UNIVERSITÉ PARIS-SUD

ECOLE DOCTORALE D'INFORMATIQUE
LABORATOIRE D'INFORMATIQUE
POUR LA MÉCANIQUE ET LES SCIENCES DE L'INGÉNIEUR

DISCIPLINE : INFORMATIQUE

THÈSE DE DOCTORAT

Soutenue le 25/11/2014 par

Li Gong

On-demand Development of Statistical Machine Translation Systems

Acknowledgement

My first and biggest gratitude is to my two thesis supervisors: François Yvon and Aurélien Max. It is my greatest pleasure to work with them. During the three and a half years at LIMSI-CNRS, their guidance and kindness constantly encouraged me to improve my competency on research and foreign languages. Thanks to them, I have became the most educated person in my whole family.

I would like to thank the other members of the jury, Marc Dymetman, Andy Way, Christian Jacquemin, Béatrice Daille and Josep M. Crego, for their comments, analyses, questions and suggestions. Special thanks to Marc Dymetman and Andy Way for spending a lot of time and effort to review this thesis and offer advice to improve my work.

It is my pleasure to work at LIMSI-CNRS. The TLP group is a very dynamic and nice group. I am fortunate to work with so many talented people and learn from them. Aurélien Max always tried to use simple French words to make sure I can understand. François Yvon, apart from the research, also taught me how to find deleted data from the backup system. Hai Son Le provided me the links to all necessary data that I needed for my work. Benjamin Marie helped me to write correct French e-mail. William Hartmann corrected many language errors in my paper. Elena Knyazeva promoted me as the "chef" of the office. Quoc Khanh Do explained many techniques of his system to me. I also have fruitful discussions with other members in the group: Alexandre Allauzen, Marianna Apidianaki, Hélène Bonneau-Maynard, Marco Dinarelli, Thiago Fraga da Silva, Souhir Gahbiche-Braham, Yulia Ivanishcheva, Thomas Lavergne, LinLin Li, Nicolas Pécheux, Guillaume Wisniewski, Yong Xu, Fan Yang ... The daily coffee/tea breaks with them and the interesting discussions, such as how to tie your shoes and the correct way to peel a banana, are unforgettable memories. I also want to thank Sophie Pageau-Maurice for helping me a lot to handle the visa paper work.

I would like to thank Yuling Zheng for accompanying me throughout every weekend of August when I went to the laboratory, ate cold sandwich at lunch and wrote the thesis.

Finally, my biggest gratitude is undoubtedly to my parents who are always beside me and supporting me.

To supertudou.

Contents

1 Overview of Statistical Machine Translation **3**

1.1 Statistical machine translation 4

1.2 Word alignment 5

1.3 Phrase-based probabilistic models 6

1.3.1 Target language model 7

1.3.2 Reordering models 9

1.4 Decoding and tuning 10

1.4.1 Scoring function 11

1.4.2 Automatic evaluation 11

1.4.3 Parameter tuning 13

1.5 Computer-Assisted Translation (CAT) 13

1.6 Summary 14

Introduction **15**

I Development of Statistical Machine Translation Systems **17**

2 Overview of Statistical Machine Translation **19**

2.1 Statistical machine translation 20

2.2 Word alignment 21

2.3 Phrase-based probabilistic models 22

2.3.1 Target language model 23

2.3.2 Reordering models 25

2.4 Decoding and tuning 26

2.4.1 Scoring function 27

2.4.2 Automatic evaluation 27

2.4.3 Parameter tuning 29
2.5 Computer-Assisted Translation (CAT) 29
2.6 Summary 30

3 Improving SMT System Development 31
3.1 Scaling to larger corpora 32
3.2 Incremental system training 33
3.2.1 Incremental alignment 34
3.2.2 Model integration 35
3.3 Data weighting schemes 36
3.3.1 Context-dependent models 36
3.3.2 Data selection 38
3.3.3 Instance weighting 40
3.3.4 Multi-domain adaptation 41
3.3.5 Translation memory integration 42
3.4 Summary 43

II On-demand Development and Contextual Adaptation for SMT 45

4 On-demand Development Framework of SMT Systems 47
4.1 On-demand word alignment 48
4.1.1 Sampling-based transpotting 49
4.1.2 Sub-sentential alignment extraction 50
4.1.3 Difference with the `giza++` alignment process 53
4.2 On-the-fly model estimation 55
4.3 On-demand system development 56
4.4 Experimental validation 57
4.4.1 Data 57
4.4.2 Validation of on-the-fly model estimation 58
4.4.3 Validation of on-demand word alignment 60
4.4.4 Validation of the framework 66
4.5 Summary 67

5 Incremental Adaptation with On-demand SMT **69**

5.1 Translation for communities with on-demand SMT . . . 70

5.1.1 No cache, no tuning (`Config0`) . . . 70

5.1.2 Using a cache (`Config1`) . . . 71

5.1.3 Sampling by reusing aligned sentences (`Config2`) . . . 72

5.1.4 Plug-and-play data integration (`+spec`) . . . 74

5.1.5 Simple online tuning (`+online`) . . . 76

5.2 Any-text translation with on-demand SMT . . . 78

5.3 Summary . . . 82

6 Contextual Adaptation for On-demand SMT **85**

6.1 Contextual sampling strategies . . . 86

6.1.1 N-gram precision . . . 86

6.1.2 TF-IDF . . . 87

6.1.3 On-demand system construction with contextual sampling . . . 87

6.2 Confidence estimation of adapted models . . . 88

6.3 Experimental validation . . . 89

6.3.1 Results . . . 90

6.4 Contextual adaptation of on-demand SMT systems . . . 92

6.5 Summary . . . 95

Conclusion **97**

Appendices

Appendix A Abbreviation **103**

Appendix B Extracts of Data **105**

B.1 `Newstest` . . . 105

B.2 `WMT'14-med` . . . 105

B.3 `Cochrane` . . . 106

Appendix C Documents in Any-text Translation **107**

Appendix D Translation for communities: translation examples **109**

List of Figures

1.1 An example of word alignment between a French sentence and its English translation. (reproduced from [?, chapter 7]). 5

1.2 Lexical weight of a phrase pair $(\bar{s}, \bar{t})$ given an alignment $\mathbf{a}$ and lexical translation probabilities. (reproduced from [?, p. 140]) 8

1.3 Illustration of different orientation types for reordering models. 10

1.4 Ten different human translations of the same Chinese sentence from the 2001 NIST evaluation set. (reproduced from [?, p. 218]) 12

2.1 An example of word alignment between a French sentence and its English translation. (reproduced from [?, chapter 7]). 21

2.2 Lexical weight of a phrase pair $(\bar{s}, \bar{t})$ given an alignment $\mathbf{a}$ and lexical translation probabilities. (reproduced from [?, p. 140]) 24

2.3 Illustration of different orientation types for reordering models. 26

2.4 Ten different human translations of the same Chinese sentence from the 2001 NIST evaluation set. (reproduced from [?, p. 218]) 28

3.1 Illustration of suffix array for a very small corpus (reproduced from [?]). 33

4.1 Illustration of the sampling-based transpotting method on an English-French sentence pair. 51

4.2 The computation of association score between English word *coke* and French word *coca* using the association table in Figure 4.1. 52

4.3 Straight rule and inversion rule for the segmentation of a pair of segments, where the gray blocks indicats that the corresponding two segments are aligned. 52

4.4 Illustration of the recursive alignment process. The number in each cell corresponds to the value of the association score, with $0 < \epsilon \leq 0.001$. A null value indicates that the two words never appear together in the translation table. Alignment points retained by the algorithm, i.e. at maximum level of recursion, are shown in grey. 54

4.5 Possible word alignments for an extract of French-English sentence pair. 54

4.6 Main processes of our framework (delimited by a red frame). 57

4.7 Example of matrices on French-English obtained using two mgiza++ baselines and our on-demand word alignment (owa). 66

5.1 Percentage of n-grams found in previous documents in the Cochrane document sequence. 72
5.2 Evolution of the cache size for a sequence of Cochrane documents. 73
5.3 Evolution of the average per token processing time for a sequence of documents. 74
5.4 Document-level comparison with the vanilla moses system. The y-axis represents the difference in BLEU score (ΔBLEU) between our systems and the vanilla moses system for each document in the sequence. 75
5.5 Document-level comparison with the vanilla moses system with online tuning. Initialization either uses moses default values (+spec+online), or parameters tuned on a development set (+dev). 76
5.6 Document-level comparison with the vanilla moses system with online tuning. Initialization either uses moses default values (+spec+online), or parameters tuned on a development set (+dev). 77
5.7 Experimental results on the book1 document. 81
5.8 Experimental results on the book2 document. 81
5.9 Experimental results on the php document. 82

6.1 Percentage of correctly translated source phrases in the trace of the decoder of the TF-IDF+conf systems against score value intervals of the confidence model (conf) in the Newstest translation task. The dashed line represents the number of phrases in each confidence value interval. 92
6.2 Document-level comparison with the moses system with online tuning. 94

List of Tables

4.1 Notations used in the remainder of this thesis. 49

4.2 Description of corpora used in our experiments. 59

4.3 Translation performance (using mean ± standard deviation of 3 runs, * significant at $p < 0.01$ level) with different sampling sizes (M). 61

4.4 Description of the corpora used for validation experiments of on-demand word alignment. 62

4.5 Description of the tuning and test sets for the `HIT` corpus. 62

4.6 Performance (using mean ± standard deviation of 3 runs, * significant at $p < 0.01$ level) of the `owa` word alignment method. 63

4.7 Percentage of tokens in selected frequency intervals. 63

4.8 Performance (using mean ± standard deviation of 3 runs) of the `owa` word alignment method on `HIT` corpus. 64

4.9 Results of experiments where a supplementary corpus is pooled and aligned by several methods. 65

4.10 Experimental results for large-scale experiments on three MT tasks. (* significant at $p < 0.01$ level) 68

5.1 Results for the `owa` system on a large-scale English-to-French translation task. The reference system is the vanilla `moses` system (cf. Table 4.10), tuned on the `Cochrane` development set. `+dev` is identical as `Config2+spec+online`, with initial weights tuned on the `Cochrane` development set. 71

5.2 Description of the documents from various domains. 79

5.3 Any-text machine translation results for English-to-French translation. 80

6.1 Experimental results for three MT tasks, in which all tasks use 100 as sample size except the `WMT'14-med` (English → French) task that uses 500 as sample size. (* significant at $p < 0.01$ level) 90

6.2 Average number of translations and average entropy of source phrases in the translation tables (considering only source phrases with a frequency higher than M). Experiments marked by * use $M = 500$, all the others experiments use $M = 100$. 91

6.3 Incremental development of SMT systems with contextual sampling (`TF-IDF`). 93

1 Overview of Statistical Machine Translation

Machine Translation (MT) is the field of computer science that studies the automatic translation from one natural language to another. It is known to be one of the very first applications of **Natural Language Processing** (NLP), while, ironically, being one of the most difficult ones, as it encompasses issues in language analysis, transfer and generation. Following the initial attempts at building MT systems, exemplified by IBM's efforts in Russian-English MT[1], a significant amount of research has been devoted to MT, with various approaches introduced and improved up to the current days.

During the 1970s, **Rule-based Machine Translation** (RBMT) was the dominent approach, with translation consisting in a series of analysis processes (morphological, syntactic and/or semantic) of the input text, followed by a process of text generation of the target text. The steps of each process are controlled by a dictionary and a grammar which are obtained through inspection by linguists, entailing a slow and time-consuming development process faced with the '*knowledge-acquisition bottleneck*'.

An alternative direction to MT was introduced to overcome these issues. **Corpus-based (or Data-driven) Machine Translation** attempts to derive the knowledge necessary to produce new translations from existing **bilingual parallel corpora**, made of translated text units, typically sentences. **Example-based Machine Translation** (EBMT) [**?**] systems, which were introduced in the early 1980s, takes sentences as basic translation units. If a sentence to translate exists exactly in the parallel corpus, its translation is extracted and reused as the system's output. Otherwise, the system searches for similar sentences in the corpus, and suitable subsequences of the sentence are iteratively replaced, modified or adapted in order to generate the new translation.

The increased availability of computational power and necessary resources progressively allowed the development of more computationally intensive approaches to MT. **Statistical Machine Translation** (SMT), introduced by **?** in **?**, is a data-driven approach characterized by its implementation of a highly developed mathematical theory. Generally, SMT systems learn

[1]For more early history of MT, see [**?**].

a **translation model** from a bilingual parallel corpus and a **language model** from a monolingual corpus. When translating, the best translation is the one that maximizes a predefined score according to the models. Additionally, the development of automatic translation **evaluation metrics** and of several free and open source SMT toolkits, have gradually facilitated the implementation and evaluation of translation systems. Furthermore, constant increases in computing power and availability of large bilingual paralllel corpora have made SMT the dominant approach in the early 2010s.

With regard to translation units, SMT can be divided into three groups: word-based, phrase-based and syntax-based SMT. Due to its simplicity and effective modeling of local contexts, the phrase-based variant is now considered a state-of-the-art choice for many language pairs. The work presented in this thesis is framed in the phrase-based framework, and the remainder of this chapter will provide a brief introduction to phrase-based SMT; other approaches to SMT are described in details in several reference works [**????**].

In this chapter, we will give a brief overview of SMT, focusing mostly on phrase-based SMT. Since the technologies of statistical machine translation have already been described in detail in many existing references, we only introduce the general principles and go into detail only for the parts that will be directly relevant to the work presented in this thesis. The content of this chapter is structured as follows. Section 2.1 and 2.2 are devoted to the general principles of SMT and word alignment. The statistical models used in SMT systems are presented in Section 2.3. The decoding process, automatic evaluation of translation quality and system tuning are described in Section 2.4. A summary of this chapter is finally given in Section 2.6.

1.1 Statistical machine translation

The base material for building an SMT system is a sentence-aligned bilingual corpus[2], where we denote the language of the given text by *source language* and the one the source text will be translated into by *target language*. Making the strong assumption that the translations can be word-aligned, the training parallel corpus is automatically analyzed to generate alignments between the words in source language and their translations in target language. From the resulting **word alignment** of the parallel corpus, a number of statistical models are estimated, including a **translation model**. A **language model** is also estimated using large quantities of text in the target language. The relative importance of all the estimated models are optimized on a development corpus before being applied to translate new source language sentences, which do not exist in the parallel corpus.

Translation generation is treated as a search problem [**?**], the goal of which is to find the most probable translation candidate $\mathbf{t} = t_1^J = t_1 \dots t_J$ for a given source language sentence $\mathbf{s} = s_1^I = s_1 \dots s_I$, where I and J represent the length of the source and target sentence, respectively. The best translation can be obtained by maximizing $P(\mathbf{t}|\mathbf{s})$ and applying the noisy channel model [**?**], defined in Equation (2.1):

[2]Note that, in such sentence-aligned bilingual corpora, which are symmetric and could be used for translations in both directions, some parts are originally written in the source language and translated into the target language, some parts are originally written in target language and translated into the source language, some parts are even written in a third language and then translated into both the source and target languages. This property have a strong influence on the SMT performance [**??**]

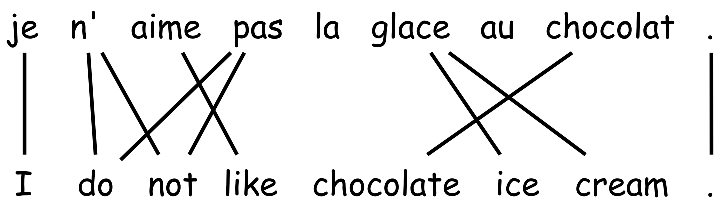

je n' aime pas la glace au chocolat .
I do not like chocolate ice cream .

Figure 1.1 – An example of word alignment between a French sentence and its English translation. (reproduced from [**?**, chapter 7]).

$$\begin{aligned}\mathbf{t}_{best} &= \underset{J,t_1^J}{\operatorname{argmax}}\, p(t_1^J|s_1^I) \\ &= \underset{J,t_1^J}{\operatorname{argmax}}\, p(s_1^I|t_1^J)p(t_1^J)\end{aligned} \tag{1.1}$$

where $p(s_1^I|t_1^J)$ denotes the translation model and $p(t_1^J)$ denotes the target language model. The translation model is in practice defined as a set of models, that can include translation, fertility, and distortion probabilities from the IBM word alignment models (see Section 2.2). They are estimated using the Expectation Maximization (EM) algorithm [**?**] on the bilingual parallel corpus.[3] Although the IBM models are no longer the state-of-the-art in translation modeling, they are still considered as state-of-the-art in word alignment.

1.2 Word alignment

A word alignment between parallel sentences attempts to represent the translations between words in the source sentence and words in the target sentence. An example of word alignment of a parallel sentence pair is illustrated in Figure 2.1. As these alignments are seldom explicitly represented in parallel data, word alignment models are usually learnt from *incomplete* data using the EM algorithm, an unsupervised Machine Learning approach. The translation model defined in Equation (2.1) can be written as in Equation (2.2):

$$p(s_1^I|t_1^J) = \sum_{a_1^J} p(s_1^I, a_1^J|t_1^J) \tag{1.2}$$

where a_1^J denotes the word alignments between s_1^I and t_1^J.

The IBM word alignment models [**?**] include five models of increasing complexity:

- **IBM Model 1** models lexical translation, in which all possible alignments are considered as equally probable. It does not model the issue of word reordering across languages.
- **IBM Model 2** adds a reordering model based on the absolute positions of aligned words in the source and target sentences.

[3]A detailed account can be found in [**?**].

- **IBM Model 3** adds a fertility model which introduces the possibility of one-to-many, one-to-zero and zero-to-one (by `NULL` insertion) translation.
- **IBM Model 4** adds a relative alignment model, in which the position of the translation of an input word is typically based on the position of the translation of the previous input word; and a dependency on word classes for distortion models to deal with data sparsity problems.
- **IBM Model 5** fixes the problem of Model 3 and Model 4 which allow multiple output words to be placed at the same position.

? proposed to model word alignment as a first-order Hidden Markov Model (HMM) [**?**], in which an alignment position a_j depends on the previously aligned position a_{j-1}. According to this model, the translation probability factors as:

$$P(\mathbf{t}, \mathbf{a}|\mathbf{s}) = P(J|I) \prod_{j=1}^{J} P(a_j|a_{j-1}, I) P(t_j|s_{a_j}) \tag{1.3}$$

where, $P(J|I)$ is the sentence length probability, $P(a_j|a_{j-1}, I)$ is the transition probability, and $P(t_j|s_{a_j})$ is the emission probability as defined by IBM model 1. The HMM model has attractive properties which permits numerous extensions: a comparative overview of various alignment methods is provided in [**??**].

1.3 Phrase-based probabilistic models

In contrast to word-based models, the translation unit in phrase-based models [**??**] is the *phrase*, that is, a sequence of words. An input sentence is translated phrase by phrase, offering the following advantages:

- By using phrases, a system can seamlessly handle the cases of many-to-one translations (and vise versa), where words may not be the best translation units. For example, the French word *pomme de terre* could be treated as a single unit and translated into *potato*. This characteristic is useful for translating idioms and generally phrases whose translation cannot be composed using word-level translations only.
- The use of phrases can significantly reduce lexical ambiguity during translation. For example, the English word form *press* could be translated into at least *appuie* or *appuyez* depending on the subject of the verb. Translating this word within a phrase such as *I press* eliminates the lexical ambiguity regarding the correct word form in the target language. However, such ambiguity reduction is limited, as it only works when contextual information is immediately neighboring the ambiguous word.
- Phrases can also contain cases of translation reordering. For example, in Figure 2.1, the French phrase *glace au chocolat* can be translated into English as *chocolate ice cream*, whose internal reordering is inherently captured in a bilingual phrase.

Phrase-based translation models are estimated from a word-aligned parallel bilingual corpus. Using the parallel corpus and its word alignments, **phrase pairs**, which associate a phrase in the source language and its translation in the target language, can be extracted from each sentence pair. In the current state-of-the-art approaches, a phrase pair can be extracted if and only if it is consistent with the word alignment. More formally, a phrase pair $(\bar{s}, \bar{t})$ is said to be consistent with an alignment $\mathbf{a}$ if all words in $\bar{s}$ that have alignment points in $\mathbf{a} = \{(i, j) \subset \{1 \ldots I\} \times \{1 \ldots J\}\}$ have these with words included in $\bar{t}$ and vice versa:

$$\begin{aligned} (\bar{s}, \bar{t}) &\text{ consistent with } \mathbf{a} \Leftrightarrow \\ \forall s_i \in \bar{s} &: (s_i, t_j) \in \mathbf{a} \Rightarrow t_j \in \bar{t} \\ \texttt{AND } \forall t_j \in \bar{t} &: (s_i, t_j) \in \mathbf{a} \Rightarrow s_i \in \bar{s} \\ \texttt{AND } \exists s_i \in \bar{s}, t_j \in \bar{t} &: (s_i, t_j) \in \mathbf{a} \end{aligned} \tag{1.4}$$

After the extraction of all phrase pairs present in the parallel corpus, translation probabilities between source and target phrases can be obtained by taking the relative frequency of the phrase pair given the source or target phrase:

$$p(\bar{t}|\bar{s}) = \frac{\text{count}(\bar{s}, \bar{t})}{\sum_{\bar{t}'} \text{count}(\bar{s}, \bar{t}')} \tag{1.5}$$

This translation probability is typically computed in both translation directions : $p(\bar{t}|\bar{s})$ and $p(\bar{s}|\bar{t})$.

Infrequent phrase pairs may cause problems, especially if they are collected from noisy data. For instance, if both source phrase $\bar{s}$ and the target phrase $\bar{t}$ only occur once in the training corpus $\mathbf{C}$, then $p(\bar{t}|\bar{s}) = p(\bar{s}|\bar{t}) = 1$, which is clearly an over-estimate of how reliable this phrase pair is. To overcome this problem, phrase pairs can be decomposed into their word translations so that it becomes possible to estimate how lexically coherent they are by means of a score called **lexical weighting**. Given the word-level alignment between two phrases in a pair, the lexical translation probability of a phrase $\bar{t}$ given the phrase $\bar{s}$ can be computed by:

$$lex(\bar{t}|\bar{s}, \mathbf{a}) = \prod_{j=j_{start}}^{j_{end}} \frac{1}{|\{i|(i,j) \in \mathbf{a}\}|} \sum_{\forall (i,j) \in \mathbf{a}} w(t_j|s_i) \tag{1.6}$$

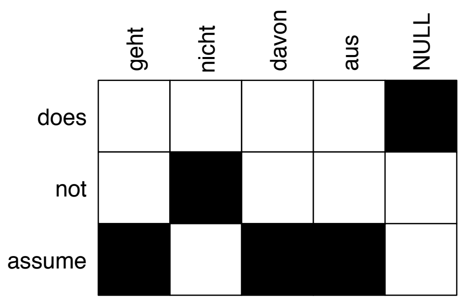
where $w(t_j|s_i)$ represents the word translation probability of the target language word t_j given the source language word s_i. If a target language word is aligned to several source language words, the average of the corresponding word translation probabilities is used. If a target language word is not aligned to any source language word, it is said to be aligned to the `NULL` word, which is also factored in as a word translation probability. This computation is illustrated in Figure 2.2. Lexical translation probabilities $w(t_j|s_i)$ can be simply estimated from the word aligned corpus using:

$$w(t|s) = \frac{\text{count}(s, t)}{\sum_{t'} \text{count}(s, t')} \tag{1.7}$$

Similarly to the phrase translation probability, the lexical weighting are also used in both translation directions: $lex(\bar{t}|\bar{s}, \mathbf{a})$ and $lex(\bar{s}|\bar{t}, \mathbf{a})$.

