirable objective, i.e. directly optimizing BLEU or another evaluation metric, is problematic for linear interpolation and instance weighting because of the non-linear behaviour of linear interpolation coefficients or instance weights in the logarithmic space in which the log-linear model operates.

We choose to optimize translation model perplexity to weight the component models of instance weighting and linear interpolation. While perplexity minimization has attractive theoretical properties, such as being a convex search problem, a major pre-condition is that lower perplexity correlates with better application performance.

Schwenk and Koehn (2008) discuss perplexity minimization for the mixture of language models, which has become common practice in MT (e.g. in the WMT 2008 shared task: (Déchelotte et al., 2008; Schwenk et al., 2008; Blackwood et al., 2008)). Schwenk and Koehn (2008) also compare perplexity to empirical BLEU scores, and find that the empirically best interpolation coefficient is not necessarily the one with minimal perplexity (although the difference between the two is small). Similarly, Haddow (2013) finds that discriminatively optimizing interpolation weights outperforms a perplexity optimization in one of the two evaluated data sets.

While perplexity does not guarantee optimal performance, it has the advantage of being cheap to compute and optimize, even for a high number of component models. Based on equation 7, we formulate our optimization problem as follows:

$$\hat{\lambda} = \arg\min_{\lambda} -\sum_{x,y} \frac{1}{N} \log_2 p(x|y;\lambda) \tag{9}$$

[15] The lexical weights are not true probability distributions, but are derived from word translation probabilities. Thus, we also optimize perplexity for lexical weights.

We can fill in equations 1 or 4 for $p(x|y;\lambda)$, and $(\bar{s},\bar{t})$ or $(\bar{t},\bar{s})$ for (x,y). The optimization itself is convex and can be done with off-the-shelf software. We use the L-BFGS algorithm with numerically approximated gradients (Byrd et al., 1995).

6.3 Evaluation of Domain Adaptation Techniques

In this section, we perform a systematic comparison of the domain adaptation methods that we discussed so far. We perform all experiments on two data sets, but vary some experimental conditions in order to evaluate performance under heterogeneous conditions.

6.3.1 Data and Methods

The general training and evaluation procedure is described in section 4.1. Log-linear combination of models and combination through alternative paths are optimized discriminatively, and this capability is built-in in Moses. To make sure that decoding is performed on the union of phrase pairs, we gave a back-off probability of 0.001 to all phrase pairs which do not exist in a model, for all features except the (constant) phrase penalty.

For all methods with perplexity minimization, we use our own implementation *tmcombine*.[16] For data selection, we use XenC (Rousseau, 2012), which implements cross-entropy difference data selection as described by Moore and Lewis (2010). We use the bilingual application of the approach as in (Axelrod et al., 2011). The amount of selected data is determined by picking the dataset that minimizes cross-entropy. For the priority merge, we use the implementation provided to Moses by Bisazza et al. (2011).

As first data set, we use *SAC_145* as in-domain corpus, *Europarl*, *Acquis* and *OpenSubtitles* as out-of-domain corpora. *SAC_test* serves as test set, *SAC_145_dev* as development set for both perplexity minimization and MERT. We use the same lexical reordering model (trained on *SAC_145*) and language model (*SAC_WMT11_interpolate_fr*) for all experiments. All phrase tables are filtered through statistical significance testing before model combination (Johnson et al., 2007).

As the second data set, we use the Haitian Creole – English data from the WMT 2011 featured translation task. It consists of emergency SMS sent in the wake of the 2010 Haiti earthquake. Originally, Microsoft Research and CMU operated under severe time constraints to build a translation system for this language pair. This limits the ability to empirically verify how much each data set contributes to translation quality, and increases the importance of

[16] The source code is available in the Moses repository `http://github.com/moses-smt/mosesdecoder`.

Data set	units	words (en)
SMS (in-domain)	16 500	380 000
Medical	1600	10 000
Newswire	13 500	330 000
Glossary	35 700	90 000
Wikipedia	8500	110 000
Wikipedia NE	10 500	34 000
Bible	30 000	920 000
Haitisurf dict	3700	4000
Krengle dict	1600	2600
Krengle	650	4200
Total train	120 000	1 900 000
Dev	900	22 000
Test	1274	25 000

Table 6.1: Parallel data sets for Haiti Creole – English translation task.

Data set	sentences	words
SMS (in-domain)	16k	380k
News	113 000k	2 650 000k

Table 6.2: Monolingual English data sets for Haiti Creole – English translation task.

automated and quick domain adaptation methods. Because of the smaller data sets, we did not perform phrase table filtering for this data set, and use a reordering model trained on all available data. The parallel data sets are listed in table 6.1, and the monolingual ones in table 6.2. The language model is adapted to the development set through linear interpolation with perplexity minimization.

To evaluate performance in more settings, we perform two manipulations of the data sets. Firstly, we additionally include results for which only 10% of parallel in-domain data is used. Note that both data sets have a relatively high ratio of in-domain to out-of-domain parallel training data (1:20 for DE–EN and 1:5 for HT–EN). Previous research has been performed with ratios of 1:100 (Foster et al., 2010) or 1:400 (Axelrod et al., 2011). Considering the ambiguity problem, we expect domain adaptation to be more important if the ratio of IN to OUT is low, because in these settings, the probability distribution of the translation model will be dominated by OUT. Also, low ratios of IN to OUT are a realistic setting in practice, and will remain so, given that the amount of freely available (out-of-domain) parallel data is steadily increasing. We refer to these two settings as *100% of IN* and *10% of IN*. Lexical reordering model and language model are the same as in the original experiments.

Secondly, we perform domain adaptation with four randomized training sets, obtained through concatenating and reshuffling the original data sets. This reflects a setting in which all data sets have a similar fitness for the target domain. There are approaches that can perform unsupervised domain adaptation on a single, heterogeneous data set (e.g. Hildebrand et al., 2005; Yamamoto and Sumita, 2007; Sennrich, 2012a), but the methods in this evaluation all operate on the granularity of pre-existing data sets or models, or, in the case of data selection, require a binary split into in-domain and out-of-domain data. We expect a simple concatenation of the data sets to be optimal in this setting, and expect that some methods perform significantly worse. For instance, alternative path decoding and priority merging do not combine the evidence of multiple models to obtain better probability estimates, which means that they rely on sparser data. The data set sizes of these random models are not uniform, but they are the same as for the original data sets. Reordering model and language model are the same as for the original experiment.

6.3.2 Results

The results are spread over various tables; for the discussion, we will proceed method by method, taking all results into account. Results for DE–FR are shown in tables 6.3 and 6.4, the former with a binary split of IN and OUT data, the latter with 4 component models. Table 6.5 shows results after reshuffling sentence pairs: this randomized setting shows how the domain adaptation methods perform when none of the component models is clearly in-domain. Tables 6.6 and 6.7 show results for the HT–EN data set, the former again with a binary split into IN and OUT data, the latter with 10 component models.

As far as the baselines are concerned, the IN systems consistently outperform the OUT systems, and the concatenation of all data outperforms the IN system in all experiments. Of course, this order is not a general rule. If we were to further decrease the amount of IN data used, we would reach a point where the OUT systems perform better, as shown in section 4.2. Also, the concatenation of all data is not guaranteed to outperform a purely in-domain system. Examples of decreasing performance are shown in figure 10.3 (p. 148), and in (Sennrich, 2012b).

A linear interpolation of the component models with uniform weights performs better than the concatenation of all data in most experiments, or just as well in the case of randomized component models. Only in the HT–EN experiment with 10 components does a concatenation of data outperform a linear interpolation with uniform weights. Again, this is an artifact of the data sets. When concatenating data, the impact of IN data on the model distribution depends on the relative amount of IN and OUT data, and is relatively low if

System	100% of IN		10% of IN	
	BLEU	METEOR	BLEU	METEOR
IN	17.9	37.0	15.7	33.5
OUT	14.8	32.3	14.8	32.3
unweighted/uniform weights				
instance weighting (concatenation)	18.3	37.3	17.1	35.4
linear interpolation (naive)	18.8	37.8	17.6	36.0
priority merge	18.7	37.8	17.3	35.9
discriminative weighting				
log-linear interpolation	18.4	37.6	17.0	35.6
alternative paths	18.6	37.8	17.4	36.0
perplexity minimization				
instance weighting	18.7	37.9	17.6	36.2
linear interpolation (naive)	18.8	37.9	17.6	36.1
linear interpolation (modified)	18.9	38.0	17.6	36.2

Table 6.3: Domain adaptation results DE–FR (*SAC_test*) with binary split into IN and OUT.

System	100% of IN		10% of IN	
	BLEU	METEOR	BLEU	METEOR
unweighted/uniform weights				
instance weighting (concatenation)	18.3	37.3	17.1	35.4
linear interpolation (naive)	18.6	37.6	17.3	35.8
discriminative weighting				
log-linear interpolation	17.5	36.4	16.2	34.5
alternative paths	18.3	37.4	16.3	35.1
perplexity minimization				
instance weighting	18.8	37.8	17.4	36.0
linear interpolation (naive)	18.5	37.7	17.3	35.9
linear interpolation (modified)	18.7	37.9	17.3	36.0

Table 6.4: Domain adaptation results DE–FR (*SAC_test*) with 4 component models.

System	BLEU	METEOR
unweighted/uniform weights		
instance weighting (concatenation)	18.4	37.3
linear interpolation (naive)	18.4	37.4
priority merge	18.2	37.2
discriminative weighting		
log-linear interpolation	18.2	37.1
alternative paths	18.0	36.9
perplexity minimization		
instance weighting	18.4	37.3
linear interpolation (naive)	18.5	37.3
linear interpolation (modified)	18.4	37.4

Table 6.5: Domain adaptation results DE–FR (*SAC_test*) with 4 randomized component models.

the IN data set is small. Thus, the impact of IN data in a linear interpolation with uniform weights may be higher (or lower) than in a concatenation.

The **priority merge** was only evaluated for the binary setting, giving priority to IN over OUT, and in the randomized setting, giving priority to the models with the lowest perplexity on the development set.[17] The priority merge is among the best methods for HT–EN, and also outperforms the baseline for DE–FR, albeit with slightly lower performance than the best adaptation methods. In the randomized setting, however, it performs 0.2 BLEU points worse than the baseline, the concatenation of all data (18.2 BLEU vs. 18.4 BLEU). This illustrates a weakness of the priority merge, namely that it uses the evidence of one translation model at most for each source phrase, which means its probability estimates are based on less data, and thus more susceptible to data sparseness, than if we consider evidence from all models. The same problem is discussed in chapter 7.

The two methods with **discriminative weighting through MERT**, a log-linear interpolation and alternative paths decoding, give mixed results. While they do outperform the baseline for DE–FR with a binary split into IN and OUT, performance is worse with 4 component models, both with the original data sets and the randomized data sets. Also, none of the discriminatively weighted systems is better than the baseline for HT–EN. Performance drops as the number of component models increases.

Table 6.8 shows results with **discriminative weighting with PRO**, with the results with MERT as comparison. The results show that some of the performance loss, particularly for HT-EN with 10 component models, can indeed be attributed to search errors with MERT.

[17] The model with the lowest perplexity happens to be the one that is trained on the least amount of data.

System	100% of IN		10% of IN	
	BLEU	METEOR	BLEU	METEOR
IN	33.4	31.7	29.7	28.6
OUT	28.9	30.2	28.9	30.2
unweighted/uniform weights				
instance weighting (concatenation)	33.6	32.4	31.3	31.3
linear interpolation (naive)	33.7	32.6	32.0	31.4
priority merge	33.8	32.4	31.8	31.2
discriminative weighting				
log-linear interpolation	29.6	31.6	29.6	30.6
alternative paths	33.2	32.4	29.8	30.7
perplexity minimization				
instance weighting	33.8	32.4	31.5	31.3
linear interpolation (naive)	33.7	32.4	31.9	31.3
linear interpolation (modified)	33.7	32.4	31.7	31.2

Table 6.6: Domain adaptation results HT–EN with binary split into IN and OUT. Domain: emergency SMS.

System	100% of IN		10% of IN	
	BLEU	METEOR	BLEU	METEOR
unweighted/uniform weights				
instance weighting (concatenation)	33.6	32.4	31.3	31.3
linear interpolation (naive)	33.4	32.3	31.6	31.3
discriminative weighting				
log-linear interpolation	27.9	30.2	18.4	27.5
alternative paths	24.3	29.1	29.8	30.9
perplexity minimization				
instance weighting	33.5	32.3	31.8	31.5
linear interpolation (naive)	33.8	32.4	31.9	31.3
linear interpolation (modified)	33.8	32.5	32.1	31.5

Table 6.7: Domain adaptation results HT–EN with 10 component models. Domain: emergency SMS.

System	DE–FR		DE–FR (random)		HT–EN	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
MERT						
concatenation	18.3	37.3	18.4	37.3	33.6	32.4
log-linear interpolation	17.5	36.4	18.2	37.1	27.9	30.2
alternative paths	18.3	37.4	18.0	36.9	24.3	29.1
PRO						
concatenation	18.1	37.0	18.2	37.0	33.0	31.7
log-linear interpolation	18.1	37.2	18.1	36.9	31.2	30.5
alternative paths	18.4	37.3	17.7	36.7	32.9	31.7

Table 6.8: Domain adaptation for discriminative weighting with MERT and PRO, and concatenation as baseline. DE–FR with 4 component models; HT–EN with 10 component models.

Performance of PRO degrades more slowly as the number of features increases, as shown in (Hopkins and May, 2011). While performance for alternative paths drops by 9.3 BLEU points with MERT, there is no significant decrease in BLEU with PRO. There are still experiments in which discriminative weighting results in a loss of performance, e.g. alternative paths with randomized models, and log-linear interpolation for HT–EN. Only one of the systems outperforms the concatenation baseline, namely alternative paths for DE–FR. Thus, while PRO mitigates some of the performance losses that we have observed, we still do not see the performance improvements that we would hope for, and observed with other adaptation techniques.

The methods with **perplexity minimization** perform well in all evaluations: none of the other systems significantly outperformed the best perplexity minimization system in any of the experiments. For DE–FR with 2 component models, the improvement over the baseline is between 0.4 and 0.6 BLEU points. In the randomized setting, and for HT–EN with 100% of IN, the systems with optimized weights do not outperform the baseline, but neither perform worse. Of course, we do not expect any of the evaluated domain adaptation methods to outperform the concatenation baseline in the randomized setting, since none of the models has a higher fitness for the target domain, and any bias towards one model would be unwarranted. Still, the fact that perplexity minimization does not perform worse than the concatenation baseline, while other methods such as a priority merge or alternative path decoding do perform worse, is an important result. It supports the use of perplexity minimization for domain adaptation in scenarios in which we do not know which model has the highest fitness for the target domain, or how large the difference in fitness is. In this case, the point of minimal perplexity for instance weighting is close to uniform weights, which

	system	selected
DE–FR	(100% of IN)	6%
	(10% of IN)	20%
	(randomized models)	100%
HT–EN	(100% of IN)	4%
	(10% of IN)	4%

Table 6.9: Data selection: amount of data selected.

means that all evidence is taken into account for the probability estimates. The reason that other methods, such a priority merge or alternative path decoding, perform worse than the concatenation baseline is that they are technically unable to join the evidence of multiple models to obtain new probability estimates for a given phrase pair.

Looking at the different combination methods that we optimized through perplexity minimization, results are inconclusive as to whether instance weighting or linear interpolation performs better. Also, we found no consistent differences between the naive version of linear interpolation and the modified one, the latter using the normalization and re-computation of lexical weights as described in section 6.1.2. The theoretical advantage of instance weighting over linear interpolation, namely that the former takes into account how well-evidenced a phrase pair is in a model, seems to play only a small role in these experiments; chapter 7 discusses a setting in which the amount of evidence in the translation models is more important.

For **data selection**, we first consider how much data from OUT is selected in each system. We select the amount of data that minimizes perplexity on the target side of the development set. For HT–EN, the amount of selected OUT data is relatively small (4%); for DE–FR, it is 6% with the full in-domain corpus, and 20% when performing data selection with only 10% of IN. When randomizing the four corpora, the point of lowest perplexity uses 100% of training data, which is the expected behaviour if there is no in-domain subset. We argue that using only a subset of training data is inherently more risky for translation modelling than for language modelling[18], since by discarding parallel data, we exacerbate the unknown word problem.[19]

We evaluate two ways of integrating the selected OUT data into the translation system which have been discussed in (Axelrod et al., 2011): a concatenation with IN, and a (naive) linear interpolation with IN, using perplexity minimization to optimize weights. Addition-

[18]The unknown word problem is a much smaller problem for language modelling, since words that are unknown to the translation model cannot be translated, while language modelling only affects the score.

[19]There are alternative data selection algorithms that explicitly select sentences which contain words that do not occur in IN (Banerjee et al., 2012), and should thus be more robust towards this problem.