1.3.1 Target language model

Another essential component in a SMT system is a target **language model** (LM), whose role it is to measure how fluent and likely a sequence of words is so as to lead artificial systems to generate texts that be as much comprehensible as possible. A statistical language model provides



$$\begin{aligned} lex(\bar{t}|\bar{s},\mathbf{a}) = & w(\text{does}|\texttt{NULL})\times \\ & w(\text{not}|\text{nicht})\times \\ & \frac{1}{3}(w(\text{assume}|\text{geht}) + w(\text{assume}|\text{davon}) + w(\text{assume}|\text{aus})) \end{aligned}$$

Figure 1.2 – Lexical weight of a phrase pair $(\bar{s},\bar{t})$ given an alignment $\mathbf{a}$ and lexical translation probabilities. (reproduced from [**?**, p. 140])

an estimation of the likeliness of all possible word strings. Mathematically, by considering natural language as a stochastic process, a LM is formulated as a probability distribution $p(t_1^J)$ over sequences of words t_1^J in $\mathcal{V}^+$ where $\mathcal{V}$ is a finite vocabulary. For instance, a probabilistic language model should assign a much larger probability for "*she eats avocado salad*" than for any of "*she eats lawyer salad*", "*eats she salad avocado*" or "*she eat avocado salad*" because, intuitively, the first one is more likely to occur than the latter ones in English. The above examples show that a language model helps to improve MT in at least three aspects: semantics, syntax and grammaticality.

Trying to directly estimate the probability of a whole string is not tractable. In **n-gram language modeling**, the estimation of such a probability for a sequence t_1^J is usually broken up into predicting one word at a time. The probability $p(t_1^J)$ is therefore usually factorized in a left-to-right manner as:

$$p(t_1^J) = p(t_1)p(t_2|t_1)\dots p(t_J|t_1,t_2\dots t_{J-1}) \tag{1.8}$$

The language model probability $p(t_1^J)$ is thus a product of word probabilities given a **history** of preceding words. In applications such a Machine Translation, a history is usually limited to $n-1$ words:

$$p(t_j|t_1,t_2\dots t_{j-1}) \simeq p(t_j|t_{j-n+1}\dots t_{j-1}) \tag{1.9}$$

Considering the training data size and the computational cost, the value of n is usually small. Most commonly, **3-gram** or **4-gram** language models are used, which consider respectively the two or the three previous words as history to predict the next word. For instance, estimating the probability of a 3-gram $p(t_3|t_1,t_2)$ can be obtained by computing:

$$p(t_3|t_1,t_2) = \frac{\text{count}(t_1,t_2,t_3)}{\sum_{t_3'}\text{count}(t_1,t_2,t_3')} \tag{1.10}$$

where count(t_1,t_2,t_3) and count(t_1,t_2,t_3') represent how often the word sequences $t_1t_2t_3$ and $t_1t_2t_3'$ occur in the training corpus, respectively. This model estimation approach is also faced with the data sparseness problem. Several techniques, such as *smoothing*, *interpolation* and *back-off*, can be applied to improve the quality of the language model [**???**].

1.3.2 Reordering models

As illustrated in Figure 2.1, words or phrases from the target language may be reordered relative to their translation equivalent in the source language. In a phrase-based SMT system, this is handled by so-called **reordering models**, which can be of two main types: **distance-based** or **lexicalized**. Moreover, a recent trend has been to consider *pre-ordering* methods, where source sentences are reordered in a pre-processing step to match the target word order and then fed into the standard Phrase-Based pipeline [**????**].

Distance-based reordering models consider phrase reordering relative to the previous phrase in the source language. Let $start_i$ be the position of the first word of the source input phrase that translates to the ith target phrase, and end_i be the position of the last word of that source phrase. A reordering distance is computed as $start_i - end_{i-1} - 1$. This distance represents the number of words skipped (either forward or backward) when taking source words out of sequence. If two phrases are translated in sequence, then $start_i = end_{i-1} + 1$ and the distance is null.

The distance-based reordering model proposed for phrase-based SMT is thus only conditioned on movement distance. It is uniformly applied to all source phrases. However, some phrases are reordered more frequently than others, suggesting to consider instead a lexicalized reordering model that conditions reordering on the actual phrases being translated.

Lexicalized reordering models consider a sequence of orientations $o = (o_1 \ldots o_n)$, and the counts of how often each extracted phrase pair is found with each of the orientation types. A probability distribution p_o is estimated based on these counts using maximum likelihood:

$$p_o(o|\bar{s},\bar{t}) = \frac{\text{count}(o,\bar{s},\bar{t})}{\sum_o \text{count}(o,\bar{s},\bar{t})} \tag{1.11}$$

Possible orientations are illustrated in Figure 2.3, where each square represents a possible alignment point between source and target words. Considering that the gray zone in the center represents the extracted phrase pair, then the reordering orientation types are defined as follows:

- **monotone**: if there is a word alignment point at the bottom left corner, this is evidence for monotonous reordering with respect to the previous phrase (Monotone Left, ML in the figure); if there is a point at the top right corner, then the reordering is monotonous with respect to the following phrase (Monotone Right, MR).
- **swap**: if there is a word alignment point at the top left corner, this is evidence for swapping relative to the previous phrase (Swap Left, SL); if there is a point at the bottom left, then it corresponds to swapping relative to the following phrase (Swap Right, SR).
- **discontinuous**: all the other possible cases are considered as discontinuous in which there are also 2 different conditions: Discontinuous Left (DL) or Discontinuous Right (DR), respectively with respect to the previous or the following phrase.

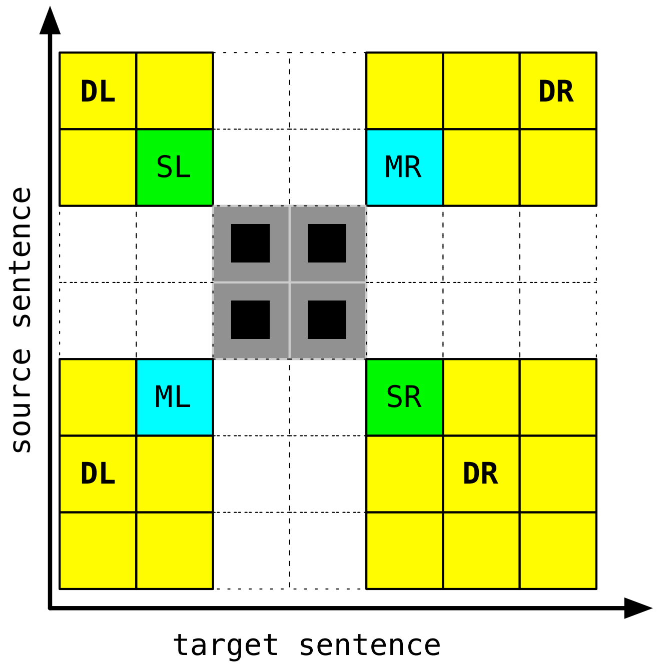


Figure 1.3 – Illustration of different orientation types for reordering models.

Typically, in state-of-the-art SMT system development, only these three reordering types will be considered. For each phrase pair, there are thus three possible reordering types on both sides, so lexicalized reordering models correspond to 6 features in the system.

1.4 Decoding and tuning

The **decoding** task in SMT consists in finding the most likely translation according to a set of previously estimated models. Given a specific input sentence, there can be many possible segmentations. Additionally, each phrase typically has several possible translations, and each phrase pair may have been seen with different orientations, leading to an exponential increase in the number of candidate translations. In fact, **?** has shown the decoding problem for simplified versions of the models discussed here to be NP-complete.

In order to reduce the computational complexity, several techniques have been proposed to prune the decoding search space, such as A* search [**?**], syntactic parsing-based decoding [**?**], greedy hill climbing decoding [**??**] and stack-based beam search decoding [**??**].

The current state-of-the-art decoder for the phrase-based SMT model is the beam-search decoder which was also used in the experiments of this thesis.

1.4.1 Scoring function

In [?], the basic phrase-based statistical machine translation decoding problem is defined as:

$$\mathbf{t}_{\text{best}} = \operatorname*{argmax}_{\mathbf{t}} \prod_{i=1}^{I} p(\bar{s}_i|\bar{t}_i) d(start_i - end_{i-1} - 1) p_{LM}(\mathbf{t}) \tag{1.12}$$

where $p(\bar{s}_i|\bar{t}_i)$ represents the probability given by the translation models, $\bar{t}_i$ is the ith target phrase in the translation and $\bar{s}_i$ its corresponding source phrase, $d(start_i - end_{i-1} - 1)$ is a distance-based reordering model, $start_i$ represents the position of the first word of $\bar{s}_i$ and end_{i-1} the position of the last word of $\bar{s}_{i-1}$, and $p_{LM}(\mathbf{t})$ represents the probability given by the language model. With this model, the translation model, language model and reordering model are combined together to determine the best attainable translation candidate. However, components in Equation (2.12) are not equally important to model translation. To quantify the relative importance of each component, **model weights** are introduced:

$$\mathbf{t}_{\text{best}} = \operatorname*{argmax}_{\mathbf{t}} \prod_{i=1}^{I} p(\bar{s}_i|\bar{t}_i)^{\lambda_p} d(start_i - end_{i-1} - 1)^{\lambda_d} p_{LM}(\mathbf{t})^{\lambda_{LM}} \tag{1.13}$$

This model can be transformed into a so-called **log-linear** form:

$$\mathbf{t}_{\text{best}} = \operatorname*{argmax}_{\mathbf{t}} \sum_{i=1}^{m} \lambda_i h_i(\mathbf{s}, \mathbf{t}, \mathbf{a}) \tag{1.14}$$

where m is the number of feature functions h_i used. The weights λ_i are trained using Minimum Error Rate Training (`MERT` [?]) or equivalent procedure during the tuning phrase (see Section 2.4.3 for details).

1.4.2 Automatic evaluation

The automatic evaluation of the quality of MT output is very important during the SMT system development phase, as it should indicate how good a system is. However it is a difficult problem. For instance, we can see in Figure 2.4 that even a short Chinese sentence may be translated into English in many different, acceptable ways by professional translators.

Developing SMT systems usually requires to perform many rounds of evaluation so as to ascertain whether a system actually improved after a change. For speed, cost and consistency reasons, such an evaluation resorts to automatic metrics, which are computed with respect to a set of possible **reference translations** produced by human translators. A large number of such automatic evaluation metrics have been proposed over the years, with BLEU (BiLingual Evaluation Understudy) [?] and TER (Translation Error Rate) [?] being two of the most influential.

The BLEU metrics compares some translation candidate with one or multiple translation references by roughly counting the matches of n-grams between the translation candidate and

这个 机场 的 安全 工作 由 以色列 方面 负责 。
Israeli officials are responsible for airport security.
Israel is in charge of the security at this airport.
The security work for this airport is the responsibility of the Israel government.
Israeli side was in charge of the security of this airport.
Israel is responsible for the airport's security.
Israel is responsible for safety work at this airport.
Israel presides over the security of the airport.
Israel took charge of the airport security.
The safety of this airport is taken charge of by Israel.
This airport's security is the responsibility of the Israeli security officials.

Figure 1.4 – Ten different human translations of the same Chinese sentence from the 2001 NIST evaluation set. (reproduced from [**?**, p. 218])

the references. It is defined as:

$$\begin{aligned} \text{BLEU} &= \text{brevity-penalty} \times \exp \sum_{n=1}^{N} \lambda_n \log \text{precision}_n \\ \text{brevity-penalty} &= \min(1, \frac{\text{output-length}}{\text{reference-length}}) \end{aligned} \tag{1.15}$$

where precision$_n$ is the modified n-gram precision, i.e., the ratio of correct n-grams of order n in relation to the total number of generated n-grams of that order; brevity-penalty is included in the metrics so as to penalize the score if the candidate translation is too short compared to the reference translation. In empirical use, the maximum order n for n-grams to be matched is typically set to 4. Moreover, the weight vector λ_n used to model the importance of different precisions are typically set to 1.

TER computes the minimum number of edits required to modify a translation candidate into exactly one of the translation references, normalized by the average length of the references. The TER score is defined as:

$$\text{TER} = \frac{\text{number of edits}}{\text{average number of reference words}} \tag{1.16}$$

Although BLEU is widely used in MT evaluation, recent work has pointed out a number of problematic aspects of such a metric. In particular, BLEU ignores the relative relevance of different words in the sentences and operates only on a very local level without considering any overall grammatical coherence. **?** showed out that BLEU is not always an appropriate metric for MT system comparison. **?** illustrated several weaknesses of BLEU metric and described experiments where BLEU was not an appropriate metric.

Besides these two metrics, other metrics are sometimes used by researchers as complementary indicators of translation performance. NIST [**?**] is a variant of the BLEU metrics, in which n-grams are no longer equally weighted, but where some informativeness of each particular n-gram is taken into account. METEOR (Metric for Evaluation of Translation with Explicit Ordering) [**??**] is designed to incorporate a stronger emphasis on recall, where the idea is to ensure that the complete meaning is captured by the output. However, it is not widely used because of its computational complexity, which involves computationally expensive word alignment, the tuning of many other parameters and the use of language-dependent linguistic resources. Finally, HTER (Human Translation Error Rate) [?] is a human-targeted variant of TER, in which the actual edit sequence that yields an acceptable closest translation is performed by a human translator. Although HTER is often regarded as one of the most informative metrics, its reliance on human intervention makes it very costly to integrate it into system development.

1.4.3 Parameter tuning

As shown in Equation (2.14), each translation candidate could be represented by a set of features. An optimization process is needed to determine the usefulness of each feature by tuning their weight λ_i in order to make the SMT system achieves the best translation results. This optimization is performed using a **development corpus**, kept separate from the original training data and any future test data. The development corpus is however supposed to be a true representation of any incoming test data to ensure that the parameters optimized on it are also optimal on the test data.

During parameter tuning, sentences in the development set are iteratively decoded with varying parameter values. The system first translates the development set using initial parameters, generates n-best lists of translation candidates, and find the optimal parameters with respect to some automatic evaluation metric, such as BLEU. Then, the system runs the decoding process again with the new parameters. Such process iterates until it converges, meaning that no more improvements on translation performance are obtained.

A number of optimization algorithms have been proposed that notably `MERT` (Minimum Error Rate Training) [?] and `KBMIRA`, a variant of the Margin Infused Relaxed Algorithm [?].

1.5 Computer-Assisted Translation (CAT)

Although SMT has made large progress in recent years, SMT systems are barely used without any human intervention in professional translation domain as they cannot deliver high-quality translations for most translation tasks. A common practice in the industry is to provide SMT output to human translators for post-editing, a strategy that has been shown to be more efficient than translation from scratch in a number of situations [??]. However, no real interaction is involved in post-editing, the human translator simply correcting and improving the system's translation output.

Interactive Machine Translation (IMT) was pioneered by projects such as TransType [?], where a SMT system assists the human translator by proposing translation completions that the translator can accept, modify or ignore. IMT was later further developed to enable more types of interaction [??] and integrating the result of the interaction to influence future choices of the system. More recently, online learning was introduced in the IMT framework [?] to improve the exploitation of the translator's feedback.

1.6 Summary

The research in machine translation has been active for more than 40 years, with very significant results achieved over the last decades. The number of approaches for MT has multiplied in these years, in which SMT has become the most studied approach. SMT approaches are also studied in the context of hybrid MT and interactive MT.

In this chapter, we have briefly presented the main processes involved in developing SMT systems, including the different probabilistic models, especially the phrase-based models, decoding algorithms and translation evaluation metrics.

However, the current state-of-the-art SMT approaches still face numerous problems. Firstly, developing a high-performance SMT system usually requires to process very large-scale parallel corpora, which is computationally expensive. Large-scale data are often used to improve the quality of word alignment and to increase the lexical coverage of the system. The current state-of-the-art SMT development toolkit, `moses`, precomputes all translation information in advance, which is time-consuming and requires powerful computing configurations in terms of processors and working memory. Secondly, the resulting systems can not be easily updated. Making use of newly available data usually requires to retrain the whole system from scratch. Additionally, as presented in Section 2.3, the statistical models used in state-of-the-art SMT systems are based on the relative frequency of the translation units in which all translation instances are equally important. The extracted probability distributions only depend on the statistics of the training data independently of the specificities of the test data, which may lead to non-appropriate translations.

In the next chapter, we will present several existing works which improve state-of-the-art SMT systems by addressing these issues.

Introduction

Our world is more and more globalized. Due to the development of the internet and transportation, access to information and communication between people become easier and faster like never before. Although the English language is now a *de facto* international language, the language barrier is still a concrete problem faced by many people in their personal communication and access to information. Many books are translated into different languages, movies are subtitled or even dubbed, and manuals of high-tech products are available in multiple languages. The availability of most written and spoken pieces into as many natural languages as possible is an undebatable objective, yet, only tiny fractions of them get actually translated, usually in small number of languages. The role that machines can play in assisting to produce more translations is thus obvious.

Machine Translation (MT), and in particular Statistical Machine Translation (SMT), has considerably matured in the past decade and is now easily available through numerous web-based translation services. Importantly, the development of such high-performance systems involves the costly processing of very large-scale data, with implications as regards economic exploitation, private access to individuals and small organizations, exploration possibilities for research purposes, not to mention the ecological cost induced. Few research works have previously tackled the issue of this computational cost. Among those, the works by **?**, **?** and **?** have empirically demonstrated that top performance can be achieved with only a fraction of the available translation examples. Random sampling of a fixed number of translation examples was in fact shown to perform competitively with systems trained on complete datasets. However, these previous works are based on the very important assumption that the required word-level alignments were previously computed. Considering that this alignment step constitutes the largest part of SMT system development time, improving how such alignements are computed is pivotal to significantly reduce the time and resources required.

An additional fact is that new bilingual data are constantly made available, which challenges standard training procedures that were optimized to accommodate *static* parallel corpora. Incorporating new data into existing SMT systems usually imposes system developers to re-train systems from scratch, which is a waste of resources and time. A limited number of previous works have addressed the issue of incremental system development and adaptation, such as [**???**], but they non-selectively align and incorporate all new available data into existing systems. Actually, a large proportion of information computed in this way will never be used to produce new translations; furthermore, non-selectively adding and using new data may be more harmful than beneficial [**?**].

It is in fact a well-established fact that translations must be adapted based on the context of the text to translate. However, state-of-the-art SMT systems only take a limited notion of context into account and essentially compute static translation probabilities once and for all from the training data. Contextual adaptation has been tackled in a variety of ways (e.g. [**?????**]), most notably by adapting the translation and language models. In most of these works, adaptation is based on a small held-out *in-domain* corpus which is used to tune the system and considered as a given, thus in effect specializing a existing, general purpose, translation system. The optimality of adaptation on unseen test data relies on the assumption that both tuning and test data observe identical probabilistic distributions, but this assumption often does not hold for real-world data. In addition, contextual adaptation is also performed once and for all and is expensive, in particular when very large quantities of training data are available. In practical terms, it is often the case that small improvements on translation quality involve a significant increase on the computational cost.

In this thesis, we propose a on-demand framework to tackle the 3 problems discussed above *jointly*, to not only enable to develop SMT systems on a per-need basis but also to enable incremental update and to adapt existing systems to each indivual input text. Unlike current state-of-the-art SMT systems which essentially pre-compute all information in advance, our framework computes all information, including word alignments, on a per-need basis. The first main contribution of this thesis is to make use of the on-the-fly model extraction approach borrowed from [**??**], and introduce a novel on-demand word alignment technique that is capable of independently aligning sentence pairs from a bilingual parallel corpus. By combining these two techniques, our framework can perform on-demand development of SMT systems from arbitrary large training data sets and for any input text. In addition, since the training data does not need to be pre-aligned, newly available data can be seamlessly integrated and used to produce subsequent translations. Besides the random sampling of translation examples strategy presented in [**??**], several contextual sampling strategies will be studied. So, the second main contribution is to integrate contextual sampling strategies into the sampling-based framework so as to adapt the computed translation information to each input text.

This manuscript is organized as follows. First, we give in Chapter 2 a brief overview of Statistical Machine Translation and some of its practical applications. In Chapter 3, we describe several previous studies addressing the main problems of the state-of-the-art approach. We then introduce in Chapter 4 our SMT system development framework that addresses all of these problems jointly. A series of incremental adaptation experiments are presented in Chapter 5 to illustrate the practical potential of our approach. Finally, several contextual sampling strategies are studied in Chapter 6. We conclude this manuscript with a summary of our main contributions and some details about the most promising continuations of our work.

Part I

Development of Statistical Machine Translation Systems

2 Overview of Statistical Machine Translation

Machine Translation (MT) is the field of computer science that studies the automatic translation from one natural language to another. It is known to be one of the very first applications of **Natural Language Processing** (NLP), while, ironically, being one of the most difficult ones, as it encompasses issues in language analysis, transfer and generation. Following the initial attempts at building MT systems, exemplified by IBM's efforts in Russian-English MT[1], a significant amount of research has been devoted to MT, with various approaches introduced and improved up to the current days.

During the 1970s, **Rule-based Machine Translation** (RBMT) was the dominent approach, with translation consisting in a series of analysis processes (morphological, syntactic and/or semantic) of the input text, followed by a process of text generation of the target text. The steps of each process are controlled by a dictionary and a grammar which are obtained through inspection by linguists, entailing a slow and time-consuming development process faced with the '*knowledge-acquisition bottleneck*'.

An alternative direction to MT was introduced to overcome these issues. **Corpus-based (or Data-driven) Machine Translation** attempts to derive the knowledge necessary to produce new translations from existing **bilingual parallel corpora**, made of translated text units, typically sentences. **Example-based Machine Translation** (EBMT) [**?**] systems, which were introduced in the early 1980s, takes sentences as basic translation units. If a sentence to translate exists exactly in the parallel corpus, its translation is extracted and reused as the system's output. Otherwise, the system searches for similar sentences in the corpus, and suitable subsequences of the sentence are iteratively replaced, modified or adapted in order to generate the new translation.

The increased availability of computational power and necessary resources progressively allowed the development of more computationally intensive approaches to MT. **Statistical Machine Translation** (SMT), introduced by **?** in **?**, is a data-driven approach characterized by its implementation of a highly developed mathematical theory. Generally, SMT systems learn

[1]For more early history of MT, see [**?**].

a **translation model** from a bilingual parallel corpus and a **language model** from a monolingual corpus. When translating, the best translation is the one that maximizes a predefined score according to the models. Additionally, the development of automatic translation **evaluation metrics** and of several free and open source SMT toolkits, have gradually facilitated the implementation and evaluation of translation systems. Furthermore, constant increases in computing power and availability of large bilingual paralllel corpora have made SMT the dominant approach in the early 2010s.

With regard to translation units, SMT can be divided into three groups: word-based, phrase-based and syntax-based SMT. Due to its simplicity and effective modeling of local contexts, the phrase-based variant is now considered a state-of-the-art choice for many language pairs. The work presented in this thesis is framed in the phrase-based framework, and the remainder of this chapter will provide a brief introduction to phrase-based SMT; other approaches to SMT are described in details in several reference works [**????**].

In this chapter, we will give a brief overview of SMT, focusing mostly on phrase-based SMT. Since the technologies of statistical machine translation have already been described in detail in many existing references, we only introduce the general principles and go into detail only for the parts that will be directly relevant to the work presented in this thesis. The content of this chapter is structured as follows. Section 2.1 and 2.2 are devoted to the general principles of SMT and word alignment. The statistical models used in SMT systems are presented in Section 2.3. The decoding process, automatic evaluation of translation quality and system tuning are described in Section 2.4. A summary of this chapter is finally given in Section 2.6.

2.1 Statistical machine translation

The base material for building an SMT system is a sentence-aligned bilingual corpus[2], where we denote the language of the given text by *source language* and the one the source text will be translated into by *target language*. Making the strong assumption that the translations can be word-aligned, the training parallel corpus is automatically analyzed to generate alignments between the words in source language and their translations in target language. From the resulting **word alignment** of the parallel corpus, a number of statistical models are estimated, including a **translation model**. A **language model** is also estimated using large quantities of text in the target language. The relative importance of all the estimated models are optimized on a development corpus before being applied to translate new source language sentences, which do not exist in the parallel corpus.

Translation generation is treated as a search problem [**?**], the goal of which is to find the most probable translation candidate $\mathbf{t} = t_1^J = t_1 \dots t_J$ for a given source language sentence $\mathbf{s} = s_1^I = s_1 \dots s_I$, where I and J represent the length of the source and target sentence, respectively. The best translation can be obtained by maximizing $P(\mathbf{t}|\mathbf{s})$ and applying the noisy channel model [**?**], defined in Equation (2.1):

[2]Note that, in such sentence-aligned bilingual corpora, which are symmetric and could be used for translations in both directions, some parts are originally written in the source language and translated into the target language, some parts are originally written in target language and translated into the source language, some parts are even written in a third language and then translated into both the source and target languages. This property have a strong influence on the SMT performance [**??**]

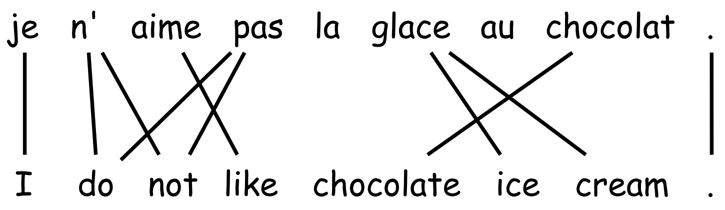

je n' aime pas la glace au chocolat .
I do not like chocolate ice cream .

Figure 2.1 – An example of word alignment between a French sentence and its English translation. (reproduced from [?, chapter 7]).

$$\begin{aligned}\mathbf{t}_{best} &= \operatorname*{argmax}_{J,t_1^J} p(t_1^J|s_1^I) \\ &= \operatorname*{argmax}_{J,t_1^J} p(s_1^I|t_1^J)p(t_1^J) \end{aligned} \tag{2.1}$$

where $p(s_1^I|t_1^J)$ denotes the translation model and $p(t_1^J)$ denotes the target language model. The translation model is in practice defined as a set of models, that can include translation, fertility, and distortion probabilities from the IBM word alignment models (see Section 2.2). They are estimated using the Expectation Maximization (EM) algorithm [?] on the bilingual parallel corpus.[3] Although the IBM models are no longer the state-of-the-art in translation modeling, they are still considered as state-of-the-art in word alignment.

2.2 Word alignment

A word alignment between parallel sentences attempts to represent the translations between words in the source sentence and words in the target sentence. An example of word alignment of a parallel sentence pair is illustrated in Figure 2.1. As these alignments are seldom explicitly represented in parallel data, word alignment models are usually learnt from *incomplete* data using the EM algorithm, an unsupervised Machine Learning approach. The translation model defined in Equation (2.1) can be written as in Equation (2.2):

$$p(s_1^I|t_1^J) = \sum_{a_1^J} p(s_1^I, a_1^J|t_1^J) \tag{2.2}$$

where a_1^J denotes the word alignments between s_1^I and t_1^J.

The IBM word alignment models [?] include five models of increasing complexity:

- **IBM Model 1** models lexical translation, in which all possible alignments are considered as equally probable. It does not model the issue of word reordering across languages.
- **IBM Model 2** adds a reordering model based on the absolute positions of aligned words in the source and target sentences.

[3]A detailed account can be found in [?].

- **IBM Model 3** adds a fertility model which introduces the possibility of one-to-many, one-to-zero and zero-to-one (by `NULL` insertion) translation.
- **IBM Model 4** adds a relative alignment model, in which the position of the translation of an input word is typically based on the position of the translation of the previous input word; and a dependency on word classes for distortion models to deal with data sparsity problems.
- **IBM Model 5** fixes the problem of Model 3 and Model 4 which allow multiple output words to be placed at the same position.

? proposed to model word alignment as a first-order Hidden Markov Model (HMM) [**?**], in which an alignment position a_j depends on the previously aligned position a_{j-1}. According to this model, the translation probability factors as:

$$P(\mathbf{t}, \mathbf{a}|\mathbf{s}) = P(J|I) \prod_{j=1}^{J} P(a_j|a_{j-1}, I) P(t_j|s_{a_j}) \tag{2.3}$$

where, $P(J|I)$ is the sentence length probability, $P(a_j|a_{j-1}, I)$ is the transition probability, and $P(t_j|s_{a_j})$ is the emission probability as defined by IBM model 1. The HMM model has attractive properties which permits numerous extensions: a comparative overview of various alignment methods is provided in [**??**].

2.3 Phrase-based probabilistic models

In contrast to word-based models, the translation unit in phrase-based models [**??**] is the *phrase*, that is, a sequence of words. An input sentence is translated phrase by phrase, offering the following advantages:

- By using phrases, a system can seamlessly handle the cases of many-to-one translations (and vise versa), where words may not be the best translation units. For example, the French word *pomme de terre* could be treated as a single unit and translated into *potato*. This characteristic is useful for translating idioms and generally phrases whose translation cannot be composed using word-level translations only.
- The use of phrases can significantly reduce lexical ambiguity during translation. For example, the English word form *press* could be translated into at least *appuie* or *appuyez* depending on the subject of the verb. Translating this word within a phrase such as *I press* eliminates the lexical ambiguity regarding the correct word form in the target language. However, such ambiguity reduction is limited, as it only works when contextual information is immediately neighboring the ambiguous word.
- Phrases can also contain cases of translation reordering. For example, in Figure 2.1, the French phrase *glace au chocolat* can be translated into English as *chocolate ice cream*, whose internal reordering is inherently captured in a bilingual phrase.

Phrase-based translation models are estimated from a word-aligned parallel bilingual corpus. Using the parallel corpus and its word alignments, **phrase pairs**, which associate a phrase in the source language and its translation in the target language, can be extracted from each sentence pair. In the current state-of-the-art approaches, a phrase pair can be extracted if and only if it is consistent with the word alignment. More formally, a phrase pair $(\bar{s}, \bar{t})$ is said to be consistent with an alignment $\mathbf{a}$ if all words in $\bar{s}$ that have alignment points in $\mathbf{a} = \{(i,j) \subset \{1 \ldots I\} \times \{1 \ldots J\}\}$ have these with words included in $\bar{t}$ and vice versa:

$$\begin{aligned} (\bar{s}, \bar{t}) \text{ consistent with } \mathbf{a} \Leftrightarrow \\ \forall s_i \in \bar{s} : (s_i, t_j) \in \mathbf{a} \Rightarrow t_j \in \bar{t} \\ \texttt{AND} \;\; \forall t_j \in \bar{t} : (s_i, t_j) \in \mathbf{a} \Rightarrow s_i \in \bar{s} \\ \texttt{AND} \;\; \exists s_i \in \bar{s}, t_j \in \bar{t} : (s_i, t_j) \in \mathbf{a} \end{aligned} \tag{2.4}$$

After the extraction of all phrase pairs present in the parallel corpus, translation probabilities between source and target phrases can be obtained by taking the relative frequency of the phrase pair given the source or target phrase:

$$p(\bar{t}|\bar{s}) = \frac{\text{count}(\bar{s}, \bar{t})}{\sum_{\bar{t}'} \text{count}(\bar{s}, \bar{t}')} \tag{2.5}$$

This translation probability is typically computed in both translation directions : $p(\bar{t}|\bar{s})$ and $p(\bar{s}|\bar{t})$.

Infrequent phrase pairs may cause problems, especially if they are collected from noisy data. For instance, if both source phrase $\bar{s}$ and the target phrase $\bar{t}$ only occur once in the training corpus **C**, then $p(\bar{t}|\bar{s}) = p(\bar{s}|\bar{t}) = 1$, which is clearly an over-estimate of how reliable this phrase pair is. To overcome this problem, phrase pairs can be decomposed into their word translations so that it becomes possible to estimate how lexically coherent they are by means of a score called **lexical weighting**. Given the word-level alignment between two phrases in a pair, the lexical translation probability of a phrase $\bar{t}$ given the phrase $\bar{s}$ can be computed by:

$$lex(\bar{t}|\bar{s}, \mathbf{a}) = \prod_{j=j_{start}}^{j_{end}} \frac{1}{|\{i|(i,j) \in \mathbf{a}\}|} \sum_{\forall (i,j) \in \mathbf{a}} w(t_j|s_i) \tag{2.6}$$

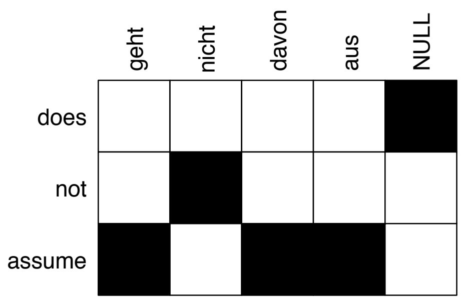
where $w(t_j|s_i)$ represents the word translation probability of the target language word t_j given the source language word s_i. If a target language word is aligned to several source language words, the average of the corresponding word translation probabilities is used. If a target language word is not aligned to any source language word, it is said to be aligned to the `NULL` word, which is also factored in as a word translation probability. This computation is illustrated in Figure 2.2. Lexical translation probabilities $w(t_j|s_i)$ can be simply estimated from the word aligned corpus using:

$$w(t|s) = \frac{\text{count}(s, t)}{\sum_{t'} \text{count}(s, t')} \tag{2.7}$$

Similarly to the phrase translation probability, the lexical weighting are also used in both translation directions: $lex(\bar{t}|\bar{s}, \mathbf{a})$ and $lex(\bar{s}|\bar{t}, \mathbf{a})$.

2.3.1 Target language model

Another essential component in a SMT system is a target **language model** (LM), whose role it is to measure how fluent and likely a sequence of words is so as to lead artificial systems to generate texts that be as much comprehensible as possible. A statistical language model provides



$$\begin{aligned} lex(\bar{t}|\bar{s}, \mathbf{a}) = & w(\text{does}|\texttt{NULL})\times \\ & w(\text{not}|\text{nicht})\times \\ & \frac{1}{3}(w(\text{assume}|\text{geht}) + w(\text{assume}|\text{davon}) + w(\text{assume}|\text{aus})) \end{aligned}$$

Figure 2.2 – Lexical weight of a phrase pair $(\bar{s}, \bar{t})$ given an alignment $\mathbf{a}$ and lexical translation probabilities. (reproduced from [**?**, p. 140])

an estimation of the likeliness of all possible word strings. Mathematically, by considering natural language as a stochastic process, a LM is formulated as a probability distribution $p(t_1^J)$ over sequences of words t_1^J in $\mathcal{V}^+$ where $\mathcal{V}$ is a finite vocabulary. For instance, a probabilistic language model should assign a much larger probability for "*she eats avocado salad*" than for any of "*she eats lawyer salad*", "*eats she salad avocado*" or "*she eat avocado salad*" because, intuitively, the first one is more likely to occur than the latter ones in English. The above examples show that a language model helps to improve MT in at least three aspects: semantics, syntax and grammaticality.

Trying to directly estimate the probability of a whole string is not tractable. In **n-gram language modeling**, the estimation of such a probability for a sequence t_1^J is usually broken up into predicting one word at a time. The probability $p(t_1^J)$ is therefore usually factorized in a left-to-right manner as:

$$p(t_1^J) = p(t_1)p(t_2|t_1)\dots p(t_J|t_1, t_2 \dots t_{J-1}) \tag{2.8}$$

The language model probability $p(t_1^J)$ is thus a product of word probabilities given a **history** of preceding words. In applications such a Machine Translation, a history is usually limited to $n-1$ words:

$$p(t_j|t_1, t_2 \dots t_{j-1}) \simeq p(t_j|t_{j-n+1} \dots t_{j-1}) \tag{2.9}$$

Considering the training data size and the computational cost, the value of n is usually small. Most commonly, **3-gram** or **4-gram** language models are used, which consider respectively the two or the three previous words as history to predict the next word. For instance, estimating the probability of a 3-gram $p(t_3|t_1, t_2)$ can be obtained by computing:

$$p(t_3|t_1, t_2) = \frac{\text{count}(t_1, t_2, t_3)}{\sum_{t_3'} \text{count}(t_1, t_2, t_3')} \tag{2.10}$$

where count(t_1, t_2, t_3) and count(t_1, t_2, t_3') represent how often the word sequences $t_1t_2t_3$ and $t_1t_2t_3'$ occur in the training corpus, respectively. This model estimation approach is also faced with the data sparseness problem. Several techniques, such as *smoothing*, *interpolation* and *back-off*, can be applied to improve the quality of the language model [**???**].

2.3.2 Reordering models

As illustrated in Figure 2.1, words or phrases from the target language may be reordered relative to their translation equivalent in the source language. In a phrase-based SMT system, this is handled by so-called **reordering models**, which can be of two main types: **distance-based** or **lexicalized**. Moreover, a recent trend has been to consider *pre-ordering* methods, where source sentences are reordered in a pre-processing step to match the target word order and then fed into the standard Phrase-Based pipeline [**????**].

Distance-based reordering models consider phrase reordering relative to the previous phrase in the source language. Let $start_i$ be the position of the first word of the source input phrase that translates to the ith target phrase, and end_i be the position of the last word of that source phrase. A reordering distance is computed as $start_i - end_{i-1} - 1$. This distance represents the number of words skipped (either forward or backward) when taking source words out of sequence. If two phrases are translated in sequence, then $start_i = end_{i-1} + 1$ and the distance is null.

The distance-based reordering model proposed for phrase-based SMT is thus only conditioned on movement distance. It is uniformly applied to all source phrases. However, some phrases are reordered more frequently than others, suggesting to consider instead a lexicalized reordering model that conditions reordering on the actual phrases being translated.

Lexicalized reordering models consider a sequence of orientations $o = (o_1 \ldots o_n)$, and the counts of how often each extracted phrase pair is found with each of the orientation types. A probability distribution p_o is estimated based on these counts using maximum likelihood:

$$p_o(o|\bar{s},\bar{t}) = \frac{\text{count}(o,\bar{s},\bar{t})}{\sum_o \text{count}(o,\bar{s},\bar{t})} \tag{2.11}$$

Possible orientations are illustrated in Figure 2.3, where each square represents a possible alignment point between source and target words. Considering that the gray zone in the center represents the extracted phrase pair, then the reordering orientation types are defined as follows:

- **monotone**: if there is a word alignment point at the bottom left corner, this is evidence for monotonous reordering with respect to the previous phrase (Monotone Left, ML in the figure); if there is a point at the top right corner, then the reordering is monotonous with respect to the following phrase (Monotone Right, MR).

- **swap**: if there is a word alignment point at the top left corner, this is evidence for swapping relative to the previous phrase (Swap Left, SL); if there is a point at the bottom left, then it corresponds to swapping relative to the following phrase (Swap Right, SR).

- **discontinuous**: all the other possible cases are considered as discontinuous in which there are also 2 different conditions: Discontinuous Left (DL) or Discontinuous Right (DR), respectively with respect to the previous or the following phrase.

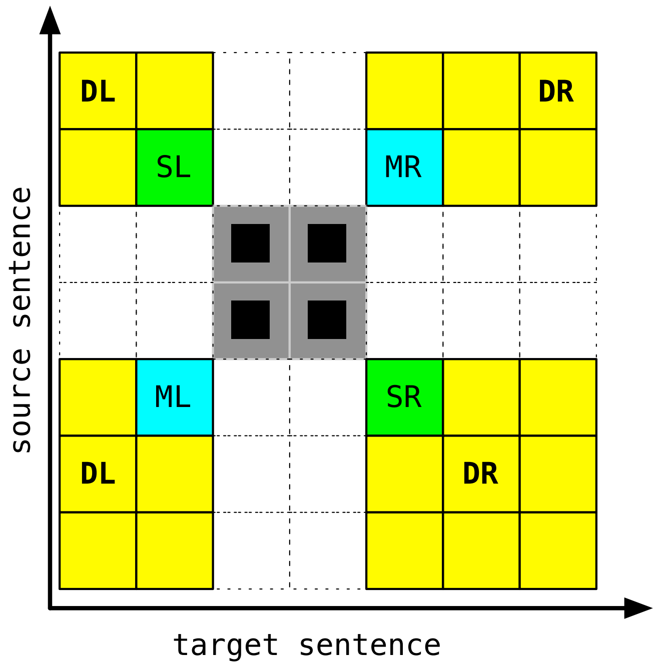


Figure 2.3 – Illustration of different orientation types for reordering models.

Typically, in state-of-the-art SMT system development, only these three reordering types will be considered. For each phrase pair, there are thus three possible reordering types on both sides, so lexicalized reordering models correspond to 6 features in the system.

2.4 Decoding and tuning

The **decoding** task in SMT consists in finding the most likely translation according to a set of previously estimated models. Given a specific input sentence, there can be many possible segmentations. Additionally, each phrase typically has several possible translations, and each phrase pair may have been seen with different orientations, leading to an exponential increase in the number of candidate translations. In fact, **?** has shown the decoding problem for simplified versions of the models discussed here to be NP-complete.

In order to reduce the computational complexity, several techniques have been proposed to prune the decoding search space, such as A* search [**?**], syntactic parsing-based decoding [**?**], greedy hill climbing decoding [**??**] and stack-based beam search decoding [**??**].

The current state-of-the-art decoder for the phrase-based SMT model is the beam-search decoder which was also used in the experiments of this thesis.

2.4.1 Scoring function

In [?], the basic phrase-based statistical machine translation decoding problem is defined as:

$$\mathbf{t}_{\text{best}} = \operatorname*{argmax}_{\mathbf{t}} \prod_{i=1}^{I} p(\bar{s}_i|\bar{t}_i) d(start_i - end_{i-1} - 1) p_{LM}(\mathbf{t}) \tag{2.12}$$

where $p(\bar{s}_i|\bar{t}_i)$ represents the probability given by the translation models, $\bar{t}_i$ is the ith target phrase in the translation and $\bar{s}_i$ its corresponding source phrase, $d(start_i - end_{i-1} - 1)$ is a distance-based reordering model, $start_i$ represents the position of the first word of $\bar{s}_i$ and end_{i-1} the position of the last word of $\bar{s}_{i-1}$, and $p_{LM}(\mathbf{t})$ represents the probability given by the language model. With this model, the translation model, language model and reordering model are combined together to determine the best attainable translation candidate. However, components in Equation (2.12) are not equally important to model translation. To quantify the relative importance of each component, **model weights** are introduced:

$$\mathbf{t}_{\text{best}} = \operatorname*{argmax}_{\mathbf{t}} \prod_{i=1}^{I} p(\bar{s}_i|\bar{t}_i)^{\lambda_p} d(start_i - end_{i-1} - 1)^{\lambda_d} p_{LM}(\mathbf{t})^{\lambda_{LM}} \tag{2.13}$$

This model can be transformed into a so-called **log-linear** form:

$$\mathbf{t}_{\text{best}} = \operatorname*{argmax}_{\mathbf{t}} \sum_{i=1}^{m} \lambda_i h_i(\mathbf{s}, \mathbf{t}, \mathbf{a}) \tag{2.14}$$

where m is the number of feature functions h_i used. The weights λ_i are trained using Minimum Error Rate Training (`MERT` [?]) or equivalent procedure during the tuning phrase (see Section 2.4.3 for details).

2.4.2 Automatic evaluation

The automatic evaluation of the quality of MT output is very important during the SMT system development phase, as it should indicate how good a system is. However it is a difficult problem. For instance, we can see in Figure 2.4 that even a short Chinese sentence may be translated into English in many different, acceptable ways by professional translators.

Developing SMT systems usually requires to perform many rounds of evaluation so as to ascertain whether a system actually improved after a change. For speed, cost and consistency reasons, such an evaluation resorts to automatic metrics, which are computed with respect to a set of possible **reference translations** produced by human translators. A large number of such automatic evaluation metrics have been proposed over the years, with BLEU (BiLingual Evaluation Understudy) [?] and TER (Translation Error Rate) [?] being two of the most influential.

The BLEU metrics compares some translation candidate with one or multiple translation references by roughly counting the matches of n-grams between the translation candidate and

这个 机场 的 安全 工作 由 以色列 方面 负责 。
Israeli officials are responsible for airport security.
Israel is in charge of the security at this airport.
The security work for this airport is the responsibility of the Israel government.
Israeli side was in charge of the security of this airport.
Israel is responsible for the airport's security.
Israel is responsible for safety work at this airport.
Israel presides over the security of the airport.
Israel took charge of the airport security.
The safety of this airport is taken charge of by Israel.
This airport's security is the responsibility of the Israeli security officials.