System	100% of IN		10% of IN	
	BLEU	METEOR	BLEU	METEOR
IN	17.9	37.0	15.7	33.5
OUT	14.8	32.3	14.8	32.3
concatenation (IN+OUT)	18.3	37.3	17.1	35.4
concatenation (IN+selected)	18.7	37.5	17.2	35.5
linear interpolation (IN+selected)	18.6	37.6	17.3	35.7
linear interpolation (IN+selected+deselected)	18.7	37.9	17.7	36.3
linear interpolation (IN+10 clusters)	18.9	38.1	17.7	36.3

Table 6.10: Data selection: translation results DE–FR.

System	100% of IN		10% of IN	
	BLEU	METEOR	BLEU	METEOR
IN	33.4	31.7	29.7	28.6
OUT	28.9	30.2	28.9	30.2
concatenation (IN+OUT)	33.6	32.4	31.3	31.3
concatenation (IN+selected)	33.6	31.9	30.1	29.4
linear interpolation (IN+selected)	33.4	31.8	30.2	29.3
linear interpolation (IN+selected+deselected)	33.7	32.4	31.7	31.3
linear interpolation (IN+10 clusters)	33.8	32.4	31.7	31.2

Table 6.11: Data selection: translation results HT–EN.

ally, we propose to re-integrate the data discarded by data selection into the system. The idea is that this gives us the benefit of mitigating the unknown word problem, but that we can set weights low enough that this data has no major effect on the probability distribution. We perform two experiments that utilize cross-entropy difference to split the OUT corpus into a number of subcorpora, which are then combined through linear interpolation with perplexity minimization. The first experiment uses a translation model trained on IN, one trained on the selected portion of OUT, and one trained on the remainder of OUT as its three component models (*IN+selected+deselected*). For the second experiment, we sort OUT according to the cross-entropy difference and split it into 10 parts of equal size. We then perform a linear interpolation of these 10 component models, plus the IN model (*IN+10 clusters*).

In tables 6.10 and 6.11, we show the results of the data selection experiments. No results for the randomized setting are shown, since they are identical to not performing data selection at all. In our experiments, data selection is competitive compared to the concatenation baseline that uses all OUT data. Only in the case with an extremely small amount of IN data, namely HT–EN with 10% of available IN data, is there a large drop compared to using all OUT data. However, data selection remains below the other domain adaptation methods in terms of performance. Consistently, a linear interpolation with all data outperforms a linear interpolation which only uses IN and the selected data from OUT.

Segmenting OUT into ten components according to cross-entropy difference did not yield any significant and/or consistent performance improvements over the binary setting with IN and OUT. One explanation is that in all our interpolation experiments, most of the weight mass is concentrated on the IN model, and even the decile from OUT that is most similar to IN receives a low weight. Thus, segmenting OUT first has little effect on the final model distribution and system performance. While it was unsuccessful in our experiments, we consider that cross-entropy difference still has potential applications for translation model training – apart from its uses for data selection for language modelling, which we do not question. Firstly, the unknown word problem is more prevalent in sparse conditions; data selection is thus more suitable for well-resourced settings in which discarding data does not increase data sparseness noticeably, but reduces the resource requirements of translation modelling. Secondly, the fact that even the top decile of OUT is very dissimilar to IN, and assigned a low weight in model combination, need not be generally true. In other settings, cross-entropy difference may be a fruitful way to find relevant data in OUT.

We finish the evaluation of domain adaptation techniques with two examples that illustrate the effect of preferring IN over OUT, shown in table 6.12. In the first example, the IN system has two main problems: an unknown word (*Vallot-Hütte*), and *schön* not being

System	sentence
Source:	**Schön brav** steigen wir zu Fuss zur Vallot-Hütte.
Reference:	**Bien sages**, nous descendons à pied jusqu'au refuge Vallot.
	***Very well-behaved**, we descend on foot to the Vallot cabin.*
IN:	**Il fait beau**, nous montons **gentiment** à pied à la Vallot-Hütte.
OUT:	**Bien sage** de grimper à pied pour nous Vallot-Hütte.
concatenation:	**Bien sage**, nous montons à pied à Vallot-Hütte.
instance weighting:	**Bien sage**, nous montons à pied la Vallot-Hütte.
Source:	20. Juni: unser dritter **Angriff** auf das Gross Grünhorn (4044 m).
Reference:	Le 20 juin eut lieu notre troisième **tentative** au Gross Grünhorn (4044 m).
	*June 20th: our third **climbing attempt** at the Gross Grünhorn (4044 m).*
IN:	20 juin: notre troisième **tentative** vers le Gross Grünhorn (4044 m).
OUT:	20 juin: notre troisième **atteinte** au Gross Grünhorn (4044 m).
concatenation:	20 juin: notre troisième **attaque** contre le Gross Grünhorn (4044 m).
instance weighting:	20 juin: notre troisième **tentative** sur le Gross Grünhorn (4044 m).

Table 6.12: Mixture modelling: example translations

translated as an intensifier of *brav*.[20] The OUT system offers a better translation of *schön brav*, but is disfluent otherwise. In the combined system, the translation of *schön brav* from OUT is combined with the IN translation of *steigen* to produce a better, but still not perfect translation. In the second example, the IN system already produces an acceptable translation. The OUT system, however, introduces a wrong translation of German *Angriff* into *atteinte*, a translation predominant in Europarl, and used in contexts such as *une atteinte aux droits de l'homme* (English: *an attack on human rights*). The concatenated system produces *attaque*, which is a more basic translation of *Angriff*, and occurs in both IN and OUT. Finally, upweighting the IN corpus through instance weighting restores *tentative*, which is the predominant French expression for *climbing attempt*, as most probable translation.

In conclusion, we found that all domain adaptation methods outperform a simple concatenation of all data in at least one setting, on the DE–FR data set with a binary split into IN and OUT data. However, one of the motivations for the experiments was the idea of an SMT architecture that consists of multiple models, which supports the adaptation to a new domain online, given a small development set. For such an architecture, we deem it desirable that a domain adaptation method works without a binary distinction of IN and OUT data, and does not regress below the performance of the baseline model, a concatenation of all data, even when none of the models is particularly fitted to the target domain. Linear interpolation and instance weighting with perplexity minimization fulfilled these requirements in our

[20] A third problem, namely that *steigen* is underspecified and that in this case refers to a *descent* rather than a *climb*, is beyond the ability of even human translators to resolve without situational knowledge.

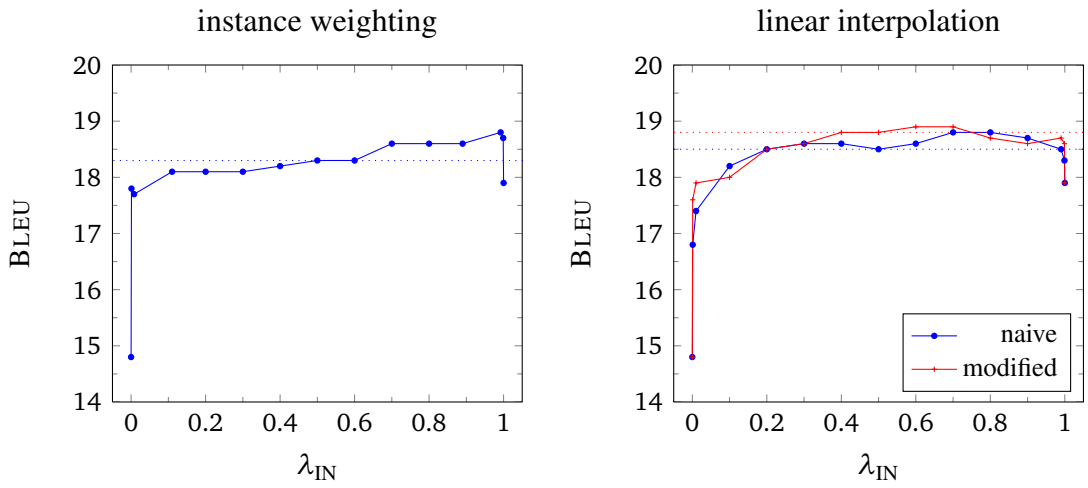

Figure 6.1: SMT performance DE–FR with weighted combination of translation models. $\lambda_{OUT} = 1 - \lambda_{IN}$. 1 MERT run per data point.

experiments, while other adaptation methods, namely alternative path decoding, a log-linear interpolation, and a priority merge, did not.

6.4 The Impact of Weights

In the evaluation, we found that the models weighted through perplexity minimization outperformed the unweighted baselines by 0.4 BLEU for instance weighting and the DE–FR data set, but for linear interpolation (table 6.3), both the unweighted and the weighted systems performed equally well. How important it is to weight translation models non-uniformly depends on the data set, and on whether other model components are adapted to the domain. In (Sennrich, 2012b), we report results with out-of-domain language models, where the effect of perplexity minimization is twice as large in terms of BLEU gain, and for HT–EN, we observe larger gains over the concatenation baseline in the setting that uses only 10% of in-domain parallel data.

To illuminate the behaviour of perplexity minimization, and the effect of model weights for both instance weighting and linear interpolation, we present the results of some additional experiments. In a first set of experiments, the models are the same as in the DE–FR experiment of the previous section, i.e. *SAC_145* as IN, and a concatenation of *Europarl*, *Acquis* and *OpenSubtitles* as OUT.

We tested two combination methods, instance weighting and linear interpolation, with different weights.[21] The resulting graphs are shown in figure 6.1. At $\lambda_{\text{IN}} = 0$, we show performance when only using OUT, and at $\lambda_{\text{IN}} = 1$, performance when only using IN. All other data points represent systems that contain the same phrase pairs, namely the union of the two translation models IN and OUT. For our data set, using the union of phrase pairs rather than a pure OUT model brings the biggest performance boost, even if the IN model is given a weight of 0.001, the most extreme weight that we evaluated. The boost is bigger for instance weighting and the normalized ('modified') version of linear interpolation[22] (2.8–3 BLEU) than for the unnormalized ('naive') version linear interpolation (2 BLEU). We attribute this to the fact that choosing a probability of 0 (or a low uniform backoff probability), which we do in the naive linear interpolation, is a poor estimate for phrase pairs that do not occur in OUT, whereas instance weighting and normalized linear interpolation distinguish between negative evidence and lack of evidence. We see a similar, but smaller boost on the other end of the graph, when going from a pure IN model to the union of phrase pairs.

The potential improvement over uniform weights ($\lambda_{\text{IN}} = 0.5$) is very small for linear interpolation (0.1–0.3 BLEU), and slightly larger for instance weighting (0.5 BLEU). This puts in perspective the results of the perplexity minimization experiments in table 6.3. On this data set, uniform weights happen to be near-optimal for linear interpolation, and in fact, the weights resulting from perplexity minimization are close to uniform: for the feature $p(\bar{t}|\bar{s})$, λ_{IN} is 0.65 for naive linear interpolation, and 0.54 for modified linear interpolation.[23] For the instance weighting approach, perplexity minimization finds the optimal λ_{IN} to be 0.87.

There are several conclusions that we can draw from the graphs in figure 6.1. First, note that different weights are optimal for instance weighting and linear interpolation. While we see the best performance for linear interpolation with λ_{IN} between 0.4 and 0.8, higher weights for IN are optimal for instance weighting, achieving the best performance with λ_{IN} between 0.7 and 0.999. Optimizing translation model perplexity allowed us to find different optima for these two mixture operations, in contrast to a more indirect measure such as language model perplexity. In past research, language model perplexity has been used both to set linear interpolation weights (Foster and Kuhn, 2007) and instance weights (Shah et al., 2012).

[21] We used the same weight vector λ for all 4 translation model features.

[22] See section 6.1.2.

[23] Figure 6.1 is slightly simplified because it only uses one independent variable λ_{IN}, whereas in the previous section, we calculate a separate weight vector λ for each translation model feature during perplexity optimization.

A second conclusion is that the weights obtained through perplexity minimization are adequate, and that there is little room for improvement through other optimization methods, at least on this data set. Instead, we hypothesize that there is room for improvement through more fine-grained optimization methods. Foster et al. (2010) operate on the level of phrase pairs, and data selection methods such as (Yasuda et al., 2008) operate on a sentence level. Both rely on the in-domain training set being known beforehand. We will pursue a different research direction and investigate increasing the granularity of domain adaptation methods through unsupervised means in section 6.5.

For a second analysis of the impact of giving high weights to IN data, we go back to the SMT learning curves that we introduced and described in more detail in section 4.2. Figure 6.2 shows an SMT learning curve that, for every combination of IN and OUT data, not only shows BLEU scores achieved through a concatenation of the data (black), but also BLEU scores with instance weighting and weights obtained through perplexity minimization (red). The graph shows the improvement from perplexity optimization for various settings.

Trivially, the uniform and optimized learning curves coincide at the upper bound, which is formed from pure IN models, and the lower bound, trained on OUT models, since there is nothing to be weighted there. If IN data forms the majority, as in the upper-left corner of the graph, we observe no gain from weighting IN data even higher. At this point, the amount of OUT data is too small to disturb the model probabilities and cause ambiguities. In the bottom-right corner of the graph, where we find models with 1 million sentences of OUT and 1000-2000 sentences of IN data, perplexity minimization has a similarly small effect. This is slightly counter to our expectations that perplexity minimization would be more important if the ratio of IN to OUT is low. However, if we consider that the IN data set is relatively sparse, with a small vocabulary and relatively poor probability estimates, it is understandable that increasing its weight has a limited effect on the model. For these small data sizes, the positive effect of IN data, which is surprisingly large, seems to be mostly due to the improved translation of phrases for which there is little to no evidence in OUT. The biggest and most consistent improvements over the baseline are to be found in the top right corner of the graph, with the weighted systems performing about 0.5 BLEU better than the uniformly weighted ones. Of course, this corner, indicating systems that use the most data and perform best in terms of BLEU, is also the most relevant.

The graph again makes the point that perplexity minimization, or upweighting IN data in general, does not improve performance for all conceivable data sets. However, it also shows that the problem that is tackled by instance weighting, namely the ambiguity problem, is not one of data sparseness; the graph thus supports speculation that domain adaptation will

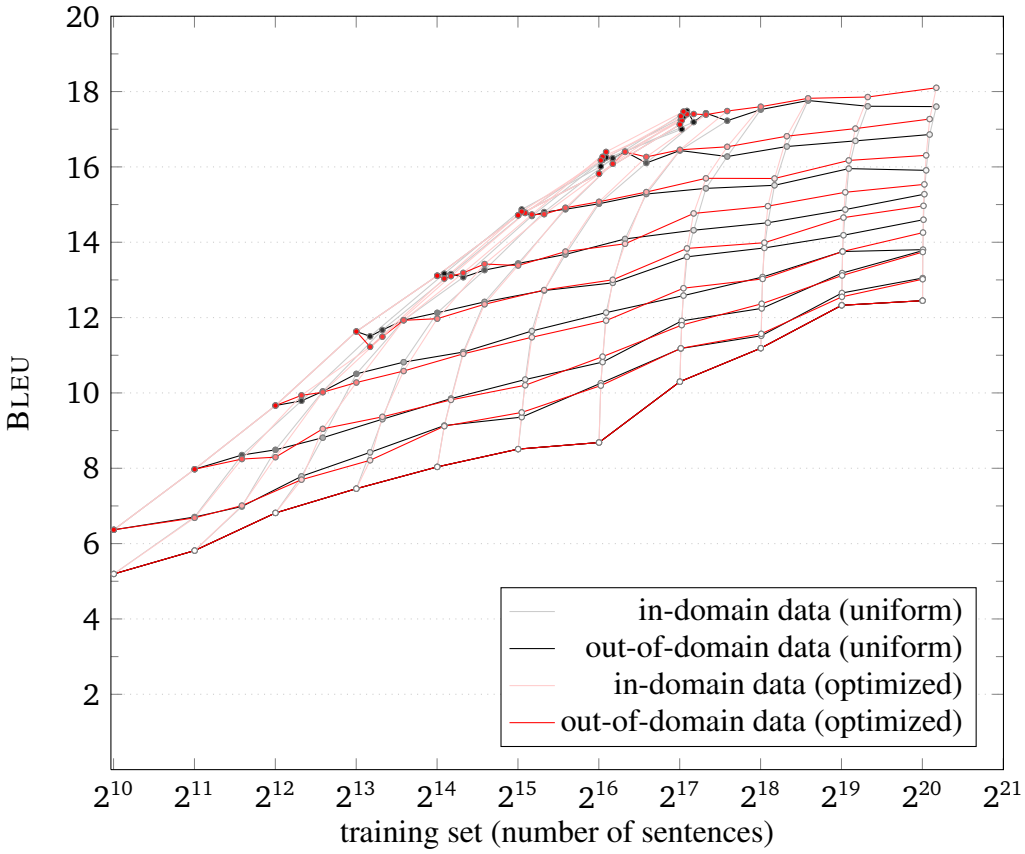

Figure 6.2: SMT learning curve with uniform corpus weights, and with instance weighting with perplexity minimization. Translation direction DE–FR.

remain relevant, or become even more so, as the amount of available training data for SMT increases.