Figure 2.4 – Ten different human translations of the same Chinese sentence from the 2001 NIST evaluation set. (reproduced from [?, p. 218])

the references. It is defined as:

$$\begin{aligned} \text{BLEU} &= \text{brevity-penalty} \times \exp \sum_{n=1}^{N} \lambda_n \log \text{precision}_n \\ \text{brevity-penalty} &= \min(1, \frac{\text{output-length}}{\text{reference-length}}) \end{aligned} \tag{2.15}$$

where precision_n is the modified n-gram precision, i.e., the ratio of correct n-grams of order n in relation to the total number of generated n-grams of that order; brevity-penalty is included in the metrics so as to penalize the score if the candidate translation is too short compared to the reference translation. In empirical use, the maximum order n for n-grams to be matched is typically set to 4. Moreover, the weight vector λ_n used to model the importance of different precisions are typically set to 1.

TER computes the minimum number of edits required to modify a translation candidate into exactly one of the translation references, normalized by the average length of the references. The TER score is defined as:

$$\text{TER} = \frac{\text{number of edits}}{\text{average number of reference words}} \tag{2.16}$$

Although BLEU is widely used in MT evaluation, recent work has pointed out a number of problematic aspects of such a metric. In particular, BLEU ignores the relative relevance of different words in the sentences and operates only on a very local level without considering any overall grammatical coherence. **?** showed out that BLEU is not always an appropriate metric for MT system comparison. **?** illustrated several weaknesses of BLEU metric and described experiments where BLEU was not an appropriate metric.

Besides these two metrics, other metrics are sometimes used by researchers as complementary indicators of translation performance. NIST [**?**] is a variant of the BLEU metrics, in which n-grams are no longer equally weighted, but where some informativeness of each particular n-gram is taken into account. METEOR (Metric for Evaluation of Translation with Explicit Ordering) [**??**] is designed to incorporate a stronger emphasis on recall, where the idea is to ensure that the complete meaning is captured by the output. However, it is not widely used because of its computational complexity, which involves computationally expensive word alignment, the tuning of many other parameters and the use of language-dependent linguistic resources. Finally, HTER (Human Translation Error Rate) [?] is a human-targeted variant of TER, in which the actual edit sequence that yields an acceptable closest translation is performed by a human translator. Although HTER is often regarded as one of the most informative metrics, its reliance on human intervention makes it very costly to integrate it into system development.

2.4.3 Parameter tuning

As shown in Equation (2.14), each translation candidate could be represented by a set of features. An optimization process is needed to determine the usefulness of each feature by tuning their weight λ_i in order to make the SMT system achieves the best translation results. This optimization is performed using a **development corpus**, kept separate from the original training data and any future test data. The development corpus is however supposed to be a true representation of any incoming test data to ensure that the parameters optimized on it are also optimal on the test data.

During parameter tuning, sentences in the development set are iteratively decoded with varying parameter values. The system first translates the development set using initial parameters, generates n-best lists of translation candidates, and find the optimal parameters with respect to some automatic evaluation metric, such as BLEU. Then, the system runs the decoding process again with the new parameters. Such process iterates until it converges, meaning that no more improvements on translation performance are obtained.

A number of optimization algorithms have been proposed that notably `MERT` (Minimum Error Rate Training) [?] and `KBMIRA`, a variant of the Margin Infused Relaxed Algorithm [?].

2.5 Computer-Assisted Translation (CAT)

Although SMT has made large progress in recent years, SMT systems are barely used without any human intervention in professional translation domain as they cannot deliver high-quality translations for most translation tasks. A common practice in the industry is to provide SMT output to human translators for post-editing, a strategy that has been shown to be more efficient than translation from scratch in a number of situations [??]. However, no real interaction is involved in post-editing, the human translator simply correcting and improving the system's translation output.

Interactive Machine Translation (IMT) was pioneered by projects such as TransType [?], where a SMT system assists the human translator by proposing translation completions that the translator can accept, modify or ignore. IMT was later further developed to enable more types of interaction [??] and integrating the result of the interaction to influence future choices of the system. More recently, online learning was introduced in the IMT framework [?] to improve the exploitation of the translator's feedback.

2.6 Summary

The research in machine translation has been active for more than 40 years, with very significant results achieved over the last decades. The number of approaches for MT has multiplied in these years, in which SMT has become the most studied approach. SMT approaches are also studied in the context of hybrid MT and interactive MT.

In this chapter, we have briefly presented the main processes involved in developing SMT systems, including the different probabilistic models, especially the phrase-based models, decoding algorithms and translation evaluation metrics.

However, the current state-of-the-art SMT approaches still face numerous problems. Firstly, developing a high-performance SMT system usually requires to process very large-scale parallel corpora, which is computationally expensive. Large-scale data are often used to improve the quality of word alignment and to increase the lexical coverage of the system. The current state-of-the-art SMT development toolkit, `moses`, precomputes all translation information in advance, which is time-consuming and requires powerful computing configurations in terms of processors and working memory. Secondly, the resulting systems can not be easily updated. Making use of newly available data usually requires to retrain the whole system from scratch. Additionally, as presented in Section 2.3, the statistical models used in state-of-the-art SMT systems are based on the relative frequency of the translation units in which all translation instances are equally important. The extracted probability distributions only depend on the statistics of the training data independently of the specificities of the test data, which may lead to non-appropriate translations.

In the next chapter, we will present several existing works which improve state-of-the-art SMT systems by addressing these issues.

3 Improving SMT System Development

Statistical Machine Translation has considerably matured in the past decade and is nowadays the preferred choice in most practical machine-assisted translation scenarios. However, some important issues remain to be solved. Among them, of particular importance to improve the development, usability and performance of systems, are the abilities to use very large corpora, to efficiently incorporate new data efficiently, and to identify which parts of the training data are more appropriate for a given translation task. Although each problem has been the subject of numerous independent research works, this thesis will target a situation where they can be tackled jointly.

Firstly, developing SMT systems is **computationally expensive**. Large to huge sets of parallel texts of the source-target language pair must be repeatedly analyzed and processed to train various types of statistical models. A conventional wisdom is that the larger the training corpus, the more accurate the models can be, and the better the systems will be. However, building systems using large data sets requires a significant preprocessing time before any translation can be produced. In a state-of-the-art SMT system, such as `moses`, all models need to estimated in advance before any text can actually be translated. Processing all the available training data is very time-consuming and requires powerful computing configurations in terms of processors and working memory. Storing the resulting models also requires large disk capacities, especially when keeping information for long phrases. Furthermore, handling such resources at decoding time also necessitates powerful hardware configurations, including large working memory capacities to effectively explore the combinatorially large search space of SMT.

Secondly, once built from a given parallel corpus, a standard SMT system is **static**. However, new bilingual data are constantly made available by daily proceedings of international organizations, professional translation agencies using computer-assisted translation and even the web-based machine translation services[1]. Unfortunately, efficiently adding new data into existing systems is difficult, and the typical solution consists in retraining new systems from scratch, a very costly enterprise for large data sets.

[1]The *Google Translate* now allows users to modify the generated translations and send them back to the MT servers in the hope that these feedback can help to improve their MT systems.

Lastly, state-of-the-art SMT systems do **not make any distinction** on training data. Given the large variety of sources and quality of the large sets of parallel bilingual data used, discriminating between useful and harmful data is required. This is however a very complex problem, because little information may be available about the training data on the one hand, and the input text on the other hand, and also because finding relevant training examples inherently requires deep **context modeling** capacities.

This chapter discusses the above problems in some details and presents previous works in these areas. More specifically, Section 3.1 discusses the efficient use of larger corpora, Section 3.2 the issues related to incremental system training, and Section 3.3 approaches to select training data for a specific input text.

3.1 Scaling to larger corpora

Typical SMT systems pre-compute all the necessary information in advance once the (initial) parallel data are available. The size of the resulting translation models quickly grows with the size of the parallel data used, which outstrips improvements in computing power and certainly hinders research on many types of new models and the experimental exploration of their variants.

? and **?** presented a solution for developing SMT systems borrowing from EBMT. The key idea is that translation rules and models are on-the-fly computed only as needed for each particular input text. Given an input text $\mathbf{d}$ to translate, the set of all potentially useful phrases, denoted $\mathbf{\Sigma}[\mathbf{d}]$, is first extracted. For each extracted phrase $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$, its occurrences $\mathbf{C}[\bar{s}]$ in the training corpus $\mathbf{C}$ are then located. Using a pre-existing word alignment $\mathbf{A}$, the translations of a source phrase are extracted from its examples and used to compute its translation model parameters $\boldsymbol{\theta}_{\bar{s}}$. This process is repeated for all source phrases $\bar{s}$ in $\mathbf{\Sigma}[\mathbf{d}]$, and a translation table is produced and subsequently used by a decoder to translate the input document. More formally, this procedure is sketched in Algorithm 1.

Algorithm 1 On-the-fly construction of translation models

```
parallel train corpus $\mathbf{C}$ and its word alignment $\mathbf{A}$
given an input text $\mathbf{d}$
compute $\mathbf{\Sigma}[\mathbf{d}]$
for all $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$ do
    $\mathbf{C}[\bar{s}]$ = find($\bar{s}$, $\mathbf{C}$)        // Find occurrence
    estimate($\boldsymbol{\theta}_{\bar{s}}, \mathbf{C}[\bar{s}], \mathbf{A}$)    // Estimation
end for
decode as usual using extracted models
```

This approach significantly reduces the computational complexity since it only estimates translation model parameters for the phrases that occur in a specific input text. This technique furthermore allows system to extract arbitrary long phrases, while in state-of-the-art phrase-based SMT systems, phrase length is limited to a given value (typically, 7 tokens). In fact, it has been shown under some experimental conditions that extracting longer phrases does not yield much improvement, while leading to a linear increase in the phrase translation table size with the maximum length limit [**?**].

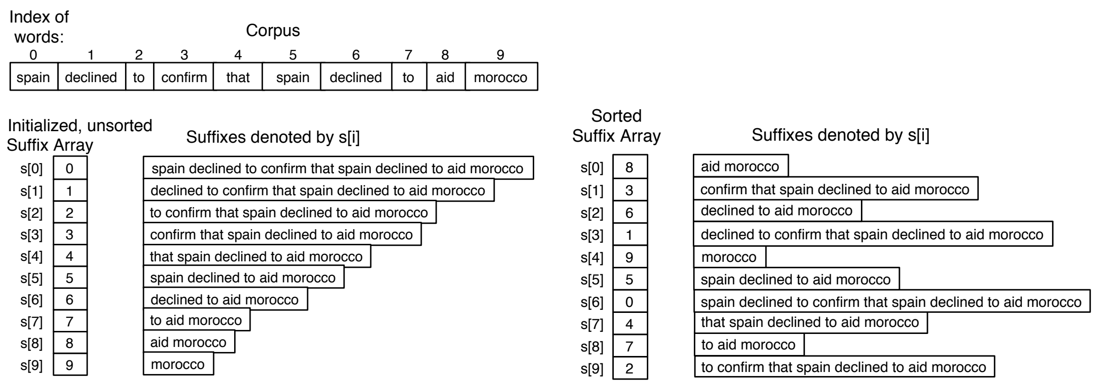

Index of words:
Corpus
0 1 2 3 4 5 6 7 8 9
spain declined to confirm that spain declined to aid morocco
Initialized, unsorted Suffix Array
Suffixes denoted by s[i]
s[0] 0 spain declined to confirm that spain declined to aid morocco
s[1] 1 declined to confirm that spain declined to aid morocco
s[2] 2 to confirm that spain declined to aid morocco
s[3] 3 confirm that spain declined to aid morocco
s[4] 4 that spain declined to aid morocco
s[5] 5 spain declined to aid morocco
s[6] 6 declined to aid morocco
s[7] 7 to aid morocco
s[8] 8 aid morocco
s[9] 9 morocco
Sorted Suffix Array
Suffixes denoted by s[i]
s[0] 8 aid morocco
s[1] 3 confirm that spain declined to aid morocco
s[2] 6 declined to aid morocco
s[3] 1 declined to confirm that spain declined to aid morocco
s[4] 9 morocco
s[5] 5 spain declined to aid morocco
s[6] 0 spain declined to confirm that spain declined to aid morocco
s[7] 4 that spain declined to aid morocco
s[8] 7 to aid morocco
s[9] 2 to confirm that spain declined to aid morocco

Figure 3.1 – Illustration of suffix array for a very small corpus (reproduced from [**?**]).

Although this approach reduces the computational complexity by estimating translation model on a *per-need* basis, it still faces some issues:

- The complexity of the naive algorithm for finding all the occurrences of a source phrase $\bar{s}$ in a training corpus **C** is linear in the length of the text, $\mathcal{O}(|\mathbf{C}|)$. This is too slow for large corpora, especially if repeated for all phrases.
- The complexity of extracting target phrases is linear in the number of source phrase occurrences. For those frequent source phrases, this is very expensive.

? and **?** resorted to two methods to solve these two problems. As for the issue of efficiently searching all occurrences of a source phrase, they used an index data structure called a **suffix array** [**?**], whose size is $4|\mathbf{C}|$ bytes and which enables the lookup of any m-length phrase of **C** in $\mathcal{O}(m + \log|\mathbf{C}|)$. Figure 3.1 illustrates how a suffix array is initialized for a corpus consisting of a single sentence, and the final state of the suffix array, which is a list of word indices in the corpus sorted by lexicographical order of the suffixes.

As for the issue of the possibly large number of examples, they extracted target phrases from restricted samples, containing a bounded number of instances. In these works, sampling is performed randomly over the complete training data so as to avoid any bias. Experimental results by these authors were competitive with the conventional estimation procedure. Importantly, sampling enables the usage of corpora that are an order of magnitude larger, paving the way to **very large-scale MT**. It should however be noted that the effect of sampling is mainly to reduce the number of output phrase pairs, but that no attempt at selecting contextually appropriate samples was proposed.

3.2 Incremental system training

Entities such as the European Union, the United Nations and other multinational organizations need to translate most of the documentation they produce. These documents are translated with high quality standards by involving human translators. Professional agencies also constantly produce multi-lingual materials from scratch or with the help of CAT. Such data could not be easily integrated in SMT systems that were developed earlier.

Integrating new data into existing SMT systems can not only increase the n-gram coverage, notably providing translations for previously unseen phrases, but it can also improve the probability distribution estimation of known phrases. Additionally, the study in [**?**] shows that systems trained on data drawn from periods that are chronologically closer to the test data usually achieve better performance, which provides further support to attempting to incorporate newly available data.

Word alignments of the new corpora have to be computed for doing so. A conventional way for this is to re-run the word alignment process over the concatenated corpus. This would be very time-consuming, and even more so if this process is to be repeated frequently. Once the word alignment for the additional data is available, the integration of new collected translation statistics into the existing translation models is also an issue. Hence, two questions arise:

1. How to update the word alignment model using additional data ?
2. How to integrate the translation statistics collected from added data into the systems ?

In order to facilitate the presentation, we call the original training data the **old data**, the new available data the **additional data**, and the concatenation of old and additional data the **new data**.

3.2.1 Incremental alignment

Word alignment estimation is the most computationally expensive process in SMT system development. The current state-of-the-art word alignment tool `giza++`, implementing a batch training regime, is based on IBM word alignment models. A conventional setting is to run a number of iterations of each IBM 1, HMM, IBM 3 and IBM 4 models in order. The output of previous iteration is used to initialize the next one. This process repeatedly analyzes the whole parallel corpus to collect statistics and train IBM alignment models. IBM model 1 and model 2 can be learned efficiently in polynomial time, while model 3 and model 4 are computationally much more expensive and resort to *hill-climbing* techniques. The whole process is time-consuming, especially when the parallel corpus is huge, so that re-training the complete alignment when a proportionally low number of new sentences are added is a waste of resource. Significant computational resources may be saved if word alignments are only computed on the additional data. However, training word alignment models on small amounts of data results in poor models. Hence, generating word alignments for additional data is an important issue for the integration of new data.

Some previous works, including e.g. [**?**], resorted to a sub-optimal **forced alignment** approach. Here, the alignment model trained on old data is used to align the additional data. This approach is sub-optimal because the statistics of the additional data are not collected and thus not used to improve either its word alignment or the word alignment of the old data. Forced alignment is usually used when only a small proportion of additional data is added, where such small quantities of additional data do not have a significant impact on the word alignment quality. But in order to achieve better performance, system developers typically choose to re-train the whole alignment models from scratch on the new data.

An alternative way to reduce the processing time of this step is to resort to faster alignment methods. For instance, **?** introduced a simple log-linear re-parameterization of IBM Model 2 that overcomes problems arising from strong assumptions in Model 1 and over-parameterization in Model 2. The reported results indicate a speed-up of up to ten times compared to `giza++`. However the new data still requires full alignment.

An alternative solution to this problem is the **stream-based translation models** proposed in [**?**]. This work introduces an adaptive training regime using an online variant of the Expectation Maximization algorithm [**??**] that is capable of incrementally aligning new parallel sentences without incurring the burdens of full retraining. Additional data are split into mini-batches and incrementally added into the old data. A step-wise online EM algorithm gathers statistics from additional data and interpolate the results with the existing models so as to produce word alignment for the additional data. The approach presented in [**?**] also takes advantage of existing models that were previously trained on old data to initialize the EM training, which is only run on the additional data.

3.2.2 Model integration

Once the word alignment of additional data is available, the next problem is to determine how to use them for producing new translations. This problem has been widely studied in the context of **domain**[2] **adaptation**, where it is usually assumed that there exists an in-domain corpus and an out-of-domain corpus whose information should be exploited jointly. The most straightforward technique is again to concatenate the in-domain and out-of-domain data prior to retraining, an approach that is not appropriate in many situations. Another simple approach is to resort to on-the-fly construction of translation models for each particular input (cf. Section 3.1).

? studied the **linear model combination**, where models are trained independently on each sub-corpus and linearly combined:

$$p(x|h) = \sum_c \lambda_c p_c(x|h) \tag{3.1}$$

where $p(x|h)$ is either a language or translation model, and $p_c(x|h)$ is a model trained on sub-corpus c, and λ_c is the corresponding weight. The weights of combination are set and optimized based on several similarity metrics between each component and the in-domain data (for example, the development set of the system).

Instead of using a linear combination of translation models, **?** proposed a phrase-table **fill-up** method, where the small in-domain phrase table is preserved and additional phrase pairs are selected from the out-of-domain phrase table and used to fill up the in-domain phrase table in order to improve model coverage.

? proposed a technique for **merging translation models** to allow rapid retraining of the translation models with incremental data. A phrase table is built for the additional data, and is then merged with the phrase table of the system to approximate the phrase table that would be obtained from full retraining on the complete training data. Small lexical tables are first built from the additional data. Then the baseline lexical tables are scanned for shared entries and the corresponding probabilities are updated using:

$$lex_{new}(t|s) = lex_{old}(t|s) \times \frac{\text{count}_{old}(s)}{\text{count}_{old}(s)+\text{count}_{add}(s)} + lex_{add}(t|s) \times \frac{\text{count}_{add}(s)}{\text{count}_{old}(s)+\text{count}_{add}(s)} \tag{3.2}$$

where, lex_{old}, count$_{old}$, lex_{add} and count$_{add}$ are the baseline (old) lexical probability trained on old data, old word count, lexical probability trained on additional data and word count on additional data, respectively. s and t represent the specific source and target words in context.

[2]Here, the notion of “domain” is meant to refer to differences in genre, style or register.

This update is applied for both directions of the lexical model ($lex(t|s)$ and $lex(t|s)$). Entries which are not shared between the old model and the additional model are simply added to the new merged lexical table. Equation (3.2) approximates the lexical probabilities which would result from full retraining. Once the lexical tables have been updated, the lexical weighting features of the phrase table can be updated. A similar approach is applied for the translation models in both directions:

$$p_{new}(\bar{t}|\bar{s}) = p_{old}(\bar{t}|\bar{s}) \times \frac{\text{count}_{old}(\bar{s})}{\text{count}_{old}(\bar{s})+\text{count}_{add}(\bar{s})} + p_{add}(\bar{t}|\bar{s}) \times \frac{\text{count}_{add}(\bar{s})}{\text{count}_{old}(\bar{s})+\text{count}_{add}(\bar{s})} \tag{3.3}$$

where, p_{old}, count$_{old}$, p_{add} and count$_{add}$ represent the baseline (old) phrase translation probability, old phrase count, additional phrase translation probability and additional phrase count, respectively. $\bar{s}$ and $\bar{t}$ represent the specific source and target phrases in context. Entries which are not shared are simply copied to the new, merged phrase table for both direction ($p_{new}(\bar{t}|\bar{s})$ and $p_{new}(\bar{s}|\bar{t})$).

Another approach is to build **mixture models**. As presented in Chapter 2, the SMT model can be transformed into a log-linear form (Equation 2.14), allowing easy inclusion of additional features. **?** used this approach to integrate an additional language model into the system. They trained this model on additional data, and then added it to their system as an additional feature whose weight was tuned together with the other feature weights of the system. They also extracted a translation table from additional data, which is combined with the translation table of the baseline system using multiple **alternative decoding paths** [**?**]. Two (or more) decoding paths can be used, as shown in Equation 3.4, for translation candidates found in the translation table derived respectively from the in-domain data or from the out-of-domain data:

$$\mathbf{t}_{\text{best}} = \underset{\mathbf{t}}{\text{argmax}} \sum_{i=1}^{m} \lambda_i h_i(\mathbf{s}, \mathbf{t}) + \sum_{j=1}^{n} \lambda_j h_j(\mathbf{s}, \mathbf{t}) \tag{3.4}$$

3.3 Data weighting schemes

As previously noted in Sections 3.1 and 3.2, access to as large quantities of training data as possible is desirable for developing SMT systems. On the one hand, using very large-scale corpora can increase the n-gram coverage. But, on the other hand, such corpora contain more counts, which help improve not only the accuracy of the alignment model but also the accuracy of the translation model. This is however theoretically true if all data are homogeneously related to the input data, but this is not the case in practice for the vast majority of SMT systems. For instance, **?** have shown that non-selectively including all data into an existing system could be more harmful than beneficial, which emphasizes the importance of evaluating the relevancy of training data with respect to the input test data. In this section, we review a variety of approaches that have been proposed to discriminate training data at various levels depending on an input text.

3.3.1 Context-dependent models

In log-linear phrase-based SMT, the translation model $P(\bar{t}_1^J|\bar{s}_1^I)$ is modeled by the log-linear combination of a set of translation features and a target language model. These translation features model the translation relation between source and target translation units. The target language model can be seen as a way to exploit *target similarity* (between the translation and other sentences in the target language), which may constrain lexical choices in the generated translation candidates. Although the use of phrases does capture some local dependency between source words, the dependencies between disjoint source phrases and words is not modeled satisfactorily. Oracle studies such as that of **?** reveal that SMT systems can produce very high quality translation candidates, demonstrating that such systems have access to very good translation phrase pairs but that they fail to make appropriate disambiguation at decoding time. This provides further support to an improved modeling of context in SMT systems.

Inspired by **Word Sense Disambiguation** (WSD), the task of determining the correct meaning or sense of a word in context, several works attempted to integrate similar techniques into SMT systems to improve the translation quality. For instance, **?** embedded state-of-the-art WSD modules into statistical MT systems. The key idea is to integrate some information about the sentence context into the phrase translation probabilistic models so as to bias the lexical choice decisions. A rich set of context features is defined, including bag-of-word context, local collocations, position-sensitive local Part-of-Speech (POS) tags and basic dependency features. Unlike the typical WSD task, the basic unit to disambiguate is not limited to single words and is extended to arbitrary, contiguous phrases. Likewise, the sense candidates are no longer built manually but automatically extracted from parallel corpora and correspond to each individual extracted translation. The trained WSD module provided a context-dependent probability for each entries in the translation table, which is then used as an additional feature in the system. This approach requires a "unique token" strategy[3], which greatly increases the size of phrase tables. However, only modest improvements were obtained by these means.

Similarly, **?**, also inspired by the WSD task, dealt with the phrase-based translation problem as a classification problem. They built classifiers for each source language phrase using local features of the sentences in which a phrase appears. As contextual features, they used the words, POS tags, lemmas and base phrase chunking IOB (Inside, Outside, Begin) labels.

Such approaches require to construct a WSD classifier for each source phrase (including single-word and multi-word phrases), a heavy burden on system development. Instead of relying on standard WSD techniques, **?** proposed to directly add context-informed features into the log-linear phrase-based SMT model. Similarly to previously mentioned works, the studied context features include neighboring words and POS tags. But because of data sparseness, they opted for a decision tree classifier that implicitly smooth relative frequency estimation. The experiments were performed on small amounts of data, and the reported results showed that integrating source context modeling into phrase-based SMT systems can positively influence the weighting and selection of target phrases, and thus improve the translation quality. **?** extended this approach to use supertags [**?**] as context features. The experimental results also showed that both neighboring words and POS tags can improve translation quality significantly. More directly, **?** proposed to incorporate context information into the relative frequency estimation (cf. Equation 2.5), as:

$$p(\bar{t}|\bar{s}, \bar{s}_{context}) = \frac{\text{count}(\bar{t}, \bar{s}, \bar{s}_{context})}{\sum_{\bar{t}'} \text{count}(\bar{t}, \bar{s}', \bar{s}_{context})} \tag{3.5}$$

Since adding conditioning variables leads to increased data sparseness, conditional features for different, separate types of context were computed. In addition to the lexical context of

[3]This means that each individual phrase occurrence requires its own set of entries in the phrase table.

words appearing in the immediate context, shallow and deep syntactic features as well as some positional features of the sentential context were also extracted. Additional contextual features were directly integrated into the log-linear model (Equation 2.14) and MERT [**?**] was used to obtain the interpolation weights λ_i.

? also proposed a feature function using the vector-space model to capture the sub-sentential source context information. The key idea behind their approach is that if the training sentence pairs from which the phrases in the translation table were extracted are memorized, a vector-space model can be used to identify those phrases that were extracted from sentences that are similar to the current input sentence to be translated. This way, priority can be given to those phrases that were extracted from a similar context to the one being translated. In their work, a phrase table is extracted for each input sentence, and a context-informed feature is estimated for each phrase pair in the phrase table. Although the reported result significantly outperformed a state-of-the-art baseline system, the high computational cost indeed prohibits running these experiments on large corpora.

Overall, the WSD-inspired approaches require to train classifiers for each source phrase in the input text or to gather context information for each unique token source phrase, and to have access to very large training corpora in order to avoid the data sparseness problem for feature estimation. An alternative approach is to evaluate the data relevancy to the input test data in a binary fashion, and estimate translation models only on data that is found to be relevant for the test data.

3.3.2 Data selection

Large amounts of data are nowadays available to train SMT systems. However, the most common situation is one where a large portion of the available data is obtained from domains that differ from the test domain of the system. Constructing a huge SMT system using all available data is in fact not only computational expensive, but also possibly more harmful than beneficial [**?**]. One possibility is to select only the portion of data that is most similar to the test data, and to add only the selected subset during training.

The data selection approach was first used for language modeling [**???**]. Monolingual sentences in out-of-domain corpora are scored and sorted by their *perplexity* score according to a language model trained on the in-domain corpus, and then only a small portion is retained and used to train a new language model. The perplexity (PP) of some string s with empirical n-gram distribution p given a language model q is:

$$PP = 2^{\sum_x p(x) \log q(x)} = 2^{H(p,q)} \tag{3.6}$$

where $H(p,q)$ is the *cross-entropy* between p and q. We simplify this notion to just $H_I(s)$, meaning the cross-entropy of string s according to a given language model LM_I with distribution q.

Rather than using the cross-entropy, **?** proposed to use the *cross-entropy difference* (CED) as the ranking function to select useful general-domain data for language modeling. In this work, the authors constructed not only a language model LM_I on the in-domain data, but also a language model LM_G on the general-domain data. The cross-entropy difference (CED) of some string s is:

$$CED = H_I(s) - H_G(s) \tag{3.7}$$

Again, the lowest-scoring portion of the general-domain data is retained to build a new language model. This criterion tends to select sentences that are *similar* to the in-domain data and at the same time *dissimilar* to the average of the general-domain data.

? further extended this approach to select bilingual sentence pairs for translation model estimation. Sentence pairs are now selected based on the *bilingual cross-entropy difference* (BCED), where the cross-entropy difference is applied on both the source language side and the target language side. It is defined as:

$$\begin{aligned} BCED &= CED^{src} + CED^{tgt} && (3.8)\\ &= [H_I^{src}(s) - H_G^{src}(s)] + [H_I^{tgt}(s) - H_G^{tgt}(s)] && (3.9) \end{aligned}$$

Again, lower scores are presumed to be better. This criterion takes into account the target side information and is well adapted to the bilingual nature of the translation model construction task.

The above mentioned works mainly discuss metrics for data selection but barely address the issue of how much data should be selected, most works relying on pre-defined empirical values. In [**?**], the quantity of additional data to use for training translation models is selected automatically. To do this, all sentences in the additional corpus are ranked using their perplexity with respect to an in-domain language model. Additional data are then split into small batches using predefined perplexity ranks. At each iteration, one batch of additional data is added into the system. If its inclusion actually improves the translation quality of the system on some development set, then it is retained, otherwise, it is not included into the training data of the system. By using such a quality-based criterion, systems can incorporate only the useful parts of additional data and discard the others. However, this approach demands to incrementally update the translation models and to repeatedly decode the development set, which is quite expensive.

In several other works [**???**], data selection is performed to select sentence pairs from additional corpora that specifically allows to decrease the *out-of-vocabulary* (OOV) rate of the system. Indeed, OOV words have been shown to represent one of the major reasons for the degradation in performance when applying SMT systems on new domains [**?**]. **?** presented a dictionary mining technique that mines translations for OOV words from comparable corpora in a domain adaptation task. **?** proposed to select sentences from additional parallel corpora based on an *infrequent* n-gram criterion in the in-domain data rather than simply OOV words. There, a n-gram is considered infrequent when it appears less often than a given threshold in the training corpus. Each sentence in the pool of additional data is scored based on the number and the frequency of infrequent n-grams that it contains.

Information Retrieval (IR) methods are also widely used for data selection purposes [**??**], where sentences are ranked by their similarity to the input text and the highest-scoring portion is selected to improve the SMT system. The most used similarity metric is the standard TF-IDF (Term Frequency and Inverse Document Frequency) measure (see e.g. [**?**]). Each document is represented as a vector $(w_{i1} \dots w_{ij} \dots w_{in})$, where n is the size of the vocabulary. Each entry w_{ij} in this vector is computed as:

$$w_{ij} = tf_{ij} \times \log(idf_j) \tag{3.10}$$

where, tf_{ij} is the term frequency (TF) of the j-th word in the vocabulary in the document, i.e. its number of occurrences; idf_j is the inverse document frequency (IDF) of j-th term, defined as:

$$idf_j = \frac{\#\text{documents}}{\#\text{documents containing } j\text{-th term}} \tag{3.11}$$

The similarity between two documents is then defined as the similarity of the two representing vectors. **?** used the TF-IDF metric to automatically find the top n similar sentences for each sentence in the test set. The selected sentences form an adapted training corpus, which is then used to estimate translation models.

The data selection approaches enable to select the data that are most similar to the test data. However, such a binary selection discards a portion of data which may also be beneficial to translation quality. A more flexible approach, *instance weighting*, does not perform this binary selection but estimates the importance of each training example as a real value in $[0, 1]$.

3.3.3 Instance weighting

Unlike data selection, which involves the binary *hard* decision (including or discarding) for sub-corpora, instance weighting approaches make *soft* decisions by assigning a weight to each unit. The most relevant units get relatively higher weights, and the least relevant parts get lower weights, possibly a null weight [4]. The empirical phrase counts are modified using these weights and the translation feature scores are modified accordingly. Instance weighting has been applied on different unit types at different levels of granularity: sub-corpus, sentence pairs and phrase pairs.

? incorporated out-of-domain corpora using a weighted combination. Each sub-corpus is associated with a weight used to combine different sub-corpora in two ways:

- Linear interpolation: $p(\bar{t}|\bar{s}; \lambda) = \sum_{i=1}^{n} \lambda_i p_i(\bar{t}|\bar{s})$, where n is the number of sub-corpora, $p_i(\bar{t}|\bar{s})$ the translation model trained on each sub-corpus i, and λ_i the interpolation weight of each model i.
- Weighted count: $p(\bar{t}|\bar{s}; \lambda) = \frac{\sum_i^n \lambda_i c_i(\bar{s},\bar{t})}{\sum_i^n \sum_{\bar{t}'} \lambda_i c_i(\bar{s},\bar{t}')}$, where c denotes the count of an observation, the observation in each sub-corpus i being weighted by λ_i. 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.

The weights of sub-corpora are optimized to minimize the perplexity (or cross-entropy) of the translation models on the development data set:

$$\hat{\lambda} = \underset{\lambda}{\operatorname{argmin}} - \sum_{\bar{s},\bar{t}} \tilde{p}(\bar{s},\bar{t}) \log_2 p(\bar{t}|\bar{s},\lambda) \tag{3.12}$$

Using this approach, the additional data is added into the system to mitigate the data sparseness problem and at the same time its effect is limited by its weight so as to avoid exacerbating the ambiguity problem.

? performed instance weighting at the sentence level. The TF-IDF metric is used to select the top n similar sentences for each sentence in the test set and to build an adapted training

[4]Hence, data selection can be seen as a special case of instance weighting in which the assigned weight could be either 0 or 1.

corpus. The selected sub-corpus is then combined with the training corpus so as to redistribute the weight of each sentence pair in the training corpus. The work by **?** also assigned a weight to each sentence in the training corpus using a discriminative objective function. Each sentence in the corpus is represented by a vector of binary features which contains the collection and genre identifier of the sentence. The feature vectors are then mapped to a scalar weight in $(0, 1)$ by a *perceptron mapping function* whose parameters are optimized on a development set. In this way, the negative effects of low quality training sentences can be limited, and the translation model can be adapted to the domain of interest.

? extended the approach of **?** to the smaller granularity of phrase pairs. Additionally, the proposed approach no longer relies on a division of the corpus into manually-assigned portions, but it exploits features intended to capture the usefulness of each phrase pair. Out-of-domain phrase pairs are weighted according to their relevance to the target domain, determined by both how similar to it they appear to be, and whether they belong to the general language or not. **?** also proposed an instance weighting scheme at the phrase-pair level based on the *vector space model* (VSM). The training corpus is divided into C sub-corpora based on their origin, which is supposed known. Each phrase pair $(\bar{s}, \bar{t})$ present in the phrase table is represented by a C-dimensional vector of TF-IDF scores, one for each sub-corpus. Each component $w_c(\bar{s}, \bar{t})$ is a standard TF-IDF weight of each phrase pair for the c^{th} sub-corpus. $tf(\bar{s}, \bar{t})$ is the raw joint count of $(\bar{s}, \bar{t})$ in the sub-corpus; the idf$(\bar{s}, \bar{t})$ is the inverse document frequency across all sub-corpora. A similar C-dimensional representation for the development set is computed as follows: word alignment and phrase pair extraction is performed; then, for each extracted phrase pair, its TF-IDF vector is computed and finally all vectors are combined to obtain the vector for the development set:

$$w_c^{dev} = \sum_{j=0}^{J} \sum_{k=0}^{K} count_{dev}(\bar{s}_j, \bar{t}_k) w_c(\bar{s}_j, \bar{t}_k) \tag{3.13}$$

where J and K are the total numbers of source and target phrases extracted from the development set, respectively, and $count_{dev}(\bar{s}_j, \bar{t}_k)$ is the joint count of the phrase pair $(\bar{s}_j, \bar{t}_k)$ in the development set. The similarity score between each phrase pair's vector and the development set represents the closeness of each phrase pair to the development set, and is added into the phrase table as an additional feature.[5]

3.3.4 Multi-domain adaptation

As previously discussed, SMT systems trained and tuned on domain-specific data perform well for their respective domains, but performance is comparatively lower for out-of-domain data. In almost all domain adaptation works, adaptation is performed during model training and only targets one specific domain. The usability of these approaches is thus limited in a multi-domain environment. Firstly, maintaining multiple domain-specific systems for all possible domains is expensive in terms of computational and human resources. Secondly, domain adaptation is performed based on the development set of the system. The optimality of adapted models on unseen test data relies on the assumption that both development and test data observe identical probabilistic distribution, which often does not hold for real-world data. Thirdly, the adaptation is performed once and for all, and the adapted models are kept unchanged for all subsequent

[5]Note that we reported experiments using the **?** in our laborabory's submission to the WMT'14 medical task [**?**].

test data. Its performance thus heavily depends on the homogeneity of the test data relative to the development data.

? adopted the data selection method for development set construction. A development set is dynamically built for a given input test set based on dataset similarity. A similarity measure between two sentences in the feature space of the model used in the log-linear SMT framework is used.

In [**?**], a sub-model is trained based on each domain-specific sub-corpus. Instead of combining the sub-models as in mixture modeling, a SVM-based classifier for domain-wise classification of the input test set sentences is used. A system is trained and tuned based on each domain specific sub-corpus and the respective domain development set. Each input sentence is classified into one of the domains and translated by the corresponding sub-system.

? presented an architecture that allows mixture modeling of translation models in a multi-domain environment. The sentences of the development set are clustered into several groups, and each cluster is treated as a sub-domain. The parameters of the mixture model and of the system are separately optimized based on each cluster of the development set. When translating an input text, each sentence is assigned to the cluster that is closest to it in the vector space.

3.3.5 Translation memory integration

SMT has developed very quickly in recent years. However, SMT is still barely used by professional translators because its quality is still far from satisfactory is many situations. In contrast, the **translation memory** (TM) technology has been widely used in the field of professional translation for many years [**?**]. When translating an input source sentence, a TM finds the most similar translation sentence (usually based on a fuzzy match threshold) in the database and uses it as the reference for *post-editing*. This approach is very useful for the translation of repetitive material, such as technical documents or weather forecasts, and can provide high quality draft translations when the similarity of the fuzzy matches is high. But for those unmatched or low fuzzy match score regions, a TM cannot provide helpful draft translations, and SMT may provide a more appropriate solution [**?**].

Since SMT and TM can complement each other, several works have attempted to integrate TMs into the SMT pipeline. **?**, **?** and **?** proposed a two-step approach. Firstly, it is determined whether the extracted TM sentence pairs should be adopted or not, based on their fuzzy match scores [**?**], or using syntactic information [**??**]. Then, the found matches are merged into the input source sentences, and the SMT system is forced to translate those matched regions with the given translations. In general, these approaches work well when the training corpus and the input text are very repetitive. In these approaches, a TM sentence could be either adopted or abandoned. To avoid making such hard decisions, **?** extracted long matches from a TM and added them into the SMT system phrase table. Similarly, **?** encoded the TM matches as very large hierarchical phrase rules, and used them in a secondary rule table of a hierarchical phrase-based model. **?** proposed a deeper level TM integration approach, in which additional features for all phrase pairs were extracted from the TM and added to the SMT phrase table.

3.4 Summary

In this chapter, we have presented not only several important methodologies but also lines of research for improving phrase-based SMT system development.

First, on-the-fly estimation of translation tables enables us to scale system development to arbitrary large corpora and arbitrary long phrases. Systems can extract translation tables on a per-need basis that are specialized for each given input. Data sampling enables to further reduce the computational burden of translation table extraction. However, in [**??**], the whole training corpus is always assumed to be pre-aligned. This strong assumption largely simplifies the matter, since in practice word alignment is much more time-consuming than translation table extraction.

Second, the presented incremental alignment methods and the approaches for model combination allows us to incrementally integrate newly available data into existing systems. However, existing incremental alignment approaches still require to wait for a sufficient quantity of supplementary data. Again, a well-estimated alignment model (usually trained on large-scale corpus) is also assumed to be available.

Third, different approaches have been presented to selectively use training data. Using WSD-like classifiers or context-informed features allows us to incorporate the source side context into translation models. However, such methods are quite expensive and are plagued by the data sparsity problem. Other approaches, such as data selection and instance weighting, allows us to adapt the translation models. But such adaptations depend on each particular translation task and are usually performed on some held-out development corpus. The optimality of the obtained model on the test data thus depends on the similarity between the development corpus and the test data, a difficult assumption in practice.

In a production environment, the training of word alignments on large-scale corpora and the preparation of a development corpus largely delay the production of translations. In the next chapter, we will introduce a framework for phrase-based SMT system development which enables us to efficiently develop SMT system on very large-scale training data from scratch. All the information needed for translation, including word alignments, will be only computed on a per-need basis (on-demand); the incorporation of new available data will be *plug-and-play* (incremental); and translation models can be adapted to the translation context.

Part II

On-demand Development and Contextual Adaptation for SMT

4 On-demand Development Framework of SMT Systems

Although SMT has become one of the prefered choices in many practical machine-assisted translation scenarios, a notable fact about this technology is that the construction of high-performance systems is expensive from a number of perspectives. In particular:

- Building SMT systems is *data-intensive*. Large to huge sets of parallel texts of the source-target language pair must be repeatedly analyzed and processed to train statistical models. Additionally, high-quality data sets must be available to tune and adapt systems to particular domains.

- Building large-scale SMT systems is *heavy on computing resources*. Processing all the available training data requires powerful computing configurations in terms of processors and working memory. Storing the resulting models also requires very large disk capacities, especially when keeping information for long phrases and large corpora. Actually, a large portion of computed information will never be used. Handling such resources at decoding time also necessitates powerful hardware configurations, including large working memory capacities to effectively explore the combinatorially large search space of SMT.

- Building large-scale SMT systems *takes a lot of time*. Even if using appropriate computing resources and parallel programming techniques, building systems for large data sets requires a significant preprocessing time before any translation can be produced. If individual processing steps may be greatly accelerated, including e.g. word alignment [**?**] or system tuning [**??**], the requirement to process the complete parallel data significantly delays the availability of a trained system. And even though a careful pre-selection of bilingual sentences may greatly reduce the size of the training material [**?**], this selection is itself time-consuming and is not justified when one only needs to translate a handful of documents or documents from various domains.

The above characteristics seriously impede the development of SMT systems in many large-scale situations, in addition to not being eco-friendly. For instance, building a complete system from scratch when only a small number of documents have to be translated is certainly counter-productive. For some uses, using a web-based automatic translation service can provide a viable, albeit suboptimal alternative, since the MT system is not adapted to the translation task. But it may also happen, especially for professional applications, that the customer cannot divulge the translated texts and/or the training material [**?**]. Thus the need for new training methodologies enabling light-weight system development without compromising too much translation quality.

Previous works have empirically shown that not all phrase translation examples are necessary to reach top performance, so that phrase tables can be built on a per-need basis for a given input text using random sampling of translation examples [**??**], and/or pruned heavily posterior to training [**???**]. The main strength of the former approaches is that they reduce the computation time and make it possible to extract translations from very large parallel data, even with arbitrarily long translation units. If they dispense with the need to estimate gigantic translation tables, containing a majority of entries that will never be used, these approaches still require to align all the available parallel data at the word level, a serious bottleneck when working with very large amounts of parallel data.

Even though sampling strategies significantly diminish the amount of processing[1] as well as disk usage, the ability to reduce computation for word alignment thus seems pivotal to making the full SMT system development process more efficient.

In this chapter, we describe an original architecture, *on-demand SMT system development*, which addresses the above issues by limiting computation to a minimum. In particular, not only the phrase tables but also the word alignments are computed on demand using a sampling-based strategy. Rather than attempting to extract phrase translations in isolation as **?**, the complete word-alignment of sentence pairs containing the source phrases of interest is computed, in an attempt to extract more precise translations. Our on-demand development system relies on two main components: on-demand word alignment and on-the-fly model estimation. Before describing them in some detail, we introduce in Table 4.1 the notations that will be used in the remainder of this thesis.

The rest of this chapter is organized as follows. We first introduce the two components of our system, on-demand word alignment and on-the-fly model estimation, respectively in Section 4.1 and Section 4.2. The procedures of our on-demand system is described in Section 4.3. We then present in Section 4.4 experiments designed to assess the performance of our on-demand framework on a standard system development scenario.

4.1 On-demand word alignment

The first originality of our system is the ability to perform word alignment on demand. In a traditional pipeline, word and phrase alignments, which are required to compute (4.8), are pre-computed during the training phase. In our system, such alignments are computed on-demand for parallel sentences that have been selected by sampling. If pre-computed word alignments

[1]For instance, **?** report a decrease from 1h40 to just 10s for translation retrieval when retrieving all translation examples or only 100 examples on a large word-aligned parallel corpus.

- $\mathbf{C}$ is a set of parallel sentences $\mathbf{C} = \{(\mathbf{s}_i, \mathbf{t}_i), i = 1 \ldots N\}$, made of $\mathbf{C}_{|\mathbf{s}}$ and $\mathbf{C}_{|\mathbf{t}}$ respectively the source and the target corpus.
- A (source) phrase $\bar{s}$ is an arbitrary sequence of contiguous words denoted $\bar{s}$; if $\bar{s}$ is a substring of a sentence $\mathbf{s}$, this is denoted as $\bar{s} \sqsubset \mathbf{s}$. A target side phrase is denoted $\bar{t}$.
- $\mathbf{C}[\bar{s}]$ is the set $\{(\mathbf{s}, \mathbf{t}) \in \mathbf{C}, \bar{s} \sqsubset \mathbf{s}\}$.
- $\mathbf{S}[\bar{s}]$ is a sample selected from $\mathbf{C}[\bar{s}]$ by sampling without repetitions ($\mathbf{S}[\bar{s}] \subseteq \mathbf{C}[\bar{s}]$), and $|\mathbf{S}[\bar{s}]| \leq M$ where M is the maximum sample size.
- An input document $\mathbf{d}$; it comprises sentences which contain phrases; $\boldsymbol{\Sigma}[\mathbf{d}]$ is the set of all phrases occurring in $\mathbf{d}$.
- The model parameters of interest are mostly phrase translation probabilities; the complete parameter set is $\boldsymbol{\theta}$, while $\boldsymbol{\theta}_{\bar{s}}$ is the set of parameters for a given phrase $\bar{s}$.