6.5 Domain Adaptation with Unsupervised Clustering of Training Data

So far, most domain adaptation methods that we evaluated operated on a corpus level. We assigned corpus-level weights with perplexity minimization or discriminative weighting, and defined a corpus-level precedence for the priority merge. Data selection (Moore and Lewis, 2010) is the only method that we evaluated that operates on a sentence level, but it only does so for out-of-domain data, and treats the in-domain corpus as an atomic unit. There are other approaches that adapt translation models with a finer granularity, e.g. (Foster et al., 2010), who train a classifier to assign a weight to each phrase pair in OUT. However, since the classifier relies on a-priori knowledge about IN and OUT at training time, it is not straightforward to consolidate such an approach with our goal of performing domain adaptation with little a-priori knowledge, and ideally for multiple target domains in a single system.

In this section, we investigate if we can obtain more fine-grained weighting of the training data by first performing unsupervised clustering. This has two main motivations: on the one hand, we want to be able to perform domain adaptation even if in-domain and out-of-domain data are not separated into two training corpora, but if the training corpus is a heterogeneous mix of in-domain and out-of-domain data. An example of such a heterogeneous resource would be a web crawl corpus. On the other hand, even if we do have a clear separation of in-domain and out-of-domain data, the boundaries between the two are fluent, and there may be subdomains within the out-of-domain corpus with a higher fitness for the target domain, which we thus want to weight accordingly.

Our basic approach is divided into two steps. First, we perform unsupervised clustering on the parallel training data to obtain a given number of clusters. Then, we perform perplexity minimization to compute a model from these clusters that is adapted to the development set. The method is most closely related to experiments by Yamamoto and Sumita (2007), who also perform unsupervised clustering of training data. However, they use unsupervised domain prediction at decoding time to translate each test set sentence by a cluster-specific model. We will assume a scenario in which the target domain remains stable, and will compute a single adapted model to translate the whole test set.

6.5.1 Clustering with Exponential Smoothing

We follow the clustering algorithm described in (Yamamoto and Sumita, 2007). It is a variant of k-means-clustering, which uses language models as centroids, trained on all sentences in a cluster, and the language model entropy as the distance between each sentence and cluster.

The algorithm is initialized with randomly generated clusters, and can be expanded to clustering sentence pairs by taking the sum of the distance on both language sides. In terms of n-gram length, we follow Yamamoto and Sumita (2007) and use unigram models.

One drawback of sentence-level clustering is that cluster assignment is made on the basis of little information, i.e. the sentence itself. If we assume that the domain of a text does not rapidly change between sentences, it is sensible to consider neighbouring sentences for cluster assignment.

We achieve this by using an exponentially decaying score for cluster assignment. In the baseline without exponential decay (equation 10), we assign the sentence pair i to the cluster c that minimizes the distance (LM entropy).

$$\hat{c}_i = \arg\min_c d(i,c) \tag{10}$$

In equation 11, the distance of sentence pair i to cluster c is the weighted average of the distance of each sentence j to c, with the weight exponentially decaying as the textual distance between i and j increases, and with the decay factor λ determining how fast the weight decays.

$$\hat{c}_i = \arg\min_c \sum_{j=1}^{n} d(j,c) \cdot \lambda^{|i-j|} \tag{11}$$

Note that the equation is two-sided, meaning that both previous and subsequent sentences are considered for the assignment.

Algorithmically, two-sided exponential smoothing only slows down cluster assignment by a constant factor; we do not need to sum over all sentences for each assignment, but can store the weighted distance of all previous sentences in a single variable. Algorithm 1 shows the smoothed assignment step for n sentences and k clusters.

Note that the decay factor λ determines the extent of smoothing, i.e. how strongly context is taken into account for the assignment of each sentence. A decay factor of 0 corresponds to the unsmoothed sentence-level score (with $0^0 = 1$). With a decay factor of 1, the algorithm returns the same distance for all sentence pairs. We use a decay factor of 0.5 throughout the experiments. This is a relatively fast decay: one third of the score is determined by the

Algorithm 1 Cluster assignment with decay

Ensure: $0 \leq decay \leq 1$

let $d(x, y)$ be a distance function for a sentence x and a centroid y
let $d\_min[n]$,$d\_curr[n]$,$\hat{c}[n]$ be arrays
set all elements of d_min to ∞
for $c = 0$ **to** k **do**
    $cache \leftarrow 0$
    set all elements of d_curr to 0
    for $i = 0$ **to** n **do**
        $cache \leftarrow decay * cache$
        $cache \leftarrow cache + d(i, c)$
        $d\_curr[i] \leftarrow cache$
    end for
    $cache \leftarrow 0$
    for $i = n$ **to** 0 **do**
        $cache \leftarrow decay * cache$
        $d\_curr[i] \leftarrow d\_curr[i] + cache$
        if $d\_curr[i] < d\_min[i]$ **then**
            $d\_min[i] \leftarrow d\_curr[i]$
            $\hat{c}[i] \leftarrow c$
        end if
        $cache \leftarrow cache + d(i, c)$
    end for
end for

sentence itself; two thirds by the sentence and its two neighbouring sentences. What decay factor is optimal may depend on the properties of the text, i.e. how quickly documents and/or domains change, so we will not evaluate different decay factors in this thesis.

We could extend the algorithm to reset the cache to 0 whenever we cross a known document boundary, and thus implement document-level scoring (with a decay factor of 1), or a hybrid (with a decay factor between 0 and 1). We did not do this since we want to demonstrate that the approach does not require document boundaries in the training text. Another point to note is that we slightly modify the LM entropy method by normalizing entropy by sentence length, which ensures that longer sentences have no inflated effect on their neighbours' cluster assignment.[24]

6.5.2 Model Combination

Having divided the parallel and monolingual corpus into k clusters, we contrast two different model combination methods. The first is a reimplementation of (Yamamoto and Sumita, 2007), i.e. a domain-prediction module and cluster-specific models. To combat data sparseness, they interpolate each cluster-specific model with a general model, using a constant interpolation coefficient. Potential problems with their approach, discussed and evaluated in more detail in (Sennrich, 2012a), are the lack of model optimization, and the inability to select multiple clusters as being relevant for translation. As an alternative model combination method, we propose and evaluate linear interpolation with perplexity minimization, both for the translation model and the language model.

6.5.3 Evaluation

We use the same DE–FR parallel data set as in the evaluation in section 6.3, namely *SAC_145* as in-domain corpus, *Europarl*, *Acquis* and *OpenSubtitles* as out-of-domain corpora. The only difference is that we held out 20 000 additional sentences of in-domain data during training, and used two different development sets, 1000 sentences each, for perplexity minimization and MERT. We concatenate all parallel data together before clustering to simulate a heterogeneous training set.

For language modelling, we use the parallel data sets for DE and FR (for domain prediction), and for decoding, additionally *News* and the monolingual French part of *SAC_145*. We used the following language model settings:

[24] We made the implementation of this clustering algorithm available as free software on `https://github.com/rsennrich/bitext_clusterer`.

system	BLEU	METEOR
general	18.5	37.3
adapted TM	18.8	37.8
adapted LM	18.8	37.8
adapted TM & LM	18.6	37.9

Table 6.13: Baseline SMT results DE–FR. Concatenation of all data and using domain adaptation with original four data sets.

clustering	model combination	adapted TM		adapted TM & LM	
		BLEU	METEOR	BLEU	METEOR
$k = 10$					
sentence-level	Yamamoto/Sumita	18.7	37.6	18.9	37.9
sentence-level	perplexity	18.8	38.0	18.9	38.2
smoothed	perplexity	19.1	38.3	19.2	38.3
$k = 100$					
sentence-level	Yamamoto/Sumita	18.8	37.7	18.6	37.6
sentence-level	perplexity	19.0	38.0	19.0	38.3
smoothed	perplexity	19.1	38.1	19.1	38.2

Table 6.14: SMT results DE–FR based on clustered training data.

- for clustering, unigram language models.
- for domain prediction, used only for our implementation of (Yamamoto and Sumita, 2007), 3-gram language models with Good-Turing smoothing.
- for translation, 5-gram language models with interpolated Kneser-Ney smoothing.

We clustered additional target language data with the method described in (Yamamoto and Sumita, 2007), i.e. assigning each sentence to the closest cluster, except if a general LM is closest, in which case the sentence is discarded.

Since the training and tuning data is slightly different from the experiments in section 6.3, table 6.13 shows new baseline results for perplexity minimization with the four original data sets. The general system (without domain adaptation) performs worst, with a BLEU score of 18.5 and a METEOR score of 37.3. Both translation model (TM) and language model (LM) adaptation significantly increase scores by 0.3 BLEU and 0.5 METEOR points. The system that combines TM and LM adaptation is not significantly different from the systems with only one model adapted in terms of BLEU, but performs best in terms of METEOR (0.6 points better than the general model).

Table 6.14 shows SMT performance when domain adaptation is not performed on the original four data sets, but on 10 or 100 unsupervised clusters. The best system, with expo-

nential smoothing for clustering ($k = 10$) and perplexity minimization for both the translation and the language model, achieves 19.2 BLEU and 38.3 METEOR, 0.6 BLEU and 0.4 METEOR better than adaptation with the original 4 data sets. This is a success on two levels. Not only do the results demonstrate that we can perform domain adaptation without a prior division of the training data into different domains, the best system also outperforms domain adaptation results on the original four data sets. This indicates that the ability to set more fine-grained weights can indeed improve performance. We also demonstrated an increase in performance compared to the domain prediction method that was proposed by Yamamoto and Sumita (2007). When adapting to a known target domain, and with access to development data from this domain, perplexity optimization has advantages over a sentence-level domain prediction. Namely, it is not limited to selecting a single cluster per sentence, but can assign high weights to multiple clusters. Increasing the number of clusters to 100 did not improve performance any further; however, neither did we observe performance regressions due to overfitting or data sparseness. Technically, we have demonstrated perplexity minimization to be feasible for a large number of component models.

There is a small performance difference between clustering with exponential smoothing and clustering on a sentence level (without smoothing). Sennrich (2012a) also performs a perplexity-based evaluation of the resulting clusters, which shows that the resulting clusters tend to be more homogeneous if exponential smoothing is applied. By homogeneous we mean that sentences from the same original data set, i.e. sentences which occur close together, are more likely to be clustered together with smoothing. An additional positive effect of exponential smoothing is that it decreased the average number of iterations of the clustering algorithm by a factor of 2–4 in our experiments.

The approach by Yamamoto and Sumita (2007) has the advantage that it can, in principle, support multiple target domains, since it builds one system per cluster, whereas we build a single model adapted to the target domain. We will now discuss how it is is also possible for mixture models with perplexity optimization to support multiple target domains.

6.6 A Multi-Domain Translation Model Architecture

In the beginning of this chapter, we described the goal of having an SMT system that is quickly adaptable to a new domain online, i.e. while the system is running, and that supports multiple domains. Some of our experiments were tailored to test the risks of such an architecture, for instance suboptimal performance when none of the component models is clearly better fitted to the target domain than the others. In this section, we will describe an architecture that supports both online adaptation to new domains, and multi-domain trans-

lation. The basic idea is to delay instance weighting to the decoding phase, storing all sufficient statistics for the calculation in a vector of component models. This allows us to change the target domain at runtime by simply changing the vector of instance weights.[25]

Most similar to our architecture is the ensemble decoding framework described by Razmara et al. (2012), which combines several translation models in the decoding step, using one of several mixture operations such as linear interpolation or a weighted maximum. Our work is similar to theirs in that the combination is done at runtime, but we also delay the computation of translation model probabilities, and thus have access to richer sufficient statistics. In principle, our architecture can support all mixture operations that Razmara et al. (2012) describe, plus additional ones such as forms of instance weighting, which are not possible after the translation probabilities have been computed.

Ortiz-Martínez et al. (2010) also delay the computation of translation model features, namely for the purpose of interactive machine translation with online training. The main difference is that we store sufficient statistics not for a single model, but a vector of models, which allows us to weight the contribution of each component model to the feature calculation. The similarity suggests that our framework could also be used for interactive learning, with the ability to learn a model incrementally from user feedback, and weight it differently than the static models, which opens new research opportunities.

Our translation model is embedded in a log-linear model as is common for SMT, and is treated as a single translation model in this log-linear combination. We implemented this architecture for phrase-based models, and will use this terminology to describe it, but in principle, it can be extended to hierarchical or syntactic models.

The architecture has two goals: move the calculation of translation model features to the decoding phase, and allow for multiple knowledge sources (e.g. bitexts or user-provided data) to contribute to their calculation. Our main motivation is domain adaptation in a multi-domain environment, but the delay of the feature computation has other potential applications, e.g. in interactive MT.

We are concerned with calculating four features during decoding, henceforth just referred to as the translation model features: $p(\bar{s}|\bar{t})$, $lex(\bar{s}|\bar{t})$, $p(\bar{t}|\bar{s})$ and $lex(\bar{t}|\bar{s})$. $\bar{s}$ and $\bar{t}$ denote the source and target phrase. We follow the definitions in (Koehn et al., 2003).

Traditionally, the phrase translation probabilities $p(\bar{s}|\bar{t})$ and $p(\bar{t}|\bar{s})$ are estimated through unsmoothed MLE.

$$p(x|y) = \frac{c(x,y)}{c(y)} = \frac{c(x,y)}{\sum_{x'} c(x',y)} \tag{12}$$

[25] We have made this architecture available as part of the Moses SMT toolkit (`http://www.statmt.org/moses/`).

where c denotes the count of an observation, and p the model probability.

The lexical weights $lex(\bar{s}|\bar{t})$ and $lex(\bar{t}|\bar{s})$ are calculated as follows, using a set of word alignments a between $\bar{s}$ and $\bar{t}$:[26]

$$lex(\bar{s}|\bar{t}, a) = \prod_{i=1}^{n} \frac{1}{|\{j|(i,j) \in a\}|} \sum_{\forall(i,j)\in a} w(s_i|t_j) \tag{13}$$

A special NULL token is added to $\bar{t}$ and aligned to each unaligned word in $\bar{s}$. $w(s_i|t_j)$ is calculated through MLE, as in equation 12, but based on the word (pair) frequencies.

To combine statistics from a vector of n component corpora, we use a weighted version of equation 12, which adds a weight vector λ of length n:[27]

$$p(x|y;\lambda) = \frac{\sum_{i=1}^{n} \lambda_i c_i(x,y)}{\sum_{i=1}^{n} \sum_{x'} \lambda_i c_i(x',y)} \tag{14}$$

The word translation probabilities $w(t_i|s_j)$ are defined analogously, and used in equation 13 for a weighted version.

In order to compute the translation model features online, a number of sufficient statistics need to be accessible at decoding time. For $p(\bar{s}|\bar{t})$ and $p(\bar{t}|\bar{s})$, we require the statistics $c(\bar{s})$, $c(\bar{t})$ and $c(\bar{s},\bar{t})$. For accessing them during decoding, we simply store them in the decoder's data structure, rather than storing pre-computed translation model features. This allows us to use existing, compact data formats for storing and accessing them.

The statistics are accessed when the decoder collects all translation options for a phrase $\bar{s}$ in the source sentence. We then access all translation options for each component table, obtaining a vector of statistics $c(\bar{s})$ for the source phrase, and $c(\bar{t})$ and $c(\bar{s},\bar{t})$ for each potential target phrase. For phrase pairs which are not found, $c(\bar{s},\bar{t})$ and $c(\bar{t})$ are initially set to 0.

Note that $c(\bar{t})$ is potentially incorrect at this point, since a phrase pair $(\bar{s},\bar{t})$ not being found does not entail that $c(\bar{t})$ is 0. After all tables have been accessed, and we thus know the full set of possible translation options $(\bar{s},\bar{t})$, we perform a second round of lookups for all $c(\bar{t})$ in the vector which are still set to 0. We introduce a second table for accessing $c(\bar{t})$ efficiently, again storing it in the decoder's data structure. We can easily create such a table by inverting the source and target phrases, deduplicating it for compactness (we only need one entry per target phrase), and storing $c(\bar{t})$ as the only feature.

[26] The equation shows $lex(\bar{s}|\bar{t})$; $lex(\bar{t}|\bar{s})$ is computed analogously.

[27] The equation is identical in the offline version described in section 6.1.3, but repeated here for readability.

For $lex(\bar{s}|\bar{t})$, we require an alignment a, plus $c(t_j)$ and $c(s_i, t_j)$ for all pairs (i, j) in a. $lex(\bar{t}|\bar{s})$ can be based on the same alignment a (with the exception of NULL alignments, which can be added online), but uses statistics $c(s_j)$ and $c(t_i, s_j)$. For estimating the lexical probabilities, we load the frequencies into a vector of four hash tables. To give an example, for lex(Gletscher|des glaciers) with the alignment $\{(0, 1)\}$ (meaning *Gletscher* is aligned to *glaciers*, and *des* is unaligned), we require the following statistics:

- c(Gletscher, glaciers)
- c(glaciers)

For the inverse direction, i.e. lex(des glaciers|Gletscher), we need:

- c(des, NULL)
- c(NULL)
- c(glaciers, Gletscher)
- c(Gletscher)

On the basis of these statistics, we can apply (the weighted version of) equation 13 to recalculate lexical weights.