Table 4.1 – Notations used in the remainder of this thesis.

are already available, our system will use them directly.

More formally, let us assume that a set $\mathbf{c}$ of parallel sentence pairs need to be aligned. Note that $\mathbf{c}$ can be a subpart of the training corpus $\mathbf{C}$ ($\mathbf{c} \subseteq \mathbf{C}$), or can correspond to newly available data ($\mathbf{c} \nsubseteq \mathbf{C}$). An association table, which for now only contains the source phrases that occur in some sentences of $\mathbf{c}$, is first extracted by a **sampling-based transpotting** method that will be detailed in Section 4.1.1. Using this table, a **recursive binary segmentation** algorithm, inspired from the work of **?**, is applied to each sentence pair in $\mathbf{c}$ so as to generate the required sub-sentential alignment. This approach will be described in Section 4.1.2.

4.1.1 Sampling-based transpotting[3]

The sampling-based transpotting method is inspired by the `Anymalign` system [**??**], which aims at extracting sub-sentential associations from multilingual parallel corpora. `Anymalign` repeatedly draws random sub-corpora from a parallel corpus, and extracts associations from each sub-corpus, which are used to build an association table between phrases. As each sub-corpus is processed independently, this process can be trivially parallelized and can be stopped at any time. However, large numbers of sub-corpora must be processed in order to achieve a good coverage of the phrases over the entire corpus.

In this work, the behavior of `Anymalign` is adapted in order to extract an association table for a specific list of sentence pairs $\mathbf{c}$. Each sentence pair $(\mathbf{s}, \mathbf{t})$ in $\mathbf{c}$ is processed separately and a number of random sub-corpora are sampled from the full parallel corpus $\mathbf{C}$ for each

[3]The term *transpotting* is used by **?** to represent *translation spotting* which is defined as a task of identifying the target language tokens that correspond to a given source language query in a parallel sentence pair.

sentence pair. For each sub-corpus, the distribution profile is computed only for units (words and phrases) occurring in $\mathbf{s}, \mathbf{t}$, and bilingual units with the same profile are extracted as likely associations. The more sub-corpora are processed for each sentence pair, the more associations may be extracted, and the more accurate the association measures are. The set of all associations extracted from all sentence pairs forms the association table of $\mathbf{c}$. In a nutshell, this procedure extracts association table *via* transpotting based on randomly sampled sub-corpora.

The complete process is illustrated on an English-French sentence pair in Figure 4.1. The complexity of this procedure for extracting an association table for a given sentence pair depends on the size of selected sub-corpora and the number of iterations (N). In this work, we choose to use only small size sub-corpora[4], and thus the complexity of this process is $\mathcal{O}(N)$.

There are notable differences between this method and `Anymalign`:

- `Anymalign` draws random sub-corpora from the parallel corpus, and computes the occurrence distribution profile for all words of all sentence pairs in the sub-corpora, while we only need to compute such profiles for words in the sentence pair that we wish to align.[5]
- `Anymalign` is *anytime* but typically requires a large number of sub-corpora to achieve a good coverage over the entire corpus. We draw N sub-corpora for each given sentence pairs to ensure a good coverage for the contents of each sentence pair to align. This enables to align sentences on a per-need basis, and furthermore offers a more interpretable running time, which is now controlled by the amount of desired sampling for each sentence pair, which could e.g. depend on its length.

4.1.2 Sub-sentential alignment extraction

Once the association table for one sentence pair is obtained, a recursive binary segmentation process, described in [**?**], and inspired by the work of **?** and **?**, is used to generate a sub-sentential alignment. Its main principle is to recursively segment the source and target sentence simultaneously on the basis of local association scores so as to find the *alignment links* between the source and target words. It thus requires some association score $w(s,t)$ between each source word s and each target word t in a sentence pair, which can be the result of the sampling-based transpotting process described in the previous section. It is worth noting that any kind of lexical scores could be in fact be used here to measure the strength of word associations.

The score $w(s,t)$ between a source word s and target word t is defined as the product of the two translation probabilities $p(s|t) \times p(t|s)$, computed from the association table produced by transpotting:

$$\begin{aligned} w(s,t) &= p(s|t) \times p(t|s) \\ &= \frac{\sum_{e=1}^{E} [\![(s,t) \in (\bar{s}_e, \bar{t}_e)]\!] c_e}{\sum_{e'=1}^{E} [\![s \in \bar{s}_{e'}]\!] c_{e'}} \times \frac{\sum_{e=1}^{E} [\![(s,t) \in (\bar{s}_e, \bar{t}_e)]\!] c_e}{\sum_{e'=1}^{E} [\![t \in \bar{t}_{e'}]\!] c_{e'}} \\ &= \frac{(\sum_{e=1}^{E} [\![(s,t) \in (\bar{s}_e, \bar{t}_e)]\!] c_e)^2}{\sum_{e'=1}^{E} [\![s \in \bar{s}_{e'}]\!] c_{e'} \times \sum_{e'=1}^{E} [\![t \in \bar{t}_{e'}]\!] c_{e'}} \end{aligned} \quad (4.1)$$

[4]In our experiments, the size of each sub-corpus is randomly selected between 1 and 100.

[5]Note that, when one's objective is in fact to align a complete parallel corpus (all sentence pairs in the corpus), all counts should be kept.

Input: Given a source-target sentence pair, extract an association table :

one diet coke , please . ↔ un coca zéro, s'il vous plaît .

(1) Draw a random sub-corpus from the parallel corpus*:

	English	French
1	one coffee, please .	un café, s'il vous plaît .
2	the coffee is not bad .	ce café est correct .
3	yes, one tea .	oui, un thé .

(2) Compute occurrence profile for each word in the current sentence pair:

words with same distribution profile			profiles
diet coke	↔	coca zéro	[0, 0, 0]
please	↔	s'il vous plaît	[1, 0, 0]
.	↔	.	[1, 1, 1]

(3) Increase the count for each contiguous phrase pairs:

1. count("one ,", "un ,") $+ = 1$
2. count("diet coke", "coca zéro") $+ = 1$
3. count("please", "s'il vous plaît") $+ = 1$
4. count(".", ".") $+ = 1$

(4) Repeat steps (2) to (4) N times, so as to obtain an association table for the given sentence pair, e.g.:

Output:

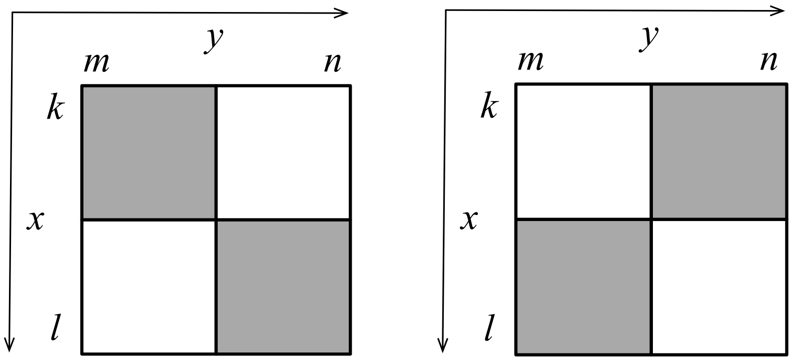
source phrase		target phrase	count
one	↔	un	830
coke	↔	coca	680
diet coke	↔	coca zéro	260
diet coke	↔	zéro	54
diet	↔	coca zéro	87
coke ,	↔	,	30
,	↔	,	900
please	↔	s'il vous plaît	160
.	↔	.	980

* The sampling could depends on the input sentence pair to improve the efficiency of the extraction of associations, this is a part of our future work. In this work, sub-corpora are randomly sampled from the training corpus.

Figure 4.1 – Illustration of the sampling-based transpotting method on an English-French sentence pair.

$$\begin{aligned} w(coke, coca) &= p(coca|coke) \times p(coke|coca) \\ &= \frac{680 + 260}{680 + 260 + 54 + 30} \times \frac{680 + 260}{680 + 260 + 87} \\ &\simeq 0.840 \end{aligned}$$

Figure 4.2 – The computation of association score between English word *coke* and French word *coca* using the association table in Figure 4.1.



(a) straight rule (b) inversion rule

Figure 4.3 – Straight rule and inversion rule for the segmentation of a pair of segments, where the gray blocks indicats that the corresponding two segments are aligned.

where:

- $[\![x]\!] = 1$ if x is true, 0 otherwise;
- E is the number of entries (source-target phrase pairs) in the association table;
- $\bar{s}_e$ and $\bar{t}_e$ are the source and target part of an entry in the association table, respectively;
- c_e is the associated count of the phrase pair $(\bar{s}_e, \bar{t}_e)$ in the association table.

This computation is illustrated on Figure 4.2.

The binary segmentation process role is to find the best segmentation point for a given block. As shown in Figure 4.3, the given block $(k, l; m, n)$ is segmented at the point (x, y), where $k \leq x < l$ and $m \leq y < n$. Two possible segmentation rules could be applied to point (x, y): the straight rule (see Figure 4.3a), denoted $r^S_{k,l;m,n}$; and the inversion rule (see Figure 4.3b), denoted $r^I_{k,l;m,n}$. This segmentation process is guided by the sum W of the association scores between each source and target words in sub-blocks. As shown in Figure 4.3, for example, the sum of the association scores in the top-left block is given by:

$$W_{k,x;m,y} = \sum_{\substack{k \leq i < x \\ m \leq j < y}} w(s_i, t_j) \tag{4.2}$$

and the score of the segmentation rules at a given point (x, y) is given by the sum of the association scores in those *unaligned* sub-blocks:

$$r^{S}_{k,l;m,n}(x,y) = W_{k,x;y,n} + W_{x,l;m,y} \tag{4.3}$$

$$r^{I}_{k,l;m,n}(x,y) = W_{k,x;m,y} + W_{x,l;y,n} \tag{4.4}$$

Then, the best segmentation for the given block $(k,l;m,n)$ is the one which minimizes the rule score, defined as:

$$cut_{k,l;m,n} = \underset{x,y}{\operatorname{argmin}} \min(r^{S}_{k,l;m,n}(x,y), r^{I}_{k,l;m,n}(x,y)) \tag{4.5}$$

Instead of using the rule scores defined in (4.3) and (4.4), **?** used a normalized variant version, where:

$$r^{S-norm}_{k,l;m,n}(x,y) = \frac{W_{k,x;y,n}+W_{x,l;m,y}}{W_{k,x;y,n}+W_{x,l;m,y}+2\times W_{k,x;m,y}} + \frac{W_{k,x;y,n}+W_{x,l;m,y}}{W_{k,x;y,n}+W_{x,l;m,y}+2\times W_{x,l;y,n}} \tag{4.6}$$

$$r^{I-norm}_{k,l;m,n}(x,y) = \frac{W_{k,x;y,n}+W_{x,l;m,y}}{W_{k,x;y,n}+W_{x,l;m,y}+2\times W_{x,l;m,y}} + \frac{W_{k,x;y,n}+W_{x,l;m,y}}{W_{k,x;y,n}+W_{x,l;m,y}+2\times W_{k,x;y,n}} \tag{4.7}$$

We reuse this definition in the experiments of this thesis,

The binary segmentation algorithm tests every possible binary segmentation point to find the best segmentation that minimizes the rule score, and recursively segments blocks in a greedy fashion. This segmentation process terminates on blocks when one side of a block only contains one single word. Figure 4.4 shows an example of segmentation, where atomic aligned biphrases correspond to framed rectangles containing values in bold. The words in aligned bi-phrases are linked with each other and define the word alignment of the bi-sentence.

This process can be viewed as approximate top-down parsing using Inverse Transduction Grammars (ITG) [**?**], where matching blocks are determined based on association scores between the words in the source and target sentences. ITG generates synchronized binary parse trees in source and target languages. This formalism models both variable-length associations at leaf nodes, and reordering at any level of the parse tree. The time complexity of parsing is $\mathcal{O}(n^6)$ [**?**], where n is the length of source or target sentences. Time complexity was reduced to $\mathcal{O}(n^4)$ by **?** by using A* search heuristics. **?** used a beam search algorithm to reduce time complexity of ITG parsing from $\mathcal{O}(n^6)$ to $\mathcal{O}(bn^3)$, where b is the beam size in the algorithm.

Regarding the recursive binary segmentation algorithm, each step has to consider $\mathcal{O}(n^2)$ segmentation points, a computation of the rule score on one segmentation point is $\mathcal{O}(n^2)$; and the number of segmentations is upper bounded by the minimum length of the source and the target sentences. Hence, its complexity is $\mathcal{O}(n^5)$. This time complexity can be easily reduced to $\mathcal{O}(n^4)$ by using dynamic programming.

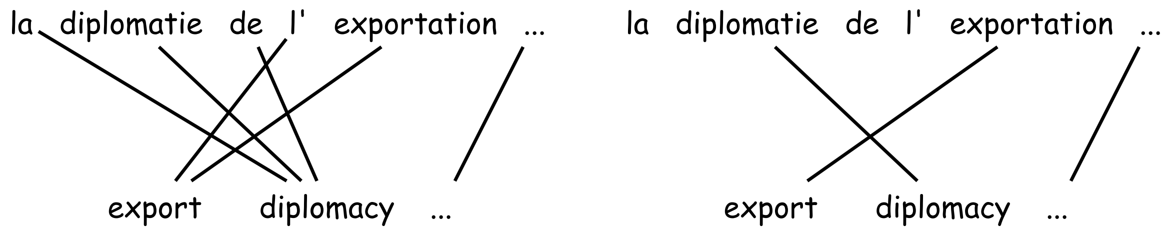
4.1.3 Difference with the `giza++` alignment process

As presented before, `giza++` alignments are based on the IBM models, which are asymmetrical. The alignment process needs to be performed in both the source-to-target and target-to-source directions before the two resulting word alignments are merged (or *symmetrized*) by heuristics.[6] In our word alignment method, alignments are generated by recursive binary segmentation of the pairs of sentences, which thus dispenses with the need to perform some additional symmetrization process.

[6]One of the most used heuristics is the so-called *grow-diag-final* heuristic [**?**].

	un	coca	,	s'il	vous	plaît	.
one	**0.246**	ϵ	ϵ	ϵ	ϵ	0	ϵ
coke	ϵ	**0.840**	ϵ	ϵ	ϵ	ϵ	ϵ
,	ϵ	ϵ	**0.624**	0.002	ϵ	ϵ	0.048
please	ϵ	0	ϵ	**0.032**	**0.008**	**0.128**	ϵ
.	ϵ	ϵ	0.020	ϵ	ϵ	ϵ	**0.873**

Figure 4.4 – Illustration of the recursive alignment process. The number in each cell corresponds to the value of the association score, with $0 < \epsilon \leq 0.001$. A null value indicates that the two words never appear together in the translation table. Alignment points retained by the algorithm, i.e. at maximum level of recursion, are shown in grey.


la diplomatie de l' exportation ...
export diplomacy ...
la diplomatie de l' exportation ...
export diplomacy ...

(a) no non-aligned tokens (b) allowing non-aligned tokens

Figure 4.5 – Possible word alignments for an extract of French-English sentence pair.

Another important difference between our method and that implemented in `giza++` is that the latter allows unaligned words in the sentences while the former does not. Our recursive binary segmentation process stops when one block can no longer be further segmented, and such a block is then considered as completely aligned, which means that all source words are aligned to all target words. Such a property significantly reduces the number of translations that can be extracted from the sentence-level alignment. This is a double-edged sword: on the one hand, the produced translation table will be more compact and decoding will be accelerated; on the other hand, this reduces the flexibility of translation models. A significant part of typically unaligned tokens correspond to function words which are often not translated literarily but are associated with other words. Aligning these function words will force them to be extracted together with other words, and will thus reduce the number of translations that will be extracted. Considering the example shown in Figure 4.5, the alignment in 4.5a only permits to extract the two following phrase pairs: (*la diplomatie de*, *diplomacy*) and (*l' exportation*, *export*). However, in 4.5b, besides these two phrase pairs, several other variants will be extracted, including e.g. (*diplomatie*, *diplomacy*), (*la diplomatie*, *diplomacy*).

Finally, the most important difference is that our method can align each parallel sentence pair in isolation, while `giza++` cannot do so. As presented before, `giza++` needs to analyze the whole parallel corpus to collect cooccurrence statistics to compute the complete set of alignments. However, our sampling-based transpotting only collects association statistics for the necessary sentence pairs. It therefore does not need to analyze the whole corpus and can generate word alignments for any single sentence pair, be it from the original parallel corpus or from some newly available data set.

4.2 On-the-fly model estimation

A first major difference between our system and a standard SMT pipeline is the ability to compute phrase translation probabilities on a per-need basis. Parallel sentence pairs are stored in a suffix array [?], enabling fast access to phrase instances.[7] Translation probabilities for all source phrases $\bar{s}$ (up to a given length) that need to be translated are computed based on a subset of their occurrences. The sample size is a parameter that enables to balance between speed and precision of estimates.

Previous approaches to sampling [??] have resorted to *deterministic random sampling*. Such a strategy picks a given number of examples by scanning the suffix array index at fixed intervals, hence the apparently random, and actually deterministic, behavior. This is a simple and effective way to select a sample for a given source phrase. The translation probability of a source phrase is then computed as:

$$p(\bar{t}|\bar{s}) = \frac{\text{count}(\bar{s},\bar{t})}{\sum_{\bar{t}'} \text{count}(\bar{s},\bar{t}')} \tag{4.8}$$

where count$(\cdot)$ is the number of occurrences of the given phrase pair *in the sample*.

Additionally, the sample may include occurrences where translation extraction will not be possible. As presented in Section 2.3, a phrase pair could be extracted if and only if it is consistent with the word alignment. For example, in Figure 4.5, the French phrase *exportation* could not be extracted in 4.5a since it is aligned to the English word *export* which is also aligned to another French word *l'*. They are not consistent with the word alignment, and hence could not be extracted. However, it could be extracted with the English word *export* in 4.5b. Based on the definition of *coherent estimation* proposed by **?**, such occurrences are nonetheless taken into account in Equation (4.8) to penalize source phrases with frequent extraction failures.

Given an input document $\mathbf{d}$ to translate, the system extracts all possible source phrases $\mathbf{\Sigma}[\mathbf{d}]$ from $\mathbf{d}$. Then, for each extracted source phrase $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$, we perform the sampling[8], to select a translation sample $\mathbf{S}[\bar{s}]$ from the corpus. If available, the word alignments $\mathbf{A}$ are then used to extract the translations and to compute model parameters $\boldsymbol{\theta}_{\bar{s}}$ for the source phrase; otherwise, these alignments are computed on-the-fly (see Section 4.3). The estimation of each source phrase, as described in Section 4.2, is thus only based on the corresponding sample. This process is repeated for all source phrases in $\mathbf{\Sigma}[\mathbf{d}]$, so as to extract the phrase table and reordering table for the input document. This procedure is sketched in Algorithm 2.

Algorithm 2 On-the-fly model estimation using existing alignments

```
parallel corpus $\mathbf{C}$, alignments $\mathbf{A}$
given an input document $\mathbf{d}$, sample size $M$
compute $\mathbf{\Sigma}[\mathbf{d}]$
for all $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$ do
    $\mathbf{S}[\bar{s}] = \texttt{sampling}(M, \mathbf{C}, \bar{s})$   // Sampling
    $\texttt{estimate}(\boldsymbol{\theta}_{\bar{s}}, \mathbf{S}[\bar{s}], \mathbf{A})$   // Estimation
end for
```

[7]Querying a suffix array for a phrase of k words can be performed in $(k + \log(|\mathbf{C}|))$ operations [?], where $|\mathbf{C}|$ is the corpus size. A suffix array could be constructed in $\mathcal{O}(|\mathbf{C}| \log(|\mathbf{C}|))$ time in the worse case.

[8]Here, the sampling strategy could be of any kind.

As sampling is performed independently for each source phrase, the computation of the inverse translation probability $p(\bar{s}|\bar{t})$ can no longer be performed exactly based on the sole selected sample. Indeed, the same sampling process would have to be performed for each extracted translation to compute the inverse translation probability, which would make the estimation process become computationally very expensive. As reported in [**?**], inverse translation probabilities can in fact be removed from the system without any significant loss of accuracy. If needed[9], the following approximation can be used instead:

$$p(\bar{s}|\bar{t}) = min(1.0, \frac{p(\bar{t}|\bar{s}) \times freq(\bar{s})}{freq(\bar{t})}) \tag{4.9}$$

where, the $p(\bar{t}|\bar{s})$ represents the on-the-fly estimated direct translation probability, $freq(\cdot)$ is the frequency of the given phrase in the entire corpus, and the numerator $p(\bar{t}|\bar{s}) \times freq(\bar{s})$ represents the predicted joint count of $\bar{s}$ and $\bar{t}$. This calculation is, by design, compatible with the on-demand estimation approach, with no significant increase in computational complexity. The only extra cost resides in computing $freq(\bar{s})$ and $freq(\bar{t})$, which are efficiently obtained with a suffix array.

All the other models needed to run `moses` [**?**] with its default setting, including lexicalized reordering models, are also estimated based on the samples (see Section 2.3).

The sample size M enables to balance between speed and precision of estimation. A larger sample size grants more precise model estimation but is computationally more expensive; a smaller sample size makes the process faster but at the cost of a lower model quality. This parameter allows users to configure the system based on their needs.

In addition, this approach can be used to extract translations for phrases of arbitrary length, which would be computationally prohibitive in a standard offline SMT pipeline. Such a property can prove useful when the training corpus contains stereotypical text that matches the genre of the text to translate [**?**]. In practice, however, only a limited number of translation situations (e.g. the corpus and input text are very repetitive) will benefit from this property.