Both space and time complexity of the lookup are linear to the number of component tables. We deem it still practical because the collection of translation options is typically only a small fraction of total decoding time, whereas search makes up the largest part. For storing and accessing the sufficient statistics (except for the word (pair) frequencies), we use an on-disk data structure provided by Moses, which reduces the memory requirements. Still, the number of components may need to be reduced, for instance through clustering of training data, as described in section 6.5.

With a small modification, our framework could be changed to use a single table that stores a vector of n statistics instead of a vector of n tables. While this would be more compact in terms of memory, and keep the number of table lookups independent of the number of components, we chose a vector of n tables for its flexibility. With a vector of tables, tables can be quickly added to or removed from the system (conceivable even at runtime), and they can be polymorph. One application where this is desirable is interactive machine translation, where one could work with a mix of compact, static tables, and tables designed to be incrementally trainable.

In the unweighted variant, the resulting features are equivalent to training on the concatenation of all training data, except for differences in word alignment, pruning[28] and rounding. The architecture is thus a drop-in replacement for a baseline system that is trained on concatenated training data, with non-uniform weights only being applied for texts for which better weights have been established. This is done either using domain labels or unsupervised methods as described in (Sennrich et al., 2013).

As a weighted combination method, we implemented instance weighting as described in equation 14. In our implementation, the weight vector is set globally, but can be overridden on a per-sentence basis. We also implemented perplexity minimization online, so that a running server instance can be adapted to a new domain without the need to reload or rewrite the whole phrase table. The framework can also be extended to support other combination methods, and we implemented a linear interpolation of models.

A possible application for this online combination of models has been shown in (Sennrich et al., 2013), where it is used for multi-domain decoding. We also use the architecture in chapter 7, where we perform multi-engine MT.

6.7 Summary

In this chapter, we have discussed and evaluated ways to integrate out-of-domain data while giving preference to in-domain data for the purpose of translation modelling. Additionally to a discussion of existing approaches, we have also proposed novel methods such as perplexity minimization for instance weighting, and a normalized version of linear interpolation. The evaluation settings were chosen not only to show how the different domain adaptation methods can be successful, but also that they may perform worse than baseline systems under some circumstances, e.g. when we combine several models with comparable fitness. We have highlighted problems with approaches such as a priority merge or alternative decoding paths, and have shown that perplexity minimization performs robustly, scales well to a high number of models, and requires no knowledge about the domains involved.[29] We have also investigated how to perform domain adaptation starting from a single, heterogeneous parallel corpus. We proposed a model adaptation method that uses a combination of unsupervised clustering and perplexity minimization, and demonstrated that we can use

[28] We prune the tables to the most frequent 50 phrase pairs per source phrase before combining them, since calculating the features for all phrase pairs of very common source phrases causes a significant slow-down. We found that this had no significant effects on BLEU.

[29] The only requirement is that the development set is from the target domain. In (Sennrich et al., 2013), we show that we can use unsupervised methods if the target domain is heterogeneous or unknown.

it to adapt SMT systems to a target domain even in the absence of a prior division into in-domain and out-of-domain data. Finally, we describe a novel translation model architecture that allows for domain adaptation to be done at decoding time, where multiple domains are supported by the same SMT system.

7 Integrating Other Knowledge Sources: Multi-Engine Machine Translation

In chapter 6, we discussed the integration of out-of-domain parallel data as a way to mitigate the unknown word problem, and how to ensure that in-domain data is preferred for well-evidenced phrases.[1] There are alternative strategies that we can use to reduce the effects of data sparseness in our system. To give just a few examples, Fung and Yee (1998) and Daumé III and Jagarlamudi (2011) mine for translations in unstructured texts, and Okuma et al. (2007) integrate a dictionary into their SMT system. In this chapter, we investigate other MT systems as knowledge sources. In contrast to raw dictionaries, rule-based MT systems have the advantage that they (ideally) perform the morphological processing necessary to deal with German morphology, both its compositionality and the richness of its inflections. We also expect that popular online SMT systems are trained on large amounts of data and thus suffer less from data sparseness than a typical system trained only on the respective in-domain data, but that they are not domain-specific for most purposes. Thus, we face the same dilemma as with out-of-domain training data: other MT systems can help us reduce data sparseness, but produce translations that are less fitted to the target domain than an in-domain system. Following this analogy, we investigate how to treat other MT systems as a type of out-of-domain resource, and how to integrate them into our system using similar techniques as in chapter 6.

7.1 Related Work

System combination or multi-engine MT has been an active research field in recent years, with several shared tasks at MT workshops (Callison-Burch et al., 2009, 2010, 2011). There are multiple approaches to multi-engine MT. Several state-of-the-art systems operate purely on the target side, aligning the different hypotheses with each other to build a confusion network, and then decoding it with various target-side features, such as a language model and constant penalties for each hypothesis-generating system (Heafield and Lavie, 2010; Barrault, 2010).

[1] This section is based on work that we published in (Sennrich, 2011a,b).

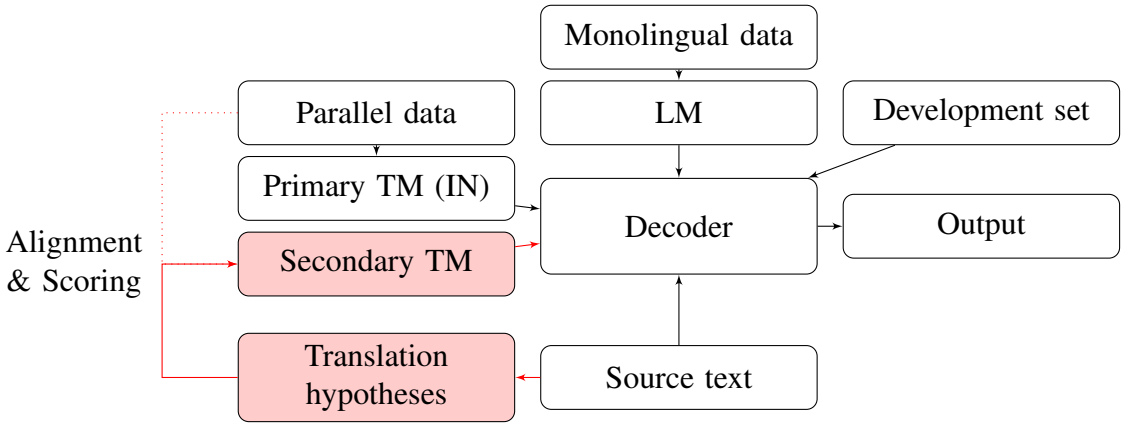

Figure 7.1: Multi-engine MT architecture. Standard components in black; extensions in red.

Such black-box approaches have the advantage of working with any MT system and requiring little parallel data (only a development set to optimize feature weights), but are blind to potentially useful information. Specifically, they do not know the source text, and how well-evidenced each source word or phrase is in the different models. Therefore, they cannot learn to generally prefer the in-domain system, but selectively use translations from an out-of-domain MT system for words that the in-domain system does not know, or for which evidence in the in-domain system is sparse.

An alternative approach is to use the individual systems' translation hypotheses, together with the original source text, as small parallel corpora, and use SMT technology to learn translation models from these corpora. These new models are then used for the final translation. The architecture by Chen et al. (2007) falls under this category. We will use this general architecture, and focus on the question of how to combine the different knowledge sources, i.e. the translation models that have been learned, for the final decoding step.

7.2 A Multi-Engine MT Architecture

Our architecture is based on a Moses SMT system, illustrated in figure 7.1. The main components (primary translation model, language model, and parameter tuning through MERT on a development set) are trained on existing corpora, and should ideally be adapted to the domain of the source text.

The translation hypotheses of other MT systems are integrated through a secondary translation model, i.e. by first aligning the translation hypotheses with the source text, calculating a new translation model, and combining it with the primary model.

The challenge in the alignment step lies in the fact that the corpus of translation hypotheses is typically small, and that word alignment needs to be fast. In a glass-box approach, we could use decoder information to obtain the alignment, but we will treat all external MT systems as a black box. A common solution is to use an existing model – possibly in a server-client architecture – to align the translation hypotheses (Chen et al., 2009; Hardt and Elming, 2010). We use MGIZA++ (Gao and Vogel, 2008), which is an extension of GIZA++ (Och and Ney, 2003), for this purpose, with a model trained on the primary corpus, performing two more iterations of IBM model 4 alignment on the translation hypotheses.

For the combination of the two translation models, different approaches have been proposed. Hardt and Elming (2010) use alternative decoding paths, while Chen et al. (2007, 2009) concatenate phrase table entries, with additional features as system markers. We propose alternative methods that aim at improving over these approaches in two respects. Firstly, when training the secondary translation model on few translation hypotheses, data sparseness is a major issue. The reliability of MLE is low when the number of observations is small, and we want to suppress spurious translation probabilities. Secondly, we want to discriminate between source words or phrases that are well-evidenced in the in-domain system, and for which the secondary translation model need not be used, and source words with little or no evidence in the in-domain system, for which the combined system shall use the hypotheses of the external MT systems.

7.3 Translation Model Combination

We take two combination methods from chapter 6 to represent existing approaches: a priority merge, which is similar to a concatenation of phrase table entries (Chen et al., 2007, 2009), and alternative paths decoding, as used by Hardt and Elming (2010). As new alternative, we propose a form of instance weighting. We will here motivate this theoretically, and proceed with an evaluation in the next section.

When estimating translation probabilities from a hypothesis corpus, data sparseness is potentially a much larger problem. This is especially true if we only want to translate a single sentence, and the hypothesis corpus is composed of only a few sentence pairs (one per hypothesis-generating MT system). If source words are only encountered a few times, we are unlikely to obtain useful probability estimates from MLE. Hence, combination methods which use the hypothesis model as a component, such as alternative path decoding or a priority merge, are sensitive to data sparseness. Taking into account the word and phrase (pair) frequencies from the in-domain corpus (IN), as we do in instance weighting, mitigates this problem, since evidence from IN can prevent spurious probability estimates.

A second motivation for instance weighting is that it implements the proposed strategy of selectively relying more on the externally provided hypotheses where evidence in IN is low. A simple example will show that the impact of the translation hypotheses on translation probabilities depends on the phrase frequency in IN. For rare or unknown phrases, observing a phrase pair in the translation hypotheses may effect a large shift in translation probabilities: $\frac{0}{2} = 0$ vs. $\frac{0+1}{2+1} = 0.33$. For phrases that are frequent in IN, the influence of the translation hypotheses is marginal, assuming uniform weights: $\frac{0}{200} = 0$ vs. $\frac{0+1}{200+1} = 0.005$.

Sennrich (2011a) proposed to perform instance weighting only for the phrase pairs in the hypothesis corpus, and then using the resulting model as an alternative decoding path to the IN model. The main motivation for this more complicated architecture was that rescoring the full IN model would be impractical at decoding time. With the online combination architecture discussed in section 6.6 (p. 96), however, we can efficiently work with a single model whose features are computed at decoding time. Still, we include this 2-step combination in our evaluation to compare its performance.

7.4 Evaluation of Multi-Engine MT

We perform experiments on the Alpine domain, with *SAC_145* as parallel corpus for translation model training, and an adapted language model (*SAC_WMT11_interpolate_fr*).[2] We use *SAC_145_dev* as development set, *SAC_test* as test set, and two external MT systems, the rule-based Personal Translator 14[3], and Google Translate[4]. The translation direction is DE–FR. Note that we perform statistical significance pruning of the translation model for the in-domain model, but not for the hypothesis models. The latter are too small for effective significance testing, especially for rare source words and phrases. Pruning the hypothesis models would thus go against our explicit goal of using them to fill in lexical gaps in the in-domain model.

Table 7.1 shows the translation results. The first set of baselines are the **individual systems**, and the system combination tools **MANY** (Barrault, 2010) and **MEMT** (Heafield and Lavie, 2010), which both operate purely on the target side. On the test domain, both Personal Translator 14 (13.1 BLEU) and Google Translate (12.8 BLEU) are markedly worse than the system trained on in-domain data (17.9 BLEU).[5] The system combination systems

[2]The setting is comparable to that in (Sennrich, 2011a), except that we use more in-domain training and development data in this evaluation.

[3]`http://www.linguatec.net/products/tr/pt`

[4]`http://translate.google.com`, translations obtained in November 2010.

[5]We are aware that automatic evaluation scores may favour SMT systems over rule-based ones.

System	weighting (step 1 / 2)	BLEU	METEOR
IN		17.9	37.0
Personal Translator 14		13.1	34.0
Google Translate		12.8	32.0
MANY		19.2	38.7
MEMT		18.3	37.1
priority merge	IN-google-pt14	19.4	38.7
priority merge (max length 3)	IN-google-pt14	19.6	38.8
alternative paths	MERT	19.7	38.9
instance weighting	uniform	19.8	39.3
instance weighting	perplexity	19.0	38.0
instance weighting / alternative paths	uniform / MERT	20.0	39.5
instance weighting / alternative paths	perplexity / MERT	19.8	39.3

Table 7.1: Multi-engine MT: SMT performance DE–FR.

MANY and MEMT both significantly outperform the best individual system, with 19.2 and 18.3 BLEU, respectively.

Next, we evaluate the combination of IN and the translation hypothesis models through a **priority merge** or **alternative path decoding**. Both methods outperform all baselines, with 19.4 BLEU for priority merge, and 19.7 BLEU for alternative path decoding. We highlighted the risk of data sparseness when training models on a small translation hypothesis corpus, and we do observe these effects in the priority merge. Long phrase pairs are typically sparser than short ones, and we decided to set a maximum length of 3 for source phrases in the translation hypothesis models. This yields an additional 0.2 BLEU improvement (from 19.4 to 19.6).

Lastly, we discuss performance of **instance weighting**. With uniform instance weights, i.e. simulating a concatenation of training data, we achieve an improvement of 1.9 BLEU over the baseline (17.9 BLEU to 19.8 BLEU), and small improvements of 0.1–0.4 BLEU over the other combination methods (not all of them being statistically significant). Keeping the primary table and the secondary table separate, the latter containing the re-scored phrase pairs extracted from the hypotheses, and using both as alternative decoding paths, yields an additional 0.2 BLEU gain (19.8 BLEU to 20.0 BLEU). The advantage is that this architecture gives us an additional means to prefer the primary model over the secondary one, and weight the translation model features in the secondary model differently. For instance, if a source phrase only occurs in the secondary model, the probability estimates will still be unreliable. By giving a higher weight to the smoothed lexical weights, which should suffer less from

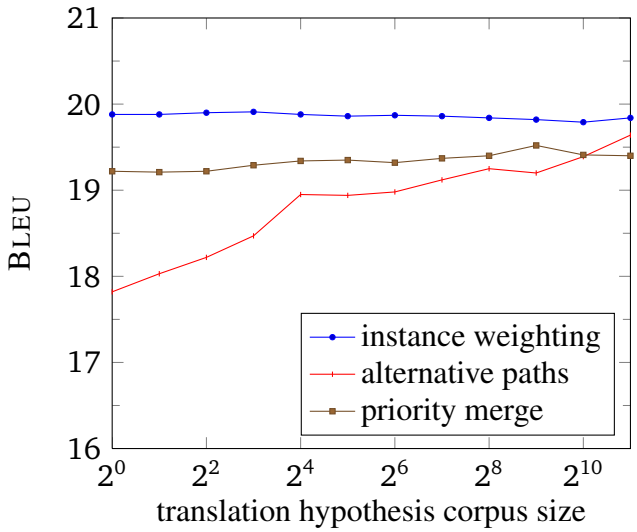

Figure 7.2: BLEU score as a function of translation hypothesis corpus size (DE–FR).

data sparseness, and by modifying the constant phrase penalty in the secondary model, the system can still learn to disprefer these phrase pairs.

The evaluation in table 7.1 has been performed with the translation hypotheses corpus encompassing the full development and test set. If we translate a smaller number of sentences, and train models on these smaller translation hypotheses, the risk of spurious probability estimates increases. We show this trend in figure 7.2, where we vary the number of sentences translated at once. For every block of n sentences, we build a translation hypothesis corpus for these n sentences, with n varying from 1 to 991 (the full size of the test set). Word alignment has been kept constant for these experiments to rule it out as a confounding factor. We see that performance with instance weighting remains stable as the size of the translation hypothesis corpus decreases; the other combination methods experience performance degradation. The biggest degradation is observed for alternative paths decoding, with BLEU decreasing from 19.7 to 17.8. This is presumably because the maximum operation in alternative paths decoding is inherently susceptible to statistical outliers, which can be caused by data sparseness. The systems with priority merge also suffer from performance degradation, although to a lower degree, with a decrease of 0.2 BLEU. This supports our hypothesis that instance weighting is a good choice for multi-engine MT, since it not only performs better, but is also more robust to data sparseness than the other approaches.

Table 7.2 shows an example translation from our test set – we chose the same sentence as in our discussion of mixture modelling (table 6.12, p. 86). The baseline has two main problems: an unknown word (*Vallot-Hütte*), and *schön* not being translated as an intensifier

System	sentence
Source:	Schön brav steigen wir zu Fuss zur Vallot-Hütte.
Reference:	Bien sages, nous descendons à pied jusqu'au refuge Vallot.
	Very well-behaved, we descend on foot to the Vallot cabin.
IN:	Il fait beau, nous montons gentiment à pied à la Vallot-Hütte .
PT 14:	Nous augmentons bien sagement à Fuss à la hutte Vallot.
Google Translate:	Nice bon on arrive à pied à la cabane Vallot.
Instance weighting:	Bien sagement, nous montons à pied à la cabane Vallot.