4.3 On-demand system development

The complete architecture of our framework is illustrated in Figure 4.6. Given an input document $\mathbf{d}$ to translate, the system first extracts all possible source phrases, $\mathbf{\Sigma}[\mathbf{d}]$. Then, for each extracted source phrase $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$, we perform deterministic random sampling (denoted as `rnd` henceforth) to select translation examples from the parallel corpus. We then obtain a translation sample of $\bar{s}$, $\mathbf{S}[\bar{s}]$, where $\mathbf{S}[\bar{s}] \subseteq \mathbf{C}[\bar{s}]$. The sentence pairs in $\mathbf{S}[\bar{s}]$ are then aligned by our on-demand word alignment (denoted as `owa` henceforth), where the generated alignments are denoted as $\mathbf{A}_{\mathbf{S}[\bar{s}]}$, and are then used to extract the translations and to compute model parameters $\boldsymbol{\theta}_{\bar{s}}$ for the source phrase $\bar{s}$. This process is repeated for all source phrases in $\mathbf{\Sigma}[\mathbf{d}]$, and the resulting translation table can then be used by a phrase-based decoder to translate the input text into the target language. This procedure is sketched in Algorithm 3.

Note that `rnd` is for now performed independently for each source phrase $\bar{s}$ with a predefined sample size M. If a source phrase $\bar{s}$ has fewer than M occurrences in the whole corpus,

[9]Although this model has been shown to be non essential, we use it for the stability of our system, especially when the systems are not tuned. Our previous experiments showed that performance drops significantly on untuned system when not using an inverse translation model or our proposed approximation.

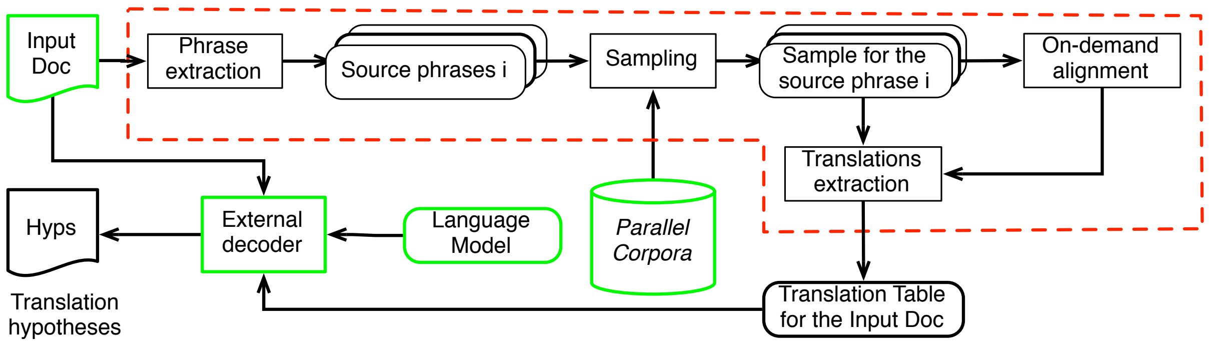


Figure 4.6 – Main processes of our framework (delimited by a red frame).

then they will all be selected. Different source phrases may occasionally cause the selection of the same sentence pairs from the corpus, but our system will align each sentence pair only once, a straightforward optimization.

As mentioned before, because of the sampling scheme, the inverse translation probability $p(\bar{s}|\bar{t})$ can no longer be estimated exactly based on the selected samples, so the approximation estimation of Equation (4.9) is used instead. Lexical translation probabilities, which are used for estimating lexical weighting features (cf. Equation (2.6)), are computed based on the computed `owa` alignments during the estimation process, and the lexical reordering features are also estimated based on the computed `owa` alignments.

Algorithm 3 Standard procedure for on-demand system development

```
parallel train corpus $\mathbf{C}$
given an input document $\mathbf{d}$, sample size $M$
compute $\mathbf{\Sigma}[\mathbf{d}]$
for all $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$ do
    $\mathbf{S}[\bar{s}] = \texttt{rnd}(M, \mathbf{C}, \bar{s})$   // Sampling
    $\mathbf{A}_{\mathbf{S}[\bar{s}]} = \texttt{owa}(\mathbf{S}[\bar{s}])$   // Alignment
    $\texttt{estimate}(\boldsymbol{\theta}_{\bar{s}}, \mathbf{S}[\bar{s}], \mathbf{A}_{\mathbf{S}[\bar{s}]})$   // Estimation
end for
```

4.4 Experimental validation

We haved designed a number of experiments intended to validate our framework. In this section, we first test separately on-the-fly model estimation and on-demand word alignment, and we then test a complete on-demand SMT system and compare its performance with the state-of-the-art SMT system `moses`.

4.4.1 Data

We selected English-French as our main language pair for this study, mostly because large quantities of parallel data are readily available for this language pair. Data from the Workshop on Statistical Machine Translation (WMT)[10] from a variety of domains were used, as well as

[10] http://www.statmt.org/wmt13

additional data from various origins from the medical domain and used in the WMT'14 medical task.[11] This parallel corpus, denoted as `WMT`, contains data from different domains, including news commentaries, parliamentary debates and medical texts. It will be used in almost all of our experiments in this chapter and subsequent chapters.

For testing, we used data from several origins: news commentary data sets from WMT, medical summaries from the WMT'14 medical task, as well as systematic summaries for specialists from the `Cochrane` collaboration.[12]

The news commentary data set is composed of news reports and analyses. They were written in different original languages and then (possibly) translated into English and French. To avoid issues related to *translationese* [**??**], we use only the English original subpart of the `Newstest` released from 2008 to 2012, in which the data from `newstest2008` and `newstest2009` are used as the development set, and the data from the others years are used as the test set.

The other two data sets (WMT'14 medical and Cochrane) are both originally written in English. In the `WMT'14-med` data, English test sentences were randomly sampled from automatically generated summaries of documents containing medical information intended for the general public or health professionals, found to be relevant to 50 topics provided for the `CLEF 2013 eHealth Task 3`[13]. The `Cochrane` data are systematic reviews which summarize the results of carefully designed healthcare studies (controlled trials) and provide a high level of evidence on the effectiveness of healthcare interventions. A main difference between the two medical data sets is that the `Cochrane` data set is made up of complete documents, while the WMT'14 medical data set if made up of independent sentences.

An extract of each of these data sets is provided in Appendix B.

Table 4.2 provides basic statistics regarding these corpora. Tokenization was performed using tools developed at LIMSI for English and French. Four 4-gram language models, one for each domain and for each language, were built. The English and French news-domain LM was used by LIMSI for WMT'13 [**?**]; the French medical-domain LM was used for the WMT'14 medical task [**?**]; and the English medical-domain LM was trained on the English side of the WMT'14 medical parallel data (containing 4.8M sentences (78M tokens)). All LMs were estimated using modified Kneser-Ney smoothing [**??**] using the SRILM toolkit [**?**].

All systems were optimized with `KBMIRA`, a variant of the Margin Infused Relaxation Algorithm described in [**?**]. Translations are computed with the `moses` phrase-based decoder. Results are reported using the BLEU [**?**] and TER [**?**] metrics. For stability, all reported experiments correspond to the test set average of 3 optimization runs [**?**].

4.4.2 Validation of on-the-fly model estimation

The previous studies of **?** and **?** have shown that deterministic random sampling yields results that are close to that of a system trained on all available data, and that the target-to-source translation probabilities can be removed without any significant loss in accuracy. Our first set of experiments is designed to reproduce these results within our framework.

We first constructed a vanilla `moses` system for English-to-French translation. We ran

[11] `http://www.statmt.org/wmt14`
[12] `http://summaries.cochrane.org`
[13] https://sites.google.com/site/shareclefehealth/

Corpus	# lines	# English token	# French token
***train* (en-fr)**			
WMT	16.6M	396.9M	475.1M
***tuning* (en-fr)**			
newsco	719	17.5K	20.1K
WMT'14-med	500	10.3K	12.2K
Cochrane	743	16.5K	21.4K
***test* (en-fr)**			
newsco	2.2K	51.8K	62.1K
WMT'14-med	1K	21.4K	25.9K
Cochrane (100 docs)	1.8K	38.6K	49.3K

Table 4.2 – Description of corpora used in our experiments.

mgiza++ on the complete available bi-corpus. With the resulting alignments, we used the moses scripts to extract the (huge) phrase table and the reordering table for the entire parallel corpus. These extracted tables were then used to construct our baseline system.

Using the same word alignments, we also constructed another system using the deterministic random sampling rnd (see Algorithm 2). As described in Section 4.2, the phrase table and the reordering table are no longer extracted in advance, but extracted for each individual input text.

As described in Section 4.4.1, three test sets of different origins were used. For Newstest, we used the WMT'13 language model, and for the others two test sets (WMT'14-med and Cochrane), we used the WMT'14 medical language model. We also tried several sampling sizes. Importantly, in all these experiments, we always used the same word alignments generated by mgiza++ on the whole training corpus.

Results

Results are presented in Table 4.3, where rnd-M represents the systems constructed using rnd with sampling size M.

As expected, we find that using on-the-fly model estimation only slightly degrades the performance of the SMT systems. For the English-to-French translation direction, the on-the-fly systems are on par with the vanilla moses systems. On the Newstest task, the performance drops by about 0.2 BLEU point (BP) or 0.2 TER point (TP) when using rnd with a sample size of 1000. However, there is no significant change when decreasing the sample size to 500 and 100, and the results with $M = 500$ are slightly superior to those of the other configurations.

On the Cochrane task, the results obtained by sampling are even closer to those of the vanilla moses system, with a loss lower than 0.1 BP in all configurations. Again, there is no significant change when decreasing the sample size down to $M = 100$.

On the WMT'14-med task, a modest 0.15 BP is lost when using a sample size of $M = 1000$. Surprisingly, a slightly better performance than the vanilla moses system is obtained for $M = 500$, but it decreases for $M = 100$. After inspecting the phrase tables, we found that when reducing the sample size from 1000 to 500, the number of source phrases in the phrase tables did not change, but when we decrease it to 100 a significant number of source phrases disappeared from the phrase table, which means none of the selected 100 examples contain extractable translations for some source phrases. These disappeared phrases may account for the loss observed.

In a nutshell, our empirical results confirm that using on-the-fly model estimation by random sampling and removing the standard inverse translation model from the phrase table achieves results that are close to a vanilla `moses` system. However, we observe that the minimum sample size to use depends on the origin of the input data.

We now turn our attention to the computational cost for constructing these systems. Since not all processes during system construction support multi-threading, it is more adequate to use the *user CPU time* (as given by the Unix `time` command[14]) as the main indicator of the time cost of usage of computing resources. During the construction of the vanilla `moses` system, computing the `mgiza++` alignments took $1,006$h, extracting the complete phrase table and reordering table for the entire training corpus took another 206h. Hence, it took $1,212$h[15] in total to extract the phrase table and the reordering table.

The extracted tables are huge: the phrase table and reordering table weight respectively 20Gb and 7.5Gb compressed on disk. Hence, they have to be filtered before they can be used in practice. In the on-the-fly model estimation approach, a phrase table and reordering table are built for each particular input text. Word alignments for the entire corpus are pre-computed, which took $1,006$h. But the table extraction for each input text is now much faster. It took less than 1h for any value of M[16]. The extracted tables are much more compact (< 50Mb) and do no have to be filtered before translation.

In the French-to-English translation direction, results are quite different. Looking at the results obtained with sampling in Table 4.3, there are no significant differences in performance when using different values of M. However, comparing with the `moses` system, the result on the `Newstest` is quite different from the results on the two other tasks. Taking the `rnd`-100 configuration as example, on the `Newstest` task, it is worse than the `moses` system by -0.8 BP. However, `rnd`-100 is better than the `moses` system on the `WMT'14-med` task ($+0.4$ BP) and on the `Cochrane` task ($+0.4$ BP). We mostly attribute this difference to the domain of the translation task. In the `Newstest` domain, the translations of phrases are more diverse than in the medical domain. Consequently, losing useful translations as a result of sampling is more likely. However, in the medical domain, the translations of phrases are relatively less diverse. A small number of examples is thus often sufficient to guarantee a good representation of translation distribution.

4.4.3 Validation of on-demand word alignment

We now describe experiments intended to assess the performance of our on-demand word alignment approach (`owa`) presented in Section 4.1. We focus on measuring the impact of several alignment strategies to build phrase-based SMT systems. Again, we used the `moses` toolkit. Its

[14]The `time` utility executes and times utility. After the utility finishes, `time` writes the total time elapsed, the time consumed by system overhead, and the time used to execute utility to the standard error stream.

[15]More intuitively, it took us 252h wall clock time, or 10.5 days, to construct such a system using 8 threads.

[16]In our current implementation, the time for loading data into memory is longer than the time needed for computation.

	`Newstest`		`WMT'14-med`		`Cochrane`	
	BLEU	**TER**	**BLEU**	**TER**	**BLEU**	**TER**
	English $\to$ French					
`moses`-vanilla	35.84±0.05	46.14±0.10	40.84±0.11	42.23±0.09	34.12±0.10	48.59±0.22
`rnd-1000`	35.62±0.10	46.30±0.12	40.69±0.07	42.21±0.02	34.05±0.07	48.86±0.11
`rnd-500`	35.69±0.09	46.23±0.08	40.87±0.12	42.23±0.04	34.10±0.05	48.56±0.07
`rnd-100`	35.64±0.15	46.31±0.08	40.18±0.06*	42.44±0.08	34.08±0.05	48.48±0.27
	French $\to$ English					
`moses`-vanilla	37.45±0.03	44.29±0.04	39.85±0.05	39.77±0.04	37.11±0.08	42.81±0.08
`rnd-1000`	36.87±0.09*	44.78±0.09	40.26±0.05*	39.65±0.06	37.60±0.09*	42.38±0.05
`rnd-500`	36.93±0.10*	44.78±0.24	40.28±0.05*	39.61±0.04	37.67±0.07*	42.39±0.04
`rnd-100`	36.64±0.05*	45.04±0.13	40.21±0.01*	39.84±0.05	37.48±0.14*	42.54±0.04

Table 4.3 – Translation performance (using mean ± standard deviation of 3 runs, * significant at $p < 0.01$ level) with different sampling sizes (M).

scripts were used in all configurations to build phrase tables and reordering tables from alignment matrices, and its decoder was used to compute candidate translations during optimization and testing.

Experiments were conducted on three language pairs and three corpora, and we made use of several reference translations when possible. The compactness of the produced phrase tables will be considered as a performance indicator, as it can be regarded as a desirable property of phrase tables licensing works on phrase table pruning [**???**].

Two sets of experiments are reported in this section. The first set of experiments is designed to validate our on-demand word alignment `owa` on some predefined bilingual corpus against `mgiza++` in translation tasks. The second set of experiments aims to assess the ability to align new bilingual data that do not belong to the original training data. For this latter experiment, we will focus on adding sentence pairs from a very large (unaligned) bilingual corpus, chosen on the basis that they contain translations for previously out-of-vocabulary (OOV) tokens. Our approach will be compared against the same alignment pipeline using the new, augmented parallel corpus. This strategy is however costly as it requires to retrain the complete models, so we also performed a comparison with alignments obtained using the original alignment models (forced alignment), without any retraining.

In these experiments, the whole bilingual parallel corpus is aligned by different alignment approaches. Instead of using the large-scale `WMT` corpus (cf. Table 4.2), which would be very costly to align in its entirety, we performed the experiments on three relatively smaller parallel corpora. Their statistics are shown in Table 4.4: `News Commentary` of WMT is a subpart of our `WMT` corpus; `EMEA` is a small subset of medical data from the European Medicine Agency (EMEA)[17]; and `HIT` is a corpus of basic traveling expressions built for the Beijing 2008 Olympics, which is available in multiple languages. We used it here in English, French and Chinese.

For the experiments on the `News Commentary` corpus, we used the same `newsco` tuning and test data set as described in Table 4.2; and for the `EMEA` corpus, experiments were performed on both the `WMT'14-med` and the `Cochrane` data sets (cf. Table 4.2). For the experiments on the `HIT` corpus, we used the `BTEC`[18] development set of 2003 (`devel03`) and

[17]`http://opus.lingfil.uu.se/EMEA.php`
[18]`http://iwslt2010.fbk.eu/node/32`

Corpus	# lines	# En tokens	# Fr tokens	# Zh tokens
News Commentary	137K	3.3M	4.0M	-
EMEA	324K	5.5M	6.6M	-
HIT	62K	600K	690K	590K
supp	3.3K	111K	121K	-

Table 4.4 – Description of the corpora used for validation experiments of on-demand word alignment.

Corpus	# lines	Avg(# En tokens)	# Fr tokens	# Zh tokens
devel03	506	4,098 (16 refs)	4,220	3,435
test09	469	3,928 (7 refs)	4,023	3,031

Table 4.5 – Description of the tuning and test sets for the HIT corpus.

the its test set of 2009 (test09) as our development and test sets (see Table 4.5). Note that the former has 16 reference translations available for English and the latter has 7, allowing for a somehow more interpretable measure of performance for language pairs with English as the target language.

Again, English and French texts are normalized and tokenized by LIMSI in-house tools. Chinese texts are segmented by the Stanford CRF-based Chinese word segmenter[19].

Basic alignment task

This experiment aims to assess the quality of the sub-sentential alignment generated by our method on an entire bilingual parallel corpus. We used the mgiza++ aligner as a baseline with default settings: 5 iterations of IBM1, HMM, IBM3, and IBM4, in both directions (source to target and target to source). The alignments in two directions are then combined using the grow-diag-final combination heuristic of moses. As for our on-demand word alignment method, its alignment quality depends on the number of sub-corpora (N) that are drawn for each sentence pair. This value enables to perform some tradeoff between alignment quality and speed.

Experimental results on the News Commentary and the EMEA corpora in both translation directions are given in Table 4.6, where owa-N represents the owa procedure applied with N sub-corpora for each sentence pair. Looking at the English to French translation results, on the News Commentary corpus, we find that owa-1000 performs a little worse than mgiza++ (-0.4 BP, $+0.5$ TP). By decreasing N to 500, we find no significant change in the results. However, when N is decreased to 100, performance drops significantly (-1.1 BP, $+1.3$ TP).

On the EMEA corpus, the same alignment is tested on two different data sets, yielding quite different results. First, among owa systems, owa-1000 always yields the best result. Reducing the value of N degrades the performance of all systems. Compared to mgiza++, owa performs better on the Cochrane task ($+0.5$ BP), while it performs worse on the WMT'14-med task (-0.4 BP).

[19]http://nlp.stanford.edu/software/segmenter.shtml

	News Commentary		EMEA			
	Newstest		WMT'14-med		Cochrane	
	BLEU	**TER**	**BLEU**	**TER**	**BLEU**	**TER**
	English → French					
mgiza++	31.62±0.10	50.22±0.09	34.36±0.09	48.48±0.02	30.06±0.13	52.78±0.17
owa-1000	31.20±0.05*	50.71±0.06	33.94±0.05*	49.19±0.09	30.54±0.03*	52.74±0.28
owa-500	31.25±0.10	50.79±0.05	33.72±0.03*	49.37±0.03	30.31±0.07	53.38±0.17
owa-100	30.50±0.04*	51.56±0.11	33.34±0.07*	49.82±0.05	29.86±0.02	53.35±0.58
	French → English					
mgiza++	33.04±0.06	47.71±0.07	33.28±0.01	45.94±0.06	32.61±0.04	46.80±0.04
owa-1000	32.49±0.05*	48.63±0.10	32.91±0.08	46.39±0.03	33.54±0.02*	46.11±0.08
owa-500	32.20±0.06*	49.03±0.13	32.64±0.09*	46.45±0.06	33.94±0.02*	45.87±0.03
owa-100	32.05±0.03*	48.82±0.08	32.04±0.07*	47.20±0.04	33.28±0.08*	46.94±0.07

Table 4.6 – Performance (using mean ± standard deviation of 3 runs, * significant at $p < 0.01$ level) of the owa word alignment method.

Freq. interval	development set		test set	
	WMT'14-med	Cochrane	WMT'14-med	Cochrane
0 (OOV)	3.59 %	4.95 %	3.45 %	3.89 %
1-10	3.55 %	**4.43** %	4.02 %	**5.75** %
11-50	5.62 %	**5.67** %	5.79 %	**5.87** %
51-100	3.73 %	**3.99** %	4.06 %	4.05 %

Table 4.7 – Percentage of tokens in selected frequency intervals.

In the other translation direction, a similar observation can be made. On the News Commentary corpus, owa-1000 underperforms mgiza++ (-0.5 BP). On the EMEA corpus, it outperforms mgiza++ on the Cochrane translation task and performs a little worse than mgiza++ on the WMT'14-med translation task.

An interesting result is that our alignment method performs better than mgiza++ on Cochrane while it performs worse than mgiza++ on WMT'14-med.[20] As mentioned before, owa is an extension of Anymalign. **?** have shown that Anymalign has better performance than mgiza++ on rare words. Hence, we hypothesize that this difference on the result of the two translation tasks is related to the proportion of rare words (and of phrases containing them) in the development and test data sets. Table 4.7 shows the percentage of tokens in the development and test data sets of the two translation tasks for several frequency intervals. Only tokens whose frequency in the EMEA corpus in the intervals: $[1, 10]$, $[11, 50]$ and $[51, 100]$ are considered as *rare* and counted in Table 4.7. We can see that both the development and test set of the Cochrane task contain higher percentages of rare tokens than the WMT'14-med sets. Therefore, the better performance of owa on the Cochrane task could be the result of its better alignment of rare words and its cascading effects.

Turning to a more interesting situation, we now consider our tri-lingual HIT corpus in which Chinese-English is a much more difficult language pair than French-English. The Chinese language is from a very different language family to English and French. In addition, multiple translation references are available for English texts, which makes the evaluation more reliable [**?**]. We performed the same experiments as before on the HIT corpus to further validate

[20]Note that we used the same owa and mgiza++ alignments for both translation tasks.

	HIT					
	En2Fr (1 ref)		Fr2En (7 refs)		Zh2En (7refs)	
	BLEU	**TER**	**BLEU**	**TER**	**BLEU**	**TER**
`mgiza++`	39.65±0.54	44.50±0.60	45.52±0.23	33.99±0.20	27.88±0.17	50.76±0.13
`owa-1000`	39.70±0.34	43.56±0.32	45.34±0.11	33.79±0.08	27.85±0.49	50.93±0.27

Table 4.8 – Performance (using mean ± standard deviation of 3 runs) of the `owa` word alignment method on `HIT` corpus.

the performance of our alignment method. The experiments were conducted in three translation directions: English-to-French, our main translation direction, French-to-English, using multiple translation references, and Chinese-to-English, a comparatively more difficult language pair with also multiple translation references. We only used `owa-1000` in all these experiments, as our previous experiments showed the superior performance of larger values for N.

Experimental results on the `HIT` corpus are presented in Table 4.8. In this experiment, `owa-1000` and `mgiza++` have very close performance. On the English to French task, `owa-1000` is slightly better than `mgiza++` on BLEU, while on TER metric `owa-1000` outperforms `mgiza++` by 1 TP. On the French to English and Chinese to English tasks, the two alignment approaches have very close performance (recall that evaluation is based on the use of 7 reference translations).