Table 7.2: Multi-engine MT: example translations.

of *brav*. Personal Translator 14 offers a good translation of *schön brav* to *bien sagement*, but does not translate *Fuss*, since the rule-based system was not configured to handle the Swiss spelling variation of German *Fuß*. Additionally, it offers an unfit translation of *steigen* to *augmentons* (Engl. *augment*). Google Translate translates Vallot-Hütte well to *cabane Vallot*, but offers a poor translation of *schön brav* to *Nice bon*. Finally, the multi-engine system combines the domain-specific translation of *steigen* with the translations from the translation hypotheses for *schön brav* and *Vallot-Hütte*, resulting in the best translation.

We also perform experiments with perplexity minimization to set the instance weights between the primary model and the two hypothesis corpora. This yields worse results; performance is 0.8 BLEU below the approach with uniform weights for pure instance weighting (19.8 BLEU to 19.0 BLEU), and 0.2 BLEU worse for the system that combines instance weighting and alternative path decoding (20.0 BLEU to 19.8 BLEU). The reason is that perplexity minimization gives a higher weight to the hypothesis corpora than to the in-domain corpus, which goes against our intuition and the empirical results which say that the in-domain system is preferable. Note that the performance drop is smaller if we use alternative path decoding, since it compensates for the poorer probability estimates in the secondary model by preferring the primary model more strongly.

7.4.1 On the Use of Perplexity for Machine-Translated Text

We here illustrate the main reason for the failure of perplexity minimization in this setting. Since automatic translations tend to be more literal and monotonic in terms of word order than human ones, we learn less noisy probability distributions from machine-translated text. This has a sizeable effect for the translation of common source words and phrases such as conjunctions, pronouns and punctuation marks. Consider the example segment in figure 7.3, showing the automatic word alignment.

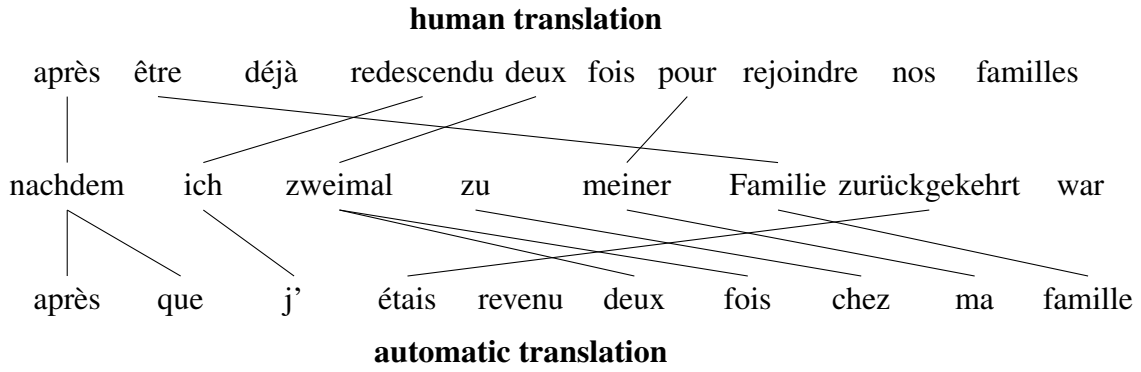

Figure 7.3: Automatic word alignment with human and automatic translation

The non-literal human translation of the sentence causes alignment errors. For instance, there is no correspondence for *ich* in the human translation, and it is misaligned to *redescendu*.[6] In the model learned from the automatic translation (PT 14 for this analysis), the translation is more literal, and *ich* is aligned to *j'*. Looking at the translation probabilities, the model learned from the translation hypotheses learns that $p(\text{je}|\text{ich}) = 0.57$, and $p(\text{j'}|\text{ich}) = 0.23$, while the probabilities are only 0.43 and 0.14 in the model trained on the original parallel text, with some of the probability mass going to phrase pairs such as $p(\text{redescendu}|\text{ich}) = 0.005$.[7] Giving more probability mass to good, frequent translations, and less to noisy phrase pairs, favours the model in perplexity minimization, since minimizing perplexity is equivalent to maximizing the probability of the development set.

As a result of this phenomenon, models learned from machine-translated text may offer a lower perplexity than even in-domain models. This unduly favours them in a perplexity minimization, resulting in a loss in translation performance compared to the unweighted experiment. While non-uniform instance weights may still be able to outperform the systems with uniform instance weighting, this may require discriminative optimization methods. On a more positive note, there is no strong need for non-uniform instance weights in multi-engine MT. As long as the hypothesis corpora are kept small, there is little risk that they disturb the probability estimates of phrase pairs that are well-evidenced in the in-domain model. Instance weighting with uniform weights already satisfies our goal of keeping the impact of the hypothesis models small, so that the hypothesis models only have a significant effect on phrase pairs with little to no evidence in the in-domain model.

[6] There are other alignment errors in the examples, for instance the alignment between *être* and *Familie*, and between *étais* and *zurückgekehrt*.

[7] For better comparability, we trained two models with the same source text and same alignment method, once with the human translation, once with the translation by Personal Translator 14.

System	models	BLEU	METEOR
baseline	IN	17.9	37.0
instance weighting	IN + PT14 + Google	19.8	39.3
instance weighting	IN + OUT + PT14 + Google	20.1	39.3
instance weighting / alt. paths	IN + PT14 + Google	20.0	39.5
instance weighting / alt. paths	IN + OUT + PT14 + Google	20.0	39.3

Table 7.3: Multi-engine MT with OUT data: SMT performance DE–FR.

7.4.2 Combining Out-of-domain Data and Translation Hypotheses

Having shown that both out-of-domain data and external translation hypotheses can help to mitigate the unknown word problem, we here experiment with combining the two sources of knowledge. We use the translation mixture model from chapter 6 that combines IN with 3 out-of-domain corpora (OUT) through instance weighting with perplexity minimization, the corpora being *Europarl*, *OpenSubtitles*, and *JRC-Acquis* (see table 4.1, page 44). This system results in a BLEU score of 18.8, compared to the in-domain baseline of 17.9 (table 6.4, p. 79).

We test two methods of integrating the external translation hypotheses. Firstly, we use instance weighting to add the two hypothesis models to the mixture model. We simply extend the weight vector to include them with weight 1, which corresponds to our uniform weights experiments in table 7.1.[8]

For a second system, we use alternative path decoding, as in the experiments in table 7.1, but with the mixture model as primary model instead of IN. The secondary model is the same as in the experiment without OUT data, namely a model obtained from uniform instance weighting of IN and the two hypothesis models, restricted to the phrase pairs that occur in the hypothesis corpora. The motivation for not using OUT in the secondary model is that we want to take into account the evidence from the in-domain model when scoring the hypothesis models, but not necessarily evidence from out-of-domain models.

The results are shown in table 7.3. There are no consistent gains from including out-of-domain parallel corpora, compared to only using IN and the translation hypothesis corpora. Our interpretation as to why the improvement from the two experiments – 0.9 BLEU from adding OUT, 2.1 BLEU from external translation hypotheses – is not additive is the following. Both our inclusion of out-of-domain parallel data and external translation hypotheses had the same motivation, namely to fill in any lexical gaps in the IN model, thus mitigating

[8]The instance weights are normalized so that IN has a weight of 1. The model vector is thus the following: (IN, Europarl, OpenSubtitles, JRC-Acquis, Google, Personal Translator), and the corresponding weight vector for $p(\bar{t}|\bar{s})$ is: (1, 0.17, 0.16, 0.03, 1, 1).

the unknown word problem. This is also illustrated in the example translations in tables 6.12 (p. 86) and 7.2 (p. 109). Both show the translation of the same sentence, and how both OUT data (table 6.12) or external MT systems (table 7.2) fill the same lexical gaps.

In our experiments, the external MT hypotheses fulfill this role better than OUT data; while adding the OUT parallel corpora reduces the OOV rate (in tokens) in the test set from 8.2% to 5.8%, the external MT hypotheses achieve a reduction to 1.3%. The combination of the two resources, yields a modest additional reduction, down to 0.9%. On the basis of OOV rate alone, it is conceivable that adding OUT data is almost redundant. The only caveat is that the OOV rate does not tell us anything about the quality of the phrase pairs. After all, the hypothesis models are trained on the actual source text that we want to translate, so it is no surprise that they cover almost all of the vocabulary, with the possible exception of unaligned words. Thus, in principle, it may still be worthwhile to combine OUT data and external MT hypotheses, especially if the former yield better translations than the latter.

In conclusion, a combination of in-domain and out-of-domain models and external MT hypotheses did not result in better performance than a combination of an in-domain model and external hypotheses alone, at least in our experimental setting. One advantage of including all available component models is that the system creator or user is not required to select the in-domain model himself, but can automatically weight the contribution of the models. This also means that this multi-engine MT architecture is compatible with a multi-domain setting as described in (Sennrich et al., 2013), where the weight vector is dynamically set to support multiple domains.

7.5 Summary

Translation hypotheses by other MT systems, for instance general-domain SMT or rule-based systems, can be used as an additional source of knowledge to tackle the unknown word problem. We demonstrate that the inclusion of MT hypotheses can be performed through the same architecture that we present in section 6.6, and perform an evaluation that shows significant improvements of up to 2 BLEU points over the pure IN system. The external translation hypotheses fulfill the same role as out-of-domain parallel data, namely that of reducing the number of unknown words; adding both out-of-domain parallel data and external MT hypotheses to the IN system did not yield better results than just adding the latter.

We have discussed data sparseness issues related to the inclusion of translation hypotheses through a separate translation model. Specifically, performance degrades as the size of the

translation hypothesis corpus becomes smaller. We have shown that a combination through instance weighting is both performant and robust to data sparseness.

The discussion of integrating external MT systems has been an example of the wider topic of integrating heterogeneous knowledge sources into an SMT system. We envision that the same architecture can be used to integrate information from dictionaries or other static resources into an SMT system, and that it can also serve to quickly integrate user feedback, for instance in the form of post-edited translations of previous sentences in a text (Hardt and Elming, 2010).

8 Multiword Expressions and Flexibility Features

When we discussed the ambiguity problem so far, we solely focused on how it is affected by the training and test domains, and how we can apply domain adaptation methods to alleviate it.[1] We did not discuss the fact that existing SMT models are able to resolve some ambiguities through contextual information. Specifically, phrase-based SMT systems use contextual information both by its use of larger phrases as translation units, rather than making an independence assumption between word translations, and through n-gram language models. If words occur in new contexts, however, the disambiguation capability of SMT systems is limited.

Some translations that are specific to a domain or text type are highly idiomatic. The English word *floor*, for instance, typically means *Boden* (referring to the ground) or *Stockwerk* (level of a building) in German. In formal debate settings, however, *floor* is often translated as *Wort* (English: *word*) – this is the predominant translation in Europarl. This is restricted to a few expressions, such as *give/have/request/take/yield the floor*. Some variation is possible, e.g. by inflecting the verb or adding a direct object (example 1), but other variants such as examples 2 and 3, while valid utterances, do not qualify as instances of this idiom, and would indicate a non-idiomatic meaning of *floor*.

1. *I gave him the floor*
2. *I gave him a floor*
3. *We take the third floor*

The problem is that, by learning that *Wort* is a frequent translation of *floor*, this translation is likely to be overgeneralised to examples 2 and 3, especially if those contexts have not been seen during training. We introduce a measure of a phrase pair's flexibility, which we think is a better indicator of how generalisable a phrase pair is than its frequency. The approach is applicable, but not limited to domain-specific systems.

[1] This chapter is based on (Sennrich, 2013).

8.1 Introduction

A defining property of multiword expressions (MWEs) is that they are idiosyncratic (Sag et al., 2002). For Statistical Machine Translation (SMT), MWEs whose meaning is non-compositional, i.e. which cannot be translated word by word, cause two major problems. The obvious problem is that MWEs may be translated incorrectly if we translate them word by word. A second problem, which has received less attention in SMT research, is that translations that we learn from the components of a MWE can rarely be generalised to other contexts. If a word frequently occurs in a MWE with an idiosyncratic translation, learning this idiosyncratic translation on the word level pollutes the translation model.

Consider for instance the English phrase *of course*, which is translated into French as *bien sûr*. SMT systems, which typically perform unsupervised word alignment to learn translation correspondences, not only learn the translation pair (*of course*, *bien sûr*), but also (*of*, *bien*) and (*course*, *sûr*). Especially if the fixed expression *of course* is more frequent during training than other translations of *course*, the translation pair (*course*, *sûr*) is misapplied to occurrences of *course* in new contexts. This problem affects various linguistic phenomena that fall under the umbrella term MWE: complex prepositions, idioms, compounds and named entities, among others.

We describe an algorithm to measure a phrase pair's flexibility, i.e. whether it occurs in many contexts or is restricted to fixed expressions. Note that the aim is not to penalize MWEs themselves, which may be flexible in terms of their contexts, but only phrases that are part of a larger MWE. The algorithm is related to Kneser-Ney smoothing, but different from previous work on translation model smoothing such as (Foster et al., 2006). In contrast to other related work on MWEs in SMT, our approach is unsupervised and language-independent.

8.2 Related Work

The fact that word-based translation techniques are inadequate to deal with MWEs, which are by definition non-compositional, has been recognized early (Smadja, 1993), and has led to approaches that extract MWEs in order to improve bilingual resources (Smadja et al., 1996), a technique that continues to be a research topic to this day (e.g. Carpuat and Diab, 2010). Using contextual information to disambiguate translations is an equally well-researched topic (Carpuat and Wu, 2007; Chiang et al., 2009). One can even argue that the success of phrase-based SMT (Koehn et al., 2003) compared to word-based approaches is in large part due to the existence of MWEs in natural language.

Our work is not concerned with improving the translation of MWEs themselves, but with preventing an overgeneralisation of translations learned from MWEs. In other words, our aim is not to improve the translation of words in known contexts, but picking a better translation if a word occurs in a new context. It shares the aim with the work by Lambert and Banchs (2006), who convert MWEs into single tokens in a preprocessing step, thus preventing MWE sub-segments from being extracted. In order to identify MWEs in the source text, they exploit asymmetries in word alignment, lemmatisation and PoS-tagging. They found that the positive effect of suppressing wrong phrase pairs in some instances was counterbalanced by increased data sparseness, especially because some word sequences were erroneously identified as MWEs. Pal et al. (2011) follow the same idea for the language pair English–Bengali, with different MWE extraction techniques.

The approach we propose is technically related to Kneser-Ney smoothing (Kneser and Ney, 1995). In translation modelling, the main aim of smoothing is to counteract unreliable probability estimations by falling back to less sparse data. For translation model smoothing, Foster et al. (2006) give an overview over different techniques, and propose an adaptation of Kneser-Ney smoothing to translation models.

$$p_b(\bar{s}|\bar{t}) = \frac{N_{1+}(\bar{s}, \bullet)}{N_{1+}(\bullet, \bullet)} \tag{1}$$

Equation 1 uses Kneser-Ney counts to reward phrase pairs whose source phrase co-occurs with many different target phrases. We follow (Kneser and Ney, 1995) in using N_{1+} to denote the number of types, and $\bullet$ as a wildcard. $N_{1+}(\bar{s}, \bullet)$ thus stands for the number of different target phrases that a source phrase $\bar{s}$ co-occurs with, and $N_{1+}(\bullet, \bullet)$ for the total number of phrase pairs in the model.

In (Foster et al., 2006), equation 1 is interpolated with the original probability distribution. The smoothed distribution is useful to penalize phrase pairs whose source phrase is rare, and which we expect to be unreliable due to data sparseness. However, it is not related, either in the motivation or in the effect, to the problem that we aim to solve.

8.3 Learning Translations in SMT

To illustrate why wrong translations are learned from MWEs, let us consider the common SMT training process. In phrase-based SMT and hierarchical phrase-based SMT, translations are extracted from a word-aligned corpus. This extraction is performed through heuristics that extract phrase pairs which are consistent with word alignment, specifically,

so that no word in the source phrase is aligned to a word outside the target phrase, and vice versa. For MWEs, this means that phrase pair extraction for the whole MWE, and its subphrases and words, are co-dependent. A MWE is only extracted if its components do not violate word alignment, and when the latter is the case, this also entails that these components will form phrase pairs of their own. In other words, the phrase table is learned with a compositionality assumption, and the model has no means to learn that a phrase pair can be correct while its components should not be used independently.

While state-of-the-art SMT systems have this technical weakness, they are easy to extend thanks to their log-linear framework. In the final translation model, each extracted phrase pair $(\bar{s}, \bar{t})$ has multiple scores, which are combined with each other and other features such as the language model probability in a log-linear model. Most common are phrase translation probabilities estimated through (smoothed) relative frequencies $p(\bar{s}|\bar{t})$ and $p(\bar{t}|\bar{s})$, and a smoothed probability distribution based on word translation probabilities (Koehn et al., 2003). We extend this log-linear model through new features that measure a phrase pair's flexibility.

8.4 Flexibility Features

We will build on the motivation that Chen and Goodman (1998) give for Kneser-Ney smoothing, different from the one in (Kneser and Ney, 1995), and reproduced here in a slightly abridged version:

> [...] consider building a bigram model on data where there exists a word that is very common, say FRANCISCO, that occurs only after a single word, say SAN. Since c(FRANCISCO) is high, the unigram probability p(FRANCISCO) will be high [...] However, intuitively this probability should not be high since in the training data the word Francisco follows only a single history. That is, perhaps the word FRANCISCO should receive a low unigram probability because the only time the word occurs is when the last word is SAN, in which case the bigram probability models its probability well. (Chen and Goodman, 1998, 15)

We can apply the same intuition to translation modelling. Phrase pairs which only occur in specific contexts are more likely to be idiomatic than phrase pairs that have been observed in many contexts. In the *San Francisco* example, we do not need to assign high probability to the unigrams because the sequence is modelled well by the bigram model. Similarly, the phrase table contains phrases of various sizes. Going back to our introductory example, we can assign a low probability to the translation pair (*course*, *sûr*) with impunity because the

correct translation of the MWE is expressed in the model through the larger phrase pair (*of course*, *bien sûr*).

How do we formalize this intuition? The Kneser-Ney distribution is not based on relative frequencies, but the relative number of types. However, in the previous section, we have seen that Kneser-Ney smoothing for phrase tables as introduced by Foster et al. (2006) is not based on the context of a phrase, but the number of target phrases a source phrase occurs with.

We introduce new probability distributions that are not based on relative frequency estimates, but on the number of different contexts in which a phrase pair occurs. Intuitively, we use them to predict how likely a phrase pair is to occur in a new context. We will call a phrase pair flexible if it occurs in many contexts, as opposed to inflexible phrase pairs that we only observe in few contexts. Note that under this definition, even fixed expressions may be considered flexible if they themselves occur in many contexts. It is not the translation of MWEs that we aim to penalize, but the translation of their individual segments.

In order to measure a phrase pair's flexibility, we introduce equation 2. Given a source phrase $\bar{s}$ and a target phrase $\bar{t}$, with $\bar{s}$ being a sequence of words from s_i to s_j, we consider triplets of the form $(s_x, \bar{s}, \bar{t})$ for the flexibility measure. Different positions can be considered for s_x. We introduce two new probability distributions; the first, with $x = i - 1$, is based on the number of contexts to the left of $\bar{s}$, and will be referred to as $p_{\text{flex\_left}}$. The second, $p_{\text{flex\_right}}$, is based on the number of right contexts, with $x = j + 1$.[2]

$$\begin{aligned} p_{\text{flex\_\{left,right\}}}(\bar{t}|\bar{s}) &= \frac{N_{1+}(\bullet, \bar{s}, \bar{t})}{\sum_{\bar{t}'} N_{1+}(\bullet, \bar{s}, \bar{t}')} \\ &= \frac{N_{1+}(\bullet, \bar{s}, \bar{t})}{N_{1+}(\bullet, \bar{s}, \bullet)} \end{aligned} \tag{2}$$

N_{1+} denotes the number of types, and $\bullet$ are wildcards. $N_{1+}(\bullet, \bar{s}, \bar{t})$ is thus the number of different triplets $(s_x, \bar{s}, \bar{t})$ observed during training. $p_{\text{flex\_left}}(\bar{s}|\bar{t})$ and $p_{\text{flex\_right}}(\bar{s}|\bar{t})$ are calculated analogously, i.e. by considering the number of contexts to the left and right of the target phrase. We can theoretically increase the window that we consider to be the context of a phrase, but as we increase the window size, the number of different types increases, and the new probability estimates tend towards the baseline relative frequency estimate.

Table 8.1 shows the effect of this calculation for selected phrase pairs.[3] The first example illustrates how the English complex preposition *in line with* is affected. In German, it is

[2] We add a special token for s_{i-1} if the phrase begins at the start of a sentence, and do the same for s_{j+1} at the end.

[3] The examples are from the models described in section 8.6.1.

$\bar{s}$	$\bar{t}$	$p_{\text{RF}}(\bar{t}\|\bar{s})$	$p_{\text{flex\_left}}(\bar{t}\|\bar{s})$	$p_{\text{flex\_right}}(\bar{t}\|\bar{s})$
line	Einklang	0.159	0.003	0.005
line	Linie	0.146	0.072	0.061
in line with	im Einklang mit	0.169	0.073	0.059
course	sûr	0.444	0.001	0.066
course	cours	0.079	0.023	0.010

Table 8.1: Translation model probabilities for selected phrase pairs with different probability estimation methods: relative frequency (p_{RF}); flexibility distribution ($p_{\text{flex\_left}}$ and $p_{\text{flex\_right}}$).

typically translated to *im Einklang mit*. However, if *line* occurs in other contexts, a typical translation would be *Linie* or *Reihe*, but almost never *Einklang*. The latter translation is restricted to *in line with*. We see that our model risks to translate *line* as *Einklang* because the relative-frequency estimate for (*Einklang*|*line*) is higher than that of (*Linie*|*line*). However, since the phrase pair (*line*, *Einklang*) occurs in very few contexts, both on the source and target side, its flexibility estimates are much lower than the estimate obtained from relative frequencies. (*Linie*|*line*) and (*im Einklang mit*|*in line with*) both occur in various contexts, and their flexibility estimates remain relatively high. The latter is an important point of our technique: the translation of MWEs as a single unit, which is a desirable property of phrase-based models, is not penalized.

The second set of phrase pairs is based on our introductory example (*of course*, *bien sûr*), and demonstrate why we measure flexibility to the right and to the left independently. The phrase pair p(*course*|*sûr*) is only inflexible to the left of *course*, but should still be penalized.

If a phrase pair is frequent, but only occurs in few contexts, this indicates that it is part of a larger MWE, and can safely be dispreferred. The cases in which an inflexible phrase pair is in fact a good translation should usually be handled by larger translation units, i.e. the MWE as a whole. However, there are exceptions to this rule. An exception are phrase pairs that typically occur at the beginning or end of a sentence. These may be inflexible according to our model, even if they are not part of a larger translation unit.

The flexibility measure has the same aim as the joining of MWEs that Lambert and Banchs (2006) describe, namely to prevent overgeneralisation of phrase pairs learned from MWEs to new contexts. It has the advantage of being language-independent and requiring no additional resources. Furthermore, it does not need to make a hard classification into MWEs and others, and thus does not suffer from an increase in data sparseness.

8.4.1 Variants for Hierarchical Phrase-based Models

We can extend the notion of a phrase pair's flexibility to hierarchical phrase-based models (Chiang, 2005). However, we argue that a naive transfer of the approach to hierarchical phrase-based systems is incomplete. Because subphrases are allowed in hierarchical rules, there are additional ways in which a rule can be inflexible. Consider these three rules that might be learned from occurrences of the phrase pair (*of course*, *bien sûr*).

1. $X \rightarrow \langle\ course\ ,\ sûr\ \rangle$
2. $X \rightarrow \langle\ X_1\ course\ ,\ X_1\ sûr\ \rangle$
3. $X \rightarrow \langle\ and\ X_1\ course\ ,\ et\ X_1\ sûr\ \rangle$

Each of these examples is a poor generalisation, and should ideally be penalized in the model. In all cases, we only expect the translation of *course* into *sûr* if *course* is preceded by *of*. In phrase-based models, this can be expressed through the relative number of left contexts observed with the source phrase, or $p_{\text{flex\_left}}(\overline{t}|\overline{s})$. This works for the hierarchical rule 1, but not for 2 and 3.[4] The reason is that *of* is not to the left of the rules extracted from the corpus, but part of the subphrase X_1. This leads to the question how we can formulate an alternative notion of context, so that the inflexibility of rules 2 and 3 can be learned.

We denote **FLEX_H1** the approach with $p_{\text{flex\_left}}$ and $p_{\text{flex\_right}}$ that has been described in the previous section, and present multiple alternatives:

FLEX_H2

The first variant, called **FLEX_H2** henceforth, redefines which word is considered the left and right context in a hierarchical rule. If a hierarchical rule starts with a subphrase, the rightmost word of this subphrase is considered the rule's left context (instead of the word to the left of the subphrase). In other words, this variant does not use $x = i - 1$ for $p_{\text{flex\_left}}$, but $x = i - 1 + n$, with n being the length of the subphrase that the rule starts with, or $n = 0$ if the rule does not start with a subphrase. If it ends with a subphrase, the leftmost word of this subphrase is considered for $p_{\text{flex\_right}}$, or $x = j + 1 - n$, with n being the length of the subphrase that the rule ends with, or $n = 0$ if the rule ends with a terminal symbol. This new definition aims to capture the inflexibility of rule 2.

[4] In our training corpus from section 8.6.1, $p_{\text{flex\_left}}(\overline{t}|\overline{s})$ is $\frac{2}{2688}$ for rule 1, $\frac{480}{3506}$ for rule 2, and $\frac{232}{722}$ for rule 3. In other words, $p_{\text{flex\_left}}(\overline{t}|\overline{s})$ successfully assigns a low probability to the inflexible rule 1, but too high a probability to rules 2 and 3.

FLEX_H3

A second possibility is to not only consider the words to the left and right of a rule to be its context, but also its subphrase(s). We start with **FLEX_H2**, and add a new feature $p_{\text{flex\_sub}}(\overline{t}|\overline{s})$ that is based on the number of different types of subphrases a hierarchical rule occurs with. This new feature is implemented through equation 2 by redefining s_x. Instead of the word to the left or the right of a rule, let s_x be defined as the full subphrase, or, if a rule contains multiple subphrases, the concatenation of all subphrases.[5] If a rule does not have a subphrase, we let $p_{\text{flex\_sub}}(\overline{t}|\overline{s})$ be 1, so that this feature is without cost for rules without subphrases. We also add $p_{\text{flex\_sub}}(\overline{s}|\overline{t})$, which is defined analogously.

8.5 Filtering Hierarchical Rule Tables

We point out again that the flexibility features do not try to combat data sparseness, in contrast to the variant of Kneser Ney smoothing described by (Foster et al., 2006). In preliminary experiments, we found that some of the differences between the baseline system and the experimental ones were due to spurious phrase or rule pairs whose probability estimates were unduly high.

Thus, we use significance test filtering (Johnson et al., 2007) for phrase tables, which, as the authors note, has a similar effect as smoothing, since both pruning and smoothing penalizes infrequent phrase pairs. We extend their approach to hierarchical rule tables. Since (Johnson et al., 2007) do not base the significance test on alignment counts, but co-occurrence counts in the parallel corpus, we decided on an approximative method to count the number of occurrences of hierarchical rules, which can be implemented with a suffix array. Three frequencies are required to perform a statistical significance test for a phrase pair or rule: c_s, the frequency of the source phrase/rule, c_t, the target phrase/rule frequency, and c_{st}, the co-occurrence frequency of the source and target phrase/rule.

For hierarchical rules without subphrases, or rules which consist of a single, uninterrupted terminal sequence with subphrases at the beginning and/or end of the rule, we can use the same procedure as for phrase-based systems, namely extracting a set of sentences in which the source terminal sequence occurs, doing the same for the target sequence, and intersecting the two sets to obtain c_{st}.

For hierarchical rules which consist of multiple terminal sequences, interrupted by subphrases, we approximate the occurrences of the rule by extracting the set of occurrences for

[5] We insert a delimiter between two subphrases to distinguish between X_1 = 'a b', X_2 = 'c' and X_1 = 'a', X_2 = 'b c'.

system	DE–EN		EN–DE		FR–EN		EN–FR	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
newstest2011								
baseline	21.0	28.6	15.6	36.2	28.6	33.8	30.3	51.2
FLEX	**21.2**	**28.7**	**15.8**	**36.4**	**28.8**	33.8	30.3	51.1
cross-domain								
baseline	29.1	28.9	27.0	42.0	25.5	32.2	22.2	44.1
FLEX	**29.4**	**29.1**	27.1	**42.2**	**26.4**	**32.6**	**22.6**	**44.5**

Table 8.2: SMT results on newstest2011 and cross-domain test sets. Phrase-based models.

each terminal sequence, and using the intersection of these sets as occurrences of the full rule.

For a rule $X \rightarrow \langle\ a\ b\ X_1\ c\ ,\ x\ X_1\ y\ z\ \rangle$, c_s is thus the number of source sentences in which $a\ b$ and c occur, c_t the number of target sentences in which x and $y\ z$ occur, and c_{st} the number of sentence pairs in which $a\ b$ and c occur in the source sentence, x and $ys\ z$ in the target sentence.

8.6 Evaluation of Flexibility Scores

8.6.1 Data and Methods

We perform the evaluation on the language pairs French–English and German–English, with training data mostly from the shared task of the 2011 Workshop on Machine Translation (Callison-Burch et al., 2011). For both language pairs, we use *Europarl* and *News-Commentary* as parallel data sets.[6] Language models are trained on the respective target language sides of *Europarl*, *News-Commentary*, and the monolingual *News* data set, interpolated for minimal perplexity on news-test2008. For FR–EN, we additionally use the 10^9 corpus and the United Nations corpus as parallel data sets (*UN-Giga* in table 4.1). As additional DE–EN data, we use *JRC-Acquis* and *OpenSubtitles*. The total amount of parallel training data is about 900 million words for FR–EN, and 110 million words for DE–EN.

We evaluate each system with two test sets. The first test set is newstest2011 from WMT 2011, with the system optimized on news-test2008; as second test set, we use patent abstracts for FR–EN[7], and help desk tickets provided to us by the software company Finnova for DE–EN. The reason for this is that we expect idiomaticity to be more of a problem if training and test set are dissimilar, since MWEs may be domain-specific (Smadja et al., 1996). We will refer to the second test set as *cross-domain* setting.

[6] For the size of the data sets, see table 4.1, p. 44.

[7] extracted from the COPPA corpus (Pouliquen and Mazenc, 2011); IPC section A: human necessities.

system	sentence
source	Le jeu comprend des cartes objectifs et de l'argent pour le **jeu**.
reference	The game apparatus includes target cards, and **game** money.
baseline	The game includes maps objectives and money for the **stake**.
+FLEX	The game includes maps objectives and money for the **game**.
source	[...] les cylindres tournant librement et servant d'organe de **support** [...]
reference	[...] the freely rotating rollers, which act as **support** members [...]
baseline	[...] freely rotating cylinders and **media** body [...]
+FLEX	[...] freely rotating cylinders and **support** body [...]
source	Improvements relating to **board games**
reference	Améliorations apportées à des **jeux de société**
baseline	Des améliorations relatives aux **jeux du conseil**
+FLEX	Des améliorations concernant les **jeux de société**

Table 8.3: Example translations from patents corpus. Phrase-based models.

8.6.2 Phrase-based Results

Table 8.2 shows our experimental results with phrase-based systems on newstest2011, and the two cross-domain test sets. The only change of our FLEX system over the baseline is the addition of four flexibility features to the log-linear model, namely $p_{\text{flex\_left}}$ and $p_{\text{flex\_right}}$ in both translation directions.

On newstest2011, we observe an improvement of 0.2 BLEU in three of the four translation directions. On the help desk and patent test sets, the flexibility features lead to larger improvements of up to 0.9 BLEU (FR–EN), with 0.3–0.4 points of improvement observed for DE–EN and EN–FR, and no significant improvement for the language pair EN–DE.

There are a number of possible explanations as to why we observe a gain in performance with some test sets, but not with others. Defining the context as the immediate neighbours of a phrase pair does have limitations. In the case of DE–EN, for instance, we note that the relatively free word order in German makes it harder to recognize if a word is part of a MWE with our approach. An example are German verb particles, which typically do not occur immediately after the verb, but at the end of the matrix clause. Without preordering, we cannot reliably distinguish between *schlägt* (engl: *beats*) and *schlägt ... vor* (engl: *proposes*).

Apart from the translation direction, the extent to which (parts of) MWEs are misused during translation depends on the training and test domain, since MWEs may be domain-specific (Smadja et al., 1996). If training and test domain are similar, using an idiomatic translation learned from the domain is more likely to be right than if the test set is from a

different domain. Conversely, we expect cross-domain translation to benefit more strongly from the flexibility features, and consider this the main reason why we observe a larger performance boost with cross-domain test sets. Given that we observe the largest performance gains in cross-domain translation, we argue that the flexibility features may be especially helpful for general purpose SMT systems, and/or systems that use training data from various different domains.

A further possible explanation for the varying results is that the extent of the problem depends on how literally and in which direction the training text was originally translated. Newmark (1991) remarks that one feature of "translationese" is that idioms are sometimes translated literally, and Lembersky et al. (2012) find that that the original language of the training data affects SMT performance.

Furthermore, adding flexibility features has side effects which may be both positive and negative. Specifically, the systems with flexibility features tend to give a higher weight to the phrase penalty in the log-linear model, meaning that the systems use fewer, but larger translation units during decoding. Such a preference of large translation units makes sense if we want to correctly translate MWEs despite the flexibility features: note our motivating examples in table 8.1, and that the flexibility features penalize (*Einklang*|*line*), but not (*im Einklang mit*|*in line with*).

Table 8.3 shows a few examples where the baseline system misapplies an inflexible phrase pair. In these examples, the translations are so wrong that it might be hard to intuitively understand why they were even learned in the model. Hence, we list the relevant phrase pairs, all frequent in the training set, that introduce these word translations into our model:

- *en* ***jeu*** *– at* ***stake***
- *format de* ***support*** *–* ***media*** *format*
- ***board*** *of directors –* ***conseil*** *de direction*

In all examples, the flexibility features successfully penalize the (misused) idiomatic translation. However, the third example nicely illustrates that the experimental system does not prevent the translation of multiword expressions when they are encountered as a whole. The experimental system penalizes the inflexible translation pair (*conseil*|*board*), but not (*jeux de société*|*board games*), which is chosen instead. The examples also illustrate our point about MWEs being domain-specific: *board of directors* only occurs once in a patent corpus of 9 million sentences, but 12 500 times in the 10^9 corpus (21 million sentences). *format de support* occurs 16 times in the patent corpus, and 3000 times in the 10^9 corpus.

system	DE–EN		EN–DE		FR–EN		EN–FR	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
newstest2011								
unfiltered	21.1	29.1	15.5	36.4	29.0	34.0	30.2	51.0
filtered	21.5	29.1	15.6	36.2	29.2	34.1	30.4	51.0
FLEX_H1	21.5	29.1	**15.8**	**36.5**	**29.6**	34.2	30.5	51.1
FLEX_H2	21.5	29.1	**15.9**	**36.5**	**29.5**	34.2	**30.6**	**51.2**
FLEX_H3	21.5	29.1	**15.8**	**36.5**	**29.7**	**34.3**	30.5	51.1
cross-domain								
filtered	29.2	29.1	26.3	41.6	24.2	31.9	22.8	44.7
FLEX_H1	29.3	29.2	**27.1**	**42.5**	**24.9**	**32.1**	22.7	44.7
FLEX_H2	29.3	29.2	**27.1**	**42.4**	**25.0**	**32.2**	22.8	44.7
FLEX_H3	29.3	29.2	**27.2**	**42.5**	**24.7**	32.0	22.8	44.8

Table 8.4: SMT results on newstest2011 and cross-domain test sets. Hierarchical models. Highlighted systems are significantly better than (filtered) baseline.

8.6.3 Hierarchical Results

Table 8.4 shows translation results for hierarchical systems. Regarding statistical significance filtering, we see an increase in BLEU by up to 0.4 points for the filtered models, along with a reduction in rule table size. METEOR remains constant, or, for EN–DE, drops slightly by 0.2 points. A closer look at the METEOR statistics gives us an explanation for this discrepancy between BLEU and METEOR. Unigram precision benefits from significance filtering, while unigram recall, which is only considered by METEOR, is slightly decreased. We conduct all future experiments with filtered tables.

Just as with the phrase-based systems, the impact of the flexibility scores varies between the different translation directions and test sets. The biggest effect is observed for the translation directions EN–DE, with 0.2-0.3 BLEU points gained on newstest2011, and 0.8–0.9 BLEU points on the cross-domain test set, and FR–EN, with 0.3-0.5 BLEU points gained on newstest2011, and 0.5-0.8 BLEU points on the cross-domain test set. For DE–EN and EN–FR, the differences are small or non-existent.

All variants of hierarchical flexibility scores perform similarly well, with no consistent winner variant. For DE–EN and EN–FR, adding flexibility scores yields no significant improvement.

A comparison between phrase-based and hierarchical systems gives a mixed picture. For the language pair FR–EN, the hierarchical system is better on newstest2011, whereas the phrase-based one performs better on the patent test set. An analysis of the METEOR statistics suggests that the main difference between the phrase-based and the hierarchical models is in METEOR's fragmentation penalty, which means that reordering phenomena are at the

root of these differences. Adding flexibility features is effective for both types of models, as it primarily affects the precision and recall scores. This indicates that the flexibility features improve the accuracy of the translation.

8.7 Summary

We describe a simple, yet effective way to measure the flexibility of phrase pairs, and show that these flexibility measures improve translation quality. By penalizing inflexible phrase pairs, i.e. phrase pairs that only occur in the context of larger multiword expressions, we measured gains in translation quality of up to 0.9 BLEU and METEOR points. The flexibility of phrase pairs is learned from the parallel training text, and expressed through new features in the log-linear SMT model. This makes the approach simple to implement and language-independent.

We apply the approach to both phrase-based and hierarchical phrase-based SMT models, and discuss variants for hierarchical models. We also demonstrate that rule table filtering based on statistical significance tests is possible and fruitful.

The flexibility scores that we propose still have their limitations. Specifically, there are MWEs which have a higher flexibility on the surface level, but which we still would like to mark as inflexible. An example are German verb particles, which do not necessarily occur immediately after the verb, but at the end of the matrix clause, and whose translation is often non-compositional. Coupling flexibility scores with reordering might help to overcome these limitations.

An alternative application for the flexibility scores could be linguistic exploration, i.e. extracting multiword expressions in a parallel corpus, similar in spirit to (Caseli et al., 2009), who use automatic word alignment in a parallel corpus to identify multiword expressions.

9 Conclusion and Outlook

This thesis presented original work in domain adaptation for translation models in Statistical Machine Translation. We introduced a novel architecture for translation model combination, which applies instance weights at decoding time to a vector of component models, each model storing the sufficient statistics (i.e. the word and phrase (pair) frequencies) necessary for the computation of translation probabilities.

We have shown that this architecture is effective at improving performance in various settings. In contrast to related work in domain adaptation, multiple domains can be supported in a single system, and a running system can be quickly adapted to new domains, meaning that domain adaptation can be performed at little cost. We have also shown that the architecture is flexible in that it allows one to integrate different types of knowledge sources into SMT systems. While we demonstrated the inclusion of out-of-domain parallel data for training, we observed even bigger gains by using the architecture to integrate the knowledge of external MT systems, which act as a source of general-domain data, into the model.

We also discussed the risk of domain adaptation, and found that our architecture continues to perform well as we scale up the number of component models, or if we apply it in a setting with several equally fit component models. Some related approaches that we evaluated failed to scale well to a high number of component models, or even fell below the performance of an unadapted system in those suboptimal settings. We also performed domain adaptation in an unlabelled setting, and demonstrated that we can even perform domain adaptation with a single, heterogeneous parallel training corpus, using unsupervised clustering of training data.

The most effective strategy to improve SMT performance on a low-resource domain is to obtain more in-domain training data. We discussed learning curves with in-domain and out-of-domain data, and introduced a novel algorithm for sentence alignment which can help to extract more useful training data from a parallel text than other algorithms. With a collection of digitised journal articles, our algorithm improved recall by more than 50% compared to other sentence alignment algorithms.

We also discussed (domain-specific) multiword expressions as a source of ambiguity, and why the compositionality assumption of state-of-the-art models results in idiomatic translations being misapplied outside of the respective idioms. We introduced new features

to the log-linear translation model which measure how flexible a phrase pair is, i.e. the number of contexts it has been observed in during training. We have shown that these features can reduce the overgeneralisation of idiomatic phrase pairs to new contexts.

We released the main results of our work as free software[1]. While we believe that our techniques are ready for deployment in research and industry, this thesis also opens new research possibilities, both in terms of research building on our work, and research addressing open questions.

- While we have shown that perplexity minimization is a quick and reasonable optimization method for instance weighting, using one of the usual MT metrics such as BLEU as objective function would be an attractive goal. The main obstacle is that such an optimization is computationally more expensive, and early attempts to optimize instance weights on BLEU suffered from having to create a new translation model for each set of weights (Shah et al., 2010). The translation model architecture that we presented, which is able to change instance weights at runtime without having to retrain the translation model, could facilitate the implementation of new optimization methods.

- We expect diverse new translation model features to become established as a result of the new discriminative optimizers such as PRO and MIRA, which promise to better scale to a high number of features. While instance weighting and perplexity minimization is an effective method to adapt translation model probabilities, many features will require different adaptation techniques. Our flexibility features fall under this category, since they estimate phrase pairs to be improbable if they only occur in few contexts, even if they are frequent. Perplexity may thus not be a suitable objective function for these features.

- This thesis mainly operated with phrase-based translation models. While the transfer of the techniques to other model types is possible in principle, the techniques may need to be adapted to the characteristics of the model. As an example, we refer to our adaptation of flexibility features to hierarchical models in section 8.4.1 (p. 121).

- We envision that our translation model architecture can be used for other purposes than the ones demonstrated in this thesis, for instance to quickly integrate – and appropriately weight – user feedback in interactive Machine Translation (Ortiz-Martínez

[1] as contributions to Moses (`https://github.com/moses-smt/mosesdecoder`) or as separate projects on `https://github.com/rsennrich`.

et al., 2010). This could lead to personalized SMT systems that share one or multiple background models, but are adapted to the user through a (small) user-specific translation model and user-specific model weights. Another application and area of further research is multi-domain machine translation. We already performed pilot studies in a multi-domain setting (Sennrich et al., 2013), demonstrating that a system is possible that requires no domain labels or user selection of the target domain, but uses unsupervised clustering to adapt to multiple target domains.

Bibliography

Sadaf Abdul-Rauf, Mark Fishel, Patrik Lambert, Sandra Noubours, and Rico Sennrich. 2012. Extrinsic evaluation of sentence alignment systems. In *Workshop on Creating Cross-language Resources for Disconnected Languages and Styles*, pages 6–10.

Sisay Fissaha Adafre and Maarten de Rijke. 2006. Finding similar sentences across multiple languages in Wikipedia. *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 62–69.

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical machine translation. Technical report, Final Report, JHU Summer Workshop.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*.

Pratyush Banerjee, Jinhua Du, Baoli Li, Sudip Kumar Naskar, Andy Way, and Josef Van Genabith. 2010. Combining multi-domain statistical machine translation models using automatic classifiers. In *9th Conference of the Association for Machine Translation in the Americas (AMTA 2010)*. Denver, Colorado, USA.

Pratyush Banerjee, Sudip Kumar Naskar, Johann Roturier, Andy Way, and Josef van Genabith. 2012. Domain Adaptation in SMT of User-Generated Forum Content Guided by OOV Word Reduction: Normalization and/or Supplementary Data? In *Proceedings of the 16th EAMT Conference*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72. Association for Computational Linguistics, Ann Arbor, Michigan.

Loïc Barrault. 2010. MANY: Open source MT system combination at WMT'10. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 277–281. Association for Computational Linguistics, Uppsala, Sweden.

Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation StatMT 09*, March, page 182. Association for Computational Linguistics.

Alexandra Birch, Phil Blunsom, and Miles Osborne. 2009. A quantitative analysis of reordering phenomena. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 197–205. Association for Computational Linguistics, Athens, Greece.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 9–16. Association for Computational Linguistics, Prague, Czech Republic.

Arianna Bisazza, Nick Ruiz, and Marcello Federico. 2011. Fill-up versus Interpolation Methods for Phrase-based SMT Adaptation. In *Proceedings of the International Workshop on Spoken Language Translation 2011*. San Francisco, CA, USA.

Graeme Blackwood, Adrià de Gispert, Jamie Brunning, and William Byrne. 2008. European language translation with weighted finite state transducers: The CUED MT system for the 2008 ACL workshop on SMT. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 131–134. Association for Computational Linguistics, Columbus, Ohio.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large Language Models in Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867.

Fabienne Braune and Alexander Fraser. 2010. Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 81–89. Association for Computational Linguistics, Beijing, China.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

Peter F. Brown, Jennifer C. Lai, and Robert L. Mercer. 1991. Aligning sentences in parallel corpora. In *Proceedings of the 29th Annual Meeting on Association for Computational Linguistics*, pages 169–176. Berkeley, California.

Noah Bubenhofer and Juliane Schröter. 2012. Die Alpen. Sprachgebrauchsgeschichte - Korpuslinguistik - Kulturanalyse. In Péter Maitz, editor, *Historische Sprachwissenschaft. Erkenntnisinteressen, Grundlagenprobleme, Desiderate*, pages 263–287. de Gruyter, Berlin.

Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. 1995. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16:1190–1208.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53. Association for Computational Linguistics, Uppsala, Sweden. Revised August 2010.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51. Association for Computational Linguistics, Montréal, Canada.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28. Association for Computational Linguistics, Athens, Greece.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64. Association for Computational Linguistics, Edinburgh, Scotland.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the Role of Bleu in Machine Translation Research. In *Proceedings the Eleventh Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256. Trento, Italy.

Nicola Cancedda, Marc Dymetman, George Foster, and Cyril Goutte. 2008. A Statistical Machine Translation Primer. In *Learning Machine Translation*, chapter 1. MIT Press.

Marine Carpuat and Mona Diab. 2010. Task-based evaluation of multiword expressions: a pilot study in statistical machine translation. In *The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 242–245. Association for Computational Linguistics, Los Angeles, CA.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72. Association for Computational Linguistics, Prague, Czech Republic.

Helena Caseli, Aline Villavicencio, André Machado, and Maria José Finatto. 2009. Statistically-driven alignment-based multiword expression identification for technical domains. In *Proceedings of the 2009 Workshop on Multiword Expressions*, pages 1–8. Suntec, Singapore.

Alexandru Ceauşu, John Tinsley, Jian Zhang, and Andy Way. 2011. Experiments on domain adaptation for patent machine translation in the PLuTO project. In *Proceedings of the 15th International Conference of the European Association for Machine Translation*. Leuven, Belgium.

Daniel Cer, Daniel Jurafsky, and Christopher Manning. 2008. Regularization and search for minimum error rate training. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 26–34. Association for Computational Linguistics, Columbus, Ohio.

Boxing Chen, Roland Kuhn, and George Foster. 2013. Vector space model for adaptation in statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1285–1293. Association for Computational Linguistics, Sofia, Bulgaria.

Stanley Chen, Douglas Beeferman, and Ronald Rosenfeld. 1998. Evaluation metrics for language models. In *DARPA Broadcast News Transcription and Understanding Workshop*.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13:359–393.

Yu Chen, Andreas Eisele, Christian Federmann, Eva Hasler, Michael Jellinghaus, and Silke Theison. 2007. Multi-engine machine translation with an open-source decoder for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 193–196. Association for Computational Linguistics, Prague, Czech Republic.

Yu Chen, Michael Jellinghaus, Andreas Eisele, Yi Zhang, Sabine Hunsicker, Silke Theison, Christian Federmann, and Hans Uszkoreit. 2009. Combining multi-engine translations with Moses. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, pages 42–46. Association for Computational Linguistics, Athens, Greece.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics, Ann Arbor, Michigan.

David Chiang. 2010. Learning to translate with source and target syntax. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1443–1452. Association for Computational Linguistics, Uppsala, Sweden.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 218–226. Association for Computational Linguistics, Boulder, CO.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 176–181. Association for Computational Linguistics, Portland, Oregon.

Trevor Cohn and Mirella Lapata. 2007. Machine Translation by Triangulation: Making Effective Use of Multi-Parallel Corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 728–735. Association for Computational Linguistics, Prague, Czech Republic.

Josep M. Crego, José B. Mariño, and Adrià De Gispert. 2004. Finite-state-based and phrase-based statistical machine translation. In *Proceedings of the 8th International Conference on Spoken Language Processing, ICSLP'04*, pages 37–40.

Josep Maria Crego and José B. Mariño. 2006. Improving statistical mt by coupling reordering and decoding. *Machine Translation*, 20(3):199–215.

Hal Daumé III and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *The 49th Annual Meeting of the Association for Computational Linguistics*, pages 407–412.

Daniel Déchelotte, Gilles Adda, Alexandre Allauzen, Hélène Bonneau-Maynard, Olivier Galibert, Jean-Luc Gauvain, Philippe Langlais, and François Yvon. 2008. LIMSI's statistical translation systems for WMT'08. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 107–110. Association for Computational Linguistics, Columbus, Ohio.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 85–91. Association for Computational Linguistics, Edinburgh, Scotland.

Matthias Eck, Stephan Vogel, and Alex Waibel. 2004. Language model adaptation for statistical machine translation based on information retrieval. In *4th International Conference on Languages Resources and Evaluation (LREC 2004)*.

Andrew Finch and Eiichiro Sumita. 2008. Dynamic model interpolation for statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 208–215. Association for Computational Linguistics, Columbus, Ohio.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459. Association for Computational Linguistics, Cambridge, Massachusetts.

George Foster and Roland Kuhn. 2007. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 128–135. Association for Computational Linguistics, Prague, Czech Republic.

George Foster and Roland Kuhn. 2009. Stabilizing minimum error rate training. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 242–249. Association for Computational Linguistics, Athens, Greece.

George F. Foster, Roland Kuhn, and Howard Johnson. 2006. Phrasetable smoothing for statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 53–61. Association for Computational Linguistics, Sydney, Australia.

Pascale Fung and Percy Cheung. 2004. Mining very-non-parallel corpora: Parallel sentence and lexicon extraction via bootstrapping and EM. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 57–63. Barcelona, Spain.

Pascale Fung and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 414–420.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Comput. Linguist.*, 19(1):75–102.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57. Association for Computational Linguistics, Columbus, Ohio.

Barry Haddow. 2013. Applying pairwise ranked optimisation to improve the interpolation of translation models. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 342–347. Association for Computational Linguistics, Atlanta, Georgia.

Barry Haddow and Philipp Koehn. 2012. Analysing the effect of out-of-domain data on smt systems. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 422–432. Association for Computational Linguistics, Montréal, Canada.

Daniel Hardt and Jakob Elming. 2010. Incremental re-training for post-editing SMT. In *Conference of the Association for Machine Translation in the Americas 2010 (AMTA 2010)*. Denver, CO.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Edinburgh, UK.

Kenneth Heafield and Alon Lavie. 2010. CMU multi-engine machine translation for WMT 2010. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, WMT '10, pages 301–306. Association for Computational Linguistics, Uppsala, Sweden.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation information retrieval. In *10th Annual Conference of the European Association for Machine Translation (EAMT 2005)*. Budapest, Hungary.

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

ISO/IEC. 2003. *ISO/IEC TR 9126-2:2003 (E). Software engineering - Product quality - Part 2: External metrics*. ISO/IEC.

Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. 2007. Improving translation quality by discarding most of the phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 967–975. Association for Computational Linguistics, Prague, Czech Republic.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, pages 181–184. Detroit, Michigan.

Philipp Koehn. 2002. Europarl: A Multilingual Corpus for Evaluation of Machine Translation. Unpublished.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*. Barcelona, Spain.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*, pages 79–86. Phuket, Thailand.

Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press.

Philipp Koehn, Amittai Axelrod, Alexandra Birch, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In *International Workshop on Spoken Language Translation*.

Philipp Koehn, Barry Haddow, Philip Williams, and Hieu Hoang. 2010. More linguistic annotation for statistical machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 115–120. Association for Computational Linguistics, Uppsala, Sweden.

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. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the ACL-2007 Demo and Poster Sessions*, pages 177–180. Association for Computational Linguistics, Prague, Czech Republic.

Philipp Koehn and Kevin Knight. 2001. Knowledge sources for word-level translation models. In Lillian Lee and Donna Harman, editors, *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 27–35.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. 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*, pages 48–54. Association for Computational Linguistics, Edmonton, Canada.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, pages 224–227. Association for Computational Linguistics, Prague, Czech Republic.

Prasanth Kolachina, Nicola Cancedda, Marc Dymetman, and Sriram Venkatapathy. 2012. Prediction of learning curves in machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22–30. Association for Computational Linguistics, Jeju Island, Korea.

Patrik Lambert and Rafael Banchs. 2006. Grouping multi-word expressions according to part-of-speech in statistical machine translation. In *Proceedings of the EACL Workshop on Multi-Word-Expressions in a Multilingual Context*, pages 9–16. Trento, Italy.

Thomas Lavergne, Alexandre Allauzen, Josep Maria Crego, and François Yvon. 2011. From n-gram-based to CRF-based translation models. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 542–553. Association for Computational Linguistics, Edinburgh, Scotland.

Hai-Son Le, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 39–48. Association for Computational Linguistics, Montréal, Canada.

David Y W Lee. 2001. Genres, registers, text types, domains, and styles clarifying the concepts and navigating a path the the BNC jungle. *Language Learning and Technology*, 5(3):37–72.

Gennadi Lembersky, Noam Ordan, and Shuly Wintner. 2012. Adapting translation models to translationese improves SMT. In *Proceedings of the 13th Conference of the European*

Chapter of the Association for Computational Linguistics, pages 255–265. Association for Computational Linguistics, Avignon, France.

Chin-Yew Lin and Franz Josef Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proceedings of Coling 2004*, pages 501–507. COLING, Geneva, Switzerland.

Adam Lopez. 2012. Putting human assessments of machine translation systems in order. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 1–9. Association for Computational Linguistics, Montréal, Canada.

Xiaoyi Ma. 2006. Champollion: A robust parallel text sentence aligner. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

José B. Mariño, Rafael E. Banchs, Josep Maria Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-Jussà. 2006. *N*-gram-based machine translation. *Computational Linguistics*, 32(4):527–549.

Spyros Matsoukas, Antti-Veikko I. Rosti, and Bing Zhang. 2009. Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, pages 708–717. Association for Computational Linguistics, Singapore.

Robert C. Moore. 2002. Fast and accurate sentence alignment of bilingual corpora. In *AMTA '02: Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users*, pages 135–144. Springer-Verlag, London, UK.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 220–224. Association for Computational Linguistics, Uppsala, Sweden.

Preslav Nakov. 2008. Improving English-Spanish statistical machine translation: experiments in domain adaptation, sentence paraphrasing, tokenization, and recasing. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 147–150. Association for Computational Linguistics, Columbus, Ohio.

Preslav Nakov and Hwee Tou Ng. 2009. Improved statistical machine translation for resource-poor languages using related resource-rich languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1358–1367. Association for Computational Linguistics, Singapore.

Preslav Nakov and Jörg Tiedemann. 2012. Combining word-level and character-level models for machine translation between closely-related languages. In *The 50th Annual Meeting of the Association for Computational Linguistics*, pages 301–305.

Peter Newmark. 1991. *About translation*. Multilingual matters. Multilingual Matters.

Jan Niehues and Alex Waibel. 2010. Domain adaptation in statistical machine translation using factored translation models. In *Proceedings of 14th Annual Conference of the European Association for Machine Translation (EAMT'10)*, May.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167. Association for Computational Linguistics, Sapporo, Japan.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *HLT-NAACL 2004: Main Proceedings*, pages 161–168. Association for Computational Linguistics, Boston, Massachusetts, USA.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 295–302. Association for Computational Linguistics, Philadelphia, PA.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *EMNLP'99*, pages 20–28.

Hideo Okuma, Hirofumi Yamamoto, and Eiichiro Sumita. 2007. Introducing translation dictionary into phrase-based SMT. In *Proceedings of the MT Summit XI*.

Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. 2010. Online learning for interactive statistical machine translation. In *HLT-NAACL*, pages 546–554. The Association for Computational Linguistics.

Santanu Pal, Tanmoy Chakraborty, and Sivaji Bandyopadhyay. 2011. Handling multiword expressions in phrase-based statistical machine translation. In *Proceedings of the 13th Machine Translation Summit*, pages 215–224. Xiamen, China.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, Philadelphia, PA.

Bruno Pouliquen and Christophe Mazenc. 2011. COPPA, CLIR and TAPTA: three tools to assist in overcoming the patent language barrier at WIPO. In *Proceedings of the 13th Machine Translation Summit*, pages 24–30. Xiamen, China.

Majid Razmara, George Foster, Baskaran Sankaran, and Anoop Sarkar. 2012. Mixing multiple translation models in statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Jeju, Republic of Korea.

Stefan Riezler and John T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64. Association for Computational Linguistics, Ann Arbor, Michigan.

Anthony Rousseau. 2012. *La traduction automatique de la parole*. Ph.D. thesis, Université du Maine.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann A. Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, pages 1–15. Springer-Verlag, London, UK.

Holger Schwenk. 2007. Continuous space language models. *Comput. Speech Lang.*, 21(3):492–518.

Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In *24th International Conference on Computational Linguistics (COLING)*. Bombay, India.

Holger Schwenk, Jean-Baptiste Fouet, and Jean Senellart. 2008. First steps towards a general purpose French/English statistical machine translation system. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 119–122. Association for Computational Linguistics, Columbus, Ohio.

Holger Schwenk and Philipp Koehn. 2008. Large and diverse language models for statistical machine translation. In *International Joint Conference on Natural Language Processing*, pages 661–666.

Rico Sennrich. 2011a. Combining multi-engine machine translation and online learning through dynamic phrase tables. In *15th Annual Conference of the European Association for Machine Translation (EAMT 2011)*, pages 89–96. Leuven, Belgium.

Rico Sennrich. 2011b. The UZH system combination system for WMT 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 166–170. Edinburgh, UK.

Rico Sennrich. 2012a. Mixture-modeling with unsupervised clusters for domain adaptation in statistical machine translation. In *16th Annual Conference of the European Association for Machine Translation (EAMT 2012)*, pages 185–192. Trento, Italy.

Rico Sennrich. 2012b. Perplexity minimization for translation model domain adaptation in statistical machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 539–549. Association for Computational Linguistics, Avignon, France.

Rico Sennrich. 2013. Promoting flexible translations in statistical machine translation. In *Proceedings of Machine Translation Summit XIV*. Nice, Fance.

Rico Sennrich, Holger Schwenk, and Walid Aransa. 2013. A multi-domain translation model framework for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sofia, Bulgaria.

Rico Sennrich and Martin Volk. 2010. MT-based sentence alignment for OCR-generated parallel texts. In *The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*. Denver, Colorado, USA.

Rico Sennrich and Martin Volk. 2011. Iterative, MT-based sentence alignment of parallel texts. In *NODALIDA 2011, Nordic Conference of Computational Linguistics*.

Kashif Shah, Loïc Barrault, and Holger Schwenk. 2010. Translation Model Adaptation by Resampling. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 392–399. Association for Computational Linguistics, Uppsala, Sweden.

Kashif Shah, Loïc Barrault, and Holger Schwenk. 2012. A general framework to weight heterogeneous parallel data for model adaptation in statistical machine translation. In *Conference of the Association for Machine Translation in the Americas 2012 (AMTA 2012)*. San Diego, California, USA.

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656.

Claude E. Shannon. 1951. Prediction and entropy of printed English. *Bell System Technical Journal*, 30:50–64.

Michel Simard, George F. Foster, and Pierre Isabelle. 1993. Using cognates to align sentences in bilingual corpora. In *CASCON '93: Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research*, pages 1071–1082. IBM Press, Toronto, Ontario, Canada.

Frank Smadja. 1993. Retrieving collocations from text: Xtract. *Computational Linguistics*, 19:1, Special issue on using large corpora:143–177.

Frank Smadja, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: a statistical approach. *Computational Linguistics*, 22(1):1–38.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Daniel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*. Genoa, Italy.

Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. In *Seventh International Conference on Spoken Language Processing*, pages 901–904. Denver, CO.

Jörg Tiedemann. 2007. Improved sentence alignment for movie subtitles. In *Proceedings of Recent Advances in Natural Language Processing (RANLP 2007)*. Borovets, Bulgaria.

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

Marco Turchi, Tijl De Bie, and Nello Cristianini. 2009. Learning to translate: a statistical and computational analysis. Technical report, University of Bristol.

Dániel Varga, László Németh, Péter Halácsy, András Kornai, Viktor Trón, and Viktor Nagy. 2005. Parallel corpora for medium density languages. In *Proceedings of Recent Advances in Natural Language Processing (RANLP 2005)*, pages 590–596.

Martin Volk, Noah Bubenhofer, Adrian Althaus, Maya Bangerter, Lenz Furrer, and Beni Ruef. 2010a. Challenges in building a multilingual alpine heritage corpus. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. Valletta, Malta.

Martin Volk, Anne Göhring, Torsten Marek, and Yvonne Samuelsson. 2010b. SMULTRON (version 3.0) – The Stockholm MULtilingual parallel TReebank. http://www.cl.uzh.ch/research/paralleltreebanks.html.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773. Association for Computational Linguistics, Prague, Czech Republic.

Warren Weaver. 1949/1955. Translation. In William N. Locke and A. Donald Boothe, editors, *Machine Translation of Languages*, pages 15–23. MIT Press, Cambridge, MA. Reprinted from a memorandum written by Weaver in 1949.

Hirofumi Yamamoto and Eiichiro Sumita. 2007. Bilingual cluster based models for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 514–523. Prague, Czech Republic.

Keiji Yasuda and Eiichiro Sumita. 2008. Method for building sentence-aligned corpus from wikipedia. In *2008 AAAI Workshop on Wikipedia and Artificial Intelligence (WikiAI08)*.

Keiji Yasuda, Ruiqiang Zhang, Hirofumi Yamamoto, and Eiichiro Sumita. 2008. Method of selecting training data to build a compact and efficient translation model. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP)*.

Eray Yildiz and A. Cüneyd Tantuğ. 2012. Evaluation of sentence alignment methods for English-Turkish parallel texts. In *First Workshop on Language Resources and Technologies for Turkic Languages*, pages 64–67.

Bing Zhao, Matthias Eck, and Stephan Vogel. 2004. Language model adaptation for statistical machine translation with structured query models. In *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, Geneva, Switzerland.

10 Appendix

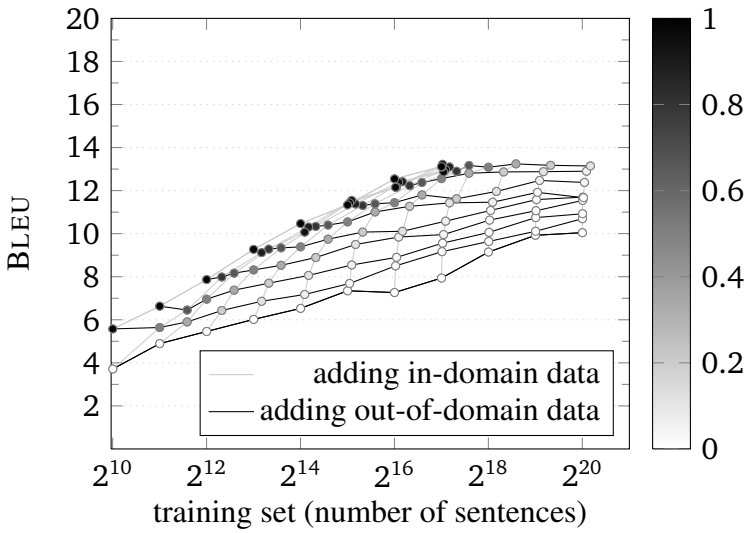

Figure 10.1: BLEU score as a function of training set size, illustrating the relative effect of adding in-domain (IN) and out-of-domain (OUT) training data. Data point colour denotes the relative amount of IN data. Translation direction FR–DE.

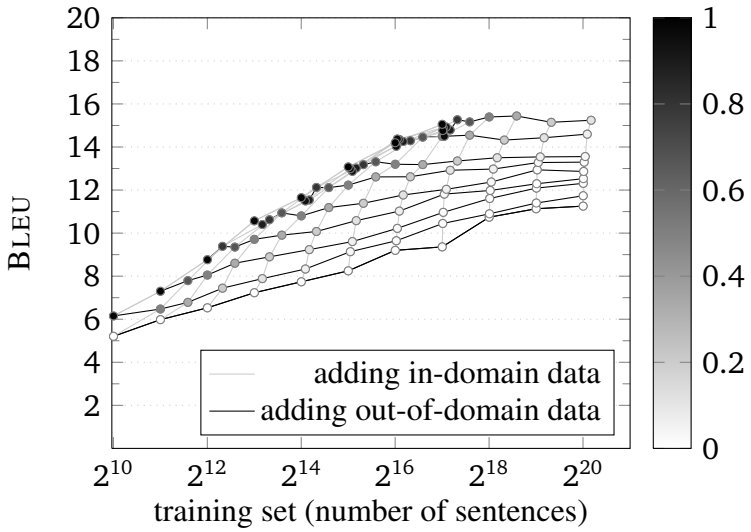

Figure 10.2: BLEU score as a function of training set size, illustrating the relative effect of adding in-domain (IN) and out-of-domain (OUT) training data. Out-of-domain language model. Data point colour denotes the relative amount of IN data. Translation direction DE–FR.

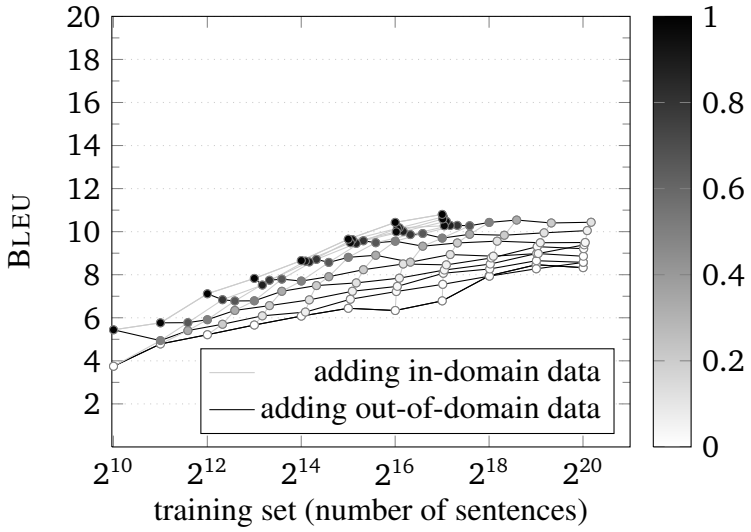

Figure 10.3: BLEU score as a function of training set size, illustrating the relative effect of adding in-domain (IN) and out-of-domain (OUT) training data. Out-of-domain language model. Data point colour denotes the relative amount of IN data. Translation direction FR–DE.