Incremental alignment task

We have just shown that our approach performs on par with the `mgiza++` baseline on the studied configurations for full corpus alignment. We now turn to the issue of aligning new data, which in many situations can only be performed incrementally. Indeed, considering that all input sentences in our test set could be translated independently over a large interval of time, it would certainly not be conceivable, time-wise and computation-wise, to perform a full static alignment of the iteratively growing bilingual corpus. We will nevertheless report evaluation results for this situation below.

Few previous works have considered the task of incremental alignment of parallel corpora [**??**]. In these works, newly available data are non-selectively aligned and incorporated into the existing system. In our experiment, we will concentrate on a very specific use of additional data: sentences will be pooled from an held-out parallel corpus on the basis that they contain at least one occurrence of a word that is OOV in the baseline parallel corpus[21]. In order to study a condition where significant numbers of such OOVs exist, we used the `HIT` corpus as our main corpus (French-to-English translation direction), relatively to which our test set contains 79 unique OOVs (436 occurrences). Our additional training data (`WMT`) provided matches for 65 of them. We retrieved a maximum of 100 sentence pairs for each of these 65 OOVs, which yielded an additional parallel corpus of $3,355$ sentence pairs (`supp` in Table 4.4).

We now describe the configurations that will be compared. A main table will be used for all configurations, corresponding either to the `mgiza++` baseline or to our on-demand word alignment approach. A supplementary table will be built from `supp` by various means:

[21]Meaning that the word was not present in the original training data, not that no translation for it could be extracted by some technique.

Phrase tables			HIT					
main	supplementary							
(62K HIT)	(3.3K supp)	# entries	BLEU	1g	2g	3g	4g	TER
French→English (7 refs)								
mgiza++	none	-	45.52	76.5	52.2	37.8	27.1	33.99
|	forced	59	47.94	76.8	55.4	41.0	29.2	34.62
|	concat	60	48.69	78.4	56.1	41.4	29.8	33.09
|	owa	64	49.83	80.9	57.3	42.0	30.5	30.61
|	concat++	62	50.23	81.5	57.8	42.6	31.1	29.81
owa	none	-	45.34	77.0	52.1	37.4	26.9	33.79
|	owa	64	50.45	81.8	58.3	42.5	30.9	29.94

Table 4.9 – Results of experiments where a supplementary corpus is pooled and aligned by several methods.

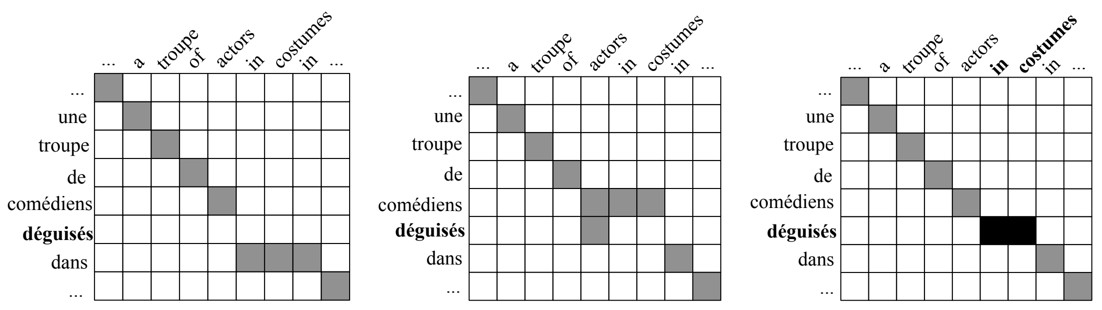
- forced alignment (see Section 3.2.1) on `supp` using the statistical models (previously) obtained on `HIT` (`forced`);
- statistical alignment on the concatenation `HIT+supp`, and extraction of the alignments on `supp` only (`concat`);
- sampling-based alignment on `supp`, sampling from the union of `HIT` and `supp` (`owa`);
- statistical alignment trained on the whole `WMT` corpus, and extraction of the alignments on `supp` only (`concat++`).

As said previously, the `concat` variants cannot be considered as practical solutions for the problem at hand. Once alignments are obtained for the `supp` corpus, a separate phrase table is used by the `moses` tools as previously, where our additional table is used as *back-off*, for unigrams only. Therefore, our additional training data, once aligned, will only be used in practice for proposing translations for previously unknown words. Note that, in this experiment, we do not extract the information needed to update the lexicalized reordering models used by `moses`.

Results for this set of experiments are given in Table 4.9. Using `mgiza++` for building the main translation table, we find a very clear ranking for all the studied strategies: `concat++` > `owa` > `concat` > `forced` > `none`. The only approach that outperforms ours (by an average of +0.4 BP) is the statistical alignment technique using more than 16.6M sentence pairs.[22] `owa` outperforms `concat` (by an average of +1.14 BP) and `forced` (by an average of +1.89 BP), the latter being the most practical baseline to consider. Significant improvements can be observed on 1-gram precision, which percolate nicely to higher-order n-grams.

Interestingly, we managed to improve this result further by using also our on-demand word alignment technique for aligning the main parallel corpus (average of $+0.62$ BP), which furthermore happens to be even slightly superior to `concat++` (average of $+0.22$ BP, with small improvements on 1-gram and 2-gram precisions). To explain this fact, we performed oracle decoding on the test set using a greedy, approximate local search strategy and a number of phrase-based operators [**?**] to get some account of the best translation score attainable given the corresponding phrase table. We found that one-best translation was slightly superior for the

[22]As presented in Section 4.4.3, the alignment process on the `WMT` corpus took $1,006$ h using modern computing resources.


... a troupe of actors in costumes in ...
... une troupe de comédiens **déguisés** dans ...
... a troupe of actors **in** **costumes** in ...

(a) `mgiza++ forced` (b) `mgiza++ concat` (c) `owa`

Figure 4.7 – Example of matrices on French-English obtained using two `mgiza++` baselines and our on-demand word alignment (`owa`).

baseline (-0.18 BP, see Table 4.8), but that the oracle for our approach was superior ($+1.01$ BP[23]), indicating that our approach did extract more useful phrases, but which were apparently poorly scored. It seems that providing the decoder with translations for previously OOV words had an additional effect on the configuration where we use the phrase table obtained using our technique.

In summary, our on-demand word alignment approach can yield results that are quite comparable to the results of the current state-of-the-art approach as implemented in `mgiza++`. Its more apparent strength emerged when aligning new data containing highly useful words (words that were previously out-of-vocabulary in the available data). As mentioned before, we hypothesize that these improvements mainly stem from the improved alignment of rare words and its cascading effects. Figure 4.7 illustrates a case where the *rare* French word *déguisés* (here: *in costumes*) was only correctly aligned by our technique, and where the negative consequences for the two `mgiza++` baselines could be important (at least, for our experiments, no translation for *déguisés* alone could be extracted from this sentence pair by `mgiza++` here).

4.4.4 Validation of the framework

In the previous sections, we have validated both our on-the-fly model estimation and our on-demand word alignment methods. We now combine these two techniques to develop new on-demand SMT systems, and assess their performance on large-scale data.

We take the same vanilla `moses` system as in Section 4.4.2 as the baseline system, in which the word alignments are generated with `mgiza++` and the translation tables are extracted with the scripts of `moses` which estimate translation models using all examples. Using the computed `mgiza++` alignments, we also built phrase tables on-the-fly using the deterministic random sampling strategy (`rnd`). Since the processing time is also a factor being discussed in this work, we also built an alternative baseline system relying on `FastAlign` [?] instead of `mgiza++`; here again, the entire corpus is aligned in both directions (source to target and target to source) and symmetrized using the same heuristic as for `mgiza++`. Phrase tables are built on-demand using `rnd`.

[23]The oracle results on BLEU of `mgiza++` and `owa` in the French to English translation task are 68.58 and 69.59, respectively.

As for our system described in Section 4.3, no pre-computation save the construction of the source corpus suffix array is necessary. At running time, our systems take a particular input text, segment it into phrases, perform `rnd` to select translation examples, run `owa` to generate the alignments for selected sentences and extract translation models based on the selected examples and generated alignments. This procedure is sketched in Algorithm 3.

As for the sample size M, we use $M = 100$ for both the `Newstest` task and the `Cochrane` task and $M = 500$ for the `WMT'14-med` task, following the results obtained in Section 4.4.2. For all experiments involving `owa`, we used `owa-1000`.

Experimental results

Our results, reported in Table 4.10, show that `mgiza++` always performs better than the other two alignment approaches. `FastAlign` performs slightly below `mgiza++` on all tasks. `owa` is 1.9 BP behind the `mgiza++` and 1.2 BP behind `FastAlign` on the `Newstest` task. On the `WMT'14-med` task, the difference is slightly smaller, `owa` being 1.2 BP behind `mgiza++` and 0.8 BP behind `FastAlign`. However, on the `Cochrane` task, our alignment method `owa` is more competitive. Although it is 0.7 BP behind `mgiza++`, it is now slightly better than `FastAlign` (+0.2 BP).

Another informative aspect of our results concerns the processing time of each system. As mentioned before, `mgiza++` took $1,006$h user CPU time to align the entire parallel corpus while `FastAlign` only took 21h[24]. Processing time for `owa`, which only aligns the necessary sentence pairs, depends on the size of the input text. For example, in the `Cochrane` task, `owa` took 146h user CPU time (16h wall clock time) to construct the translation tables for the development and test data sets.

A first conclusion is that the `owa` alignment underperforms probabilistic alignment models in a standard MT scenario; this alignment strategy however strongly departs from its competitors, as each sentences pair is processed on a per-need basis, independently from the rest of the corpus. This property allows us to obtain dynamic rather than static systems, an important property as will be illustrated in the next chapters of this thesis. Since all information are computed on-demand when necessary, updating system is straightforward. Integrating new data into the system only involves re-constructing the suffix array which is much simpler and faster than retraining a system from scratch. Furthermore, this will be accomplished with much smaller resources (for instance, compressed tables occupy less than 1Mb to translate a single `Cochrane` document), which would make it possible to produce translations on very light devices.

4.5 Summary

In this chapter, we have presented an original approach for phrase-based SMT system development, which allows us to construct SMT system very quickly from scratch using very large-scale training data. The key element that permits this is the on-demand computation for translation information. Although experimental results have shown that our system slightly underperforms the state-of-the-art `mgiza++/moses` system, it has several important properties. Firstly, extraction of translation rules is performed on-demand. Unlike `moses` that pre-computes every-

[24]Since `FastAlign` does not support multi-threading, 21h is also its wall clock time.

Task	Configuration		Translation quality		ΔBLEU
	aligner	sampling	BLEU	TER	
Newstest	mgiza++	full	35.84±0.05	46.14±0.10	0.0
		rnd-100	35.64±0.15	46.31±0.08	-0.2
	FastAlign	rnd-100	34.92±0.09*	46.95±0.02	-0.7
	owa	rnd-100	33.87±0.01*	47.78±0.07	-1.9
WMT'14-med	mgiza++	full	40.84±0.11	42.23±0.09	0.0
		rnd-500	40.87±0.12	42.23±0.04	0.0
	FastAlign	rnd-500	40.48±0.08	42.64±0.05	-0.4
	owa	rnd-500	39.64±0.13*	43.96±0.02	-1.2
Cochrane	mgiza++	full	34.12±0.10	48.59±0.22	0.0
		rnd-100	34.08±0.05	48.48±0.27	0.0
	FastAlign	rnd-100	33.17±0.03*	49.43±0.29	-0.9
	owa	rnd-100	33.35±0.02*	49.62±0.05	-0.7

Table 4.10 – Experimental results for large-scale experiments on three MT tasks. (* significant at $p < 0.01$ level)

thing in advance, independently of the actual future use, our system extracts the minimal set of information required to translate each particular input text. This property eliminates the needs for *a posteriori* filtering of the translation tables.

Secondly, model estimation in our system is sampling-based, which means that it does not use the entire training data. This property makes model estimation much faster, especially for those frequent translation units. It also makes the translation tables more compact, which saves computational resources and accelerates the decoding process.

Thirdly, the word alignment method in our system processes each sentence pair independently, which makes our work significantly differ from previous works [**??**].

However, these properties are not fully taken advantage of in the standard MT experiments reported in this chapter. In the next chapter, we will present an empirical study of our system where these properties will be leveraged: we will show how our system can deliver good translation performance much faster than traditional systems.

5 Incremental Adaptation with On-demand SMT

In the previous chapter, we have presented a novel SMT framework for on-demand system development and have illustrated its performance *via* a number of control experiments. Although our approach has been shown to underperform the baseline in standard experimental scenarios, its on-demand character has many potential advantages that can be explored in more pragmatic scenarios. One of its main advantages lies in its capacity to build SMT systems on-demand and to subsequently update them dynamically. By contrast, the state-of-the-art system development approach pre-computes all information in advance and the resulting systems are then static in two aspects.

Firstly, the trained translation models are static. In a state-of-the-art system, all models are extracted from a pre-defined parallel corpus, and are then used to translate any type of input text. However, as argued before, new data are constantly made available, and the state-of-the-art SMT approaches cannot seamlessly take advantage of them to improve their performance. Incorporating newly available data can help to improve not only the n-gram coverage but also the model estimation accuracy of the existing system. Additionally, in some contexts, more recent training data is more adapted to build systems than older data [**?**]. These observations provide motivation for incorporating newly available data into existing systems, in particular when the new data is known to be directly relevant to the application documents.

Secondly, a strong requirement of the state-of-the-art framework is the availability of a development corpus. Such a corpus is assumed to be representative of the test distribution, so that parameters optimized on the development corpus are also optimal for the test data. However, this assumption is usually too simplistic for real world data. Even for data from the same origin, the statistical distribution of words changes over time. Furthermore, a development set would be necessary for each type of documents that the system would be asked to translate, greatly limiting the usefulness of a single data set.

The two above reasons lead us to devise ways to incrementally develop on-demand systems and to update their parameters incrementally. In this chapter, we have chosen to illustrate two more favorable use cases of our framework in order to demonstrate its capabilities and flexibility. In Section 5.1, we will use our system in a translation for communities task, where documents to be translated are from the same origin, to show its ability to quickly adapt to a specific domain and to outperform a competitive baseline. In Section 5.2, another even more difficult use case, which we called *any-text* translation, will be studied.

5.1 Translation for communities with on-demand SMT

In the translation for communities task, we make two important assumptions: the first one is that it can be desirable to provide automatic translations early, even before any human translation has been performed, to handle *documents of unknown origin so far* (as is the case when a new application domain is considered); the second one is that there exist some clear relation between consecutive application documents, so that their set of optimal parameters are close to one another. A consequence of these assumptions is that a classical development set will not be needed anymore, a significant economy in practice. Nonetheless, our proposal only makes sense if it also compares favorably in terms of translation evaluation to a standard system making use of a development set.

We consider a situation where a stream of documents needs to be translated. After each document has been automatically translated, we also make the plausible assumption that it is post-edited by a human translator, thus providing new data that can be used to update both the models and parameters of the systems before translating the next document. This situation will be illustrated using the concrete `Cochrane` situation. In our experiments, we will take the 100 documents in the test set (see Table 4.2) to simulate the document stream[1].

In the rest of this section, we will describe a series of increasingly richer configurations to show the ability of incremental training of our on-demand SMT system. For the parameters of our systems (the sampling size (M) of `rnd` and the number of sub-corpora (N) processed for each sentence pair by `owa`), we use in the sequel the same parameters as for the `Cochrane` experiments in Section 4.4.4, setting $M = 100$ and $N = 1000$.

5.1.1 No cache, no tuning (`Config0`)

The first, basic configuration, processes input documents independently in sequence as described in Algorithm 3. Each document-specific translation table is fed to the decoder[2], which uses the default values for model parameters. In this configuration, no tuning is actually performed, which eliminates completely the need for a development corpus, and allows us to obtain translations of documents almost instantly. Results for this untuned configuration (see `Config0` in Table 5.1) are lower by 5.8 BP than those of the conventionally tuned `moses` system, which can be mostly attributed to the absence of tuning. However, translations for the

[1] Since we do not have the test data over a large time scale, here we simulate the document stream only for illustrating the incremental training ability of our framework. The effects of *recency* of the data will not be discussed in this manuscript.

[2] We used the `moses` decoder in our experiments, whose default parameters are: 0.3 for all 7 reordering features, including 6 lexical reordering feature and 1 distance-based reordering feature; 0.2 for all 5 translation features; 0.5 for the language model and -1 for the word penalty.

Configs	Translation quality		PT construction time	
	BLEU	TER	user CPU	wall clock
moses	34.12	48.59	1,212h	252h
Config0	28.24	49.92	363h	27h
Config1	28.87	49.40	111h	10h
Config2	28.58	49.54	76h	7h
+spec	32.33	46.42	76.5h	7h
+online	36.41	46.44	76.5h	7h
+dev	36.20	46.10	148.5h	+7h

Table 5.1 – Results for the owa system on a large-scale English-to-French translation task. The reference system is the vanilla moses system (cf. Table 4.10), tuned on the Cochrane development set. +dev is identical as Config2+spec+online, with initial weights tuned on the Cochrane development set.

test set are delivered much faster, with a x9.3 times wall clock speed-up compared to moses.

5.1.2 Using a cache (Config1)

In Config0, the system processes each document in isolation, so it has to repeatedly reestimate translation probabilities for those frequent phrases $\bar{s}$ that occur in multiple documents. Figure 5.1 shows the percentage of n-grams occurring in document $\mathbf{d}_t$ that were also seen in the previous $t-1$ documents $\{\mathbf{d}_i, i = 1 \ldots t-1\}$. For instance, for $t > 20$, more than 80% of the unigrams (and 40% of the bigrams) were found in the previous documents. To take advantage of the fact that many phrases occurring in the document sequence occur more than once, our system uses a *caching mechanism* to save previously computed alignments, meaning that it is no longer necessary to recompute alignments for sentences that were selected in the past. Note that, since the random sampling strategy rnd is deterministic, this has no bearing on the samples used to estimate phrase probabilities for phrases seen in previous documents: the system always selects the same sample for any given phrase $\bar{s}$.

More formally, denoting $\mathbf{K}$ the cache of past alignments when translating document $\mathbf{d}_t$, the cache-based procedure for Config1 is described in Algorithm 4.

Algorithm 4 Estimation procedure for Config1

```
parallel train corpus $\mathbf{C}$, alignment cache $\mathbf{K}$
given an input document $\mathbf{d}_t$, sample size $M$
compute $\Sigma[\mathbf{d}_t]$
for all $\bar{s} \in \Sigma[\mathbf{d}_t]$ do
    $\mathbf{S}[\bar{s}] = \texttt{rnd}(M, \mathbf{C}, \bar{s})$        // Sampling
    $\mathbf{K} = \mathbf{K} \cup \texttt{owa}(\mathbf{S}[\bar{s}] \setminus \mathbf{K})$ // Aligning new selected sentences
    $\texttt{estimate}(\boldsymbol{\theta}_{\bar{s}}, \mathbf{S}[\bar{s}], \mathbf{K})$  // Estimation
end for
```

Experimental results (see Config1 in Table 5.1) show the effectiveness of this approach, with a x2.7 speed-up (wall clock) as compared to Config0, or a x25.2 speed-up as compared to moses. Interestingly, using the cache yields small improvements in both the BLEU score (+0.6) and the TER score (-0.5). Since the sampling strategy is deterministic, each source phrase

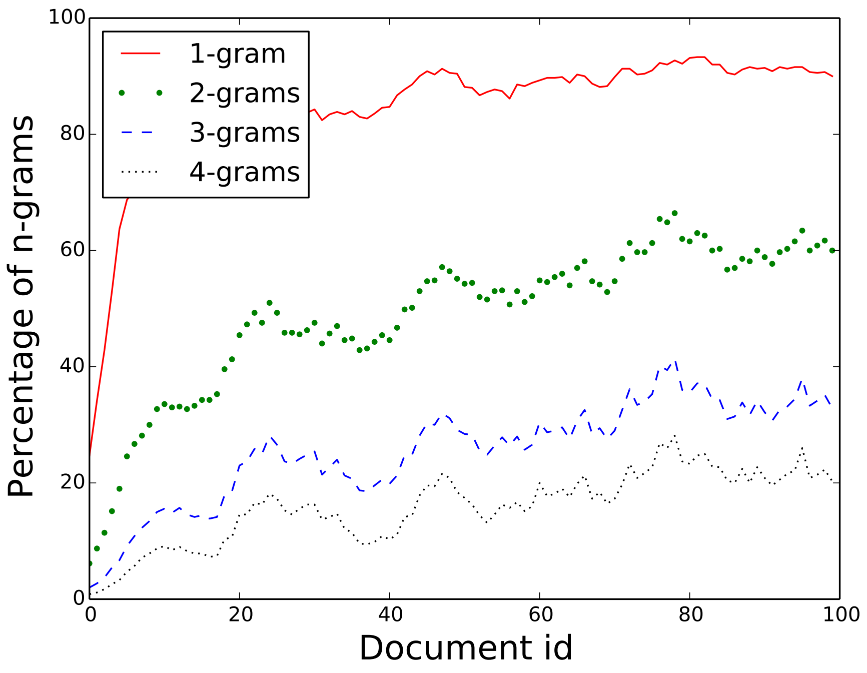


Figure 5.1 – Percentage of n-grams found in previous documents in the `Cochrane` document sequence.

is estimated based on the same examples, there should be no difference on the translation models. However, the lexical weighting models are quite different in these two configurations. These models are estimated from the lexical translation probability derived from the word alignments. In `Config0`, each document is processed in isolation, so there are few alignments (in average $43\ 000$ aligned sentences for each single document) to estimate the lexical probabilities. In `Config1`, in contrast, all computed alignments are saved in the cache. With a growing cache, the more word alignments there are, the better the lexical translation probability estimates become. The growth of the number of aligned sentences in the `Config1` cache is displayed in Figure 5.2 using a dashed line. This confirms that there are much more alignments available for the last documents than for the first ones, and that these are succesfully used to better estimate the lexical models.

5.1.3 Sampling by reusing aligned sentences (`Config2`)

In [**??**], sampling is performed independently for each phrase. This was rightly not described as a shortcoming, because word alignment, by far more time-consuming than translation extraction, was already available for all the training sentences. In our situation, the number of sentences to align has a direct impact on processing time. Taking better advantage of the alignment cache, we can further reduce the number of sentences that must be aligned by resorting to a computationally simple strategy (see Algorithm 5, where $\mathbf{K}[\bar{s}]$ denotes the set of cached sentences containing phrase $\bar{s}$). The alignment cache is used to bias sampling to prefer using the already aligned sentences. When performing sampling for a *new* phrase, we first draw examples for this phrase from the cached sentences, and only if necessary take a complementary sample using unaligned sentences. In practice, we perform `rnd` on the cached sentences and then `rnd`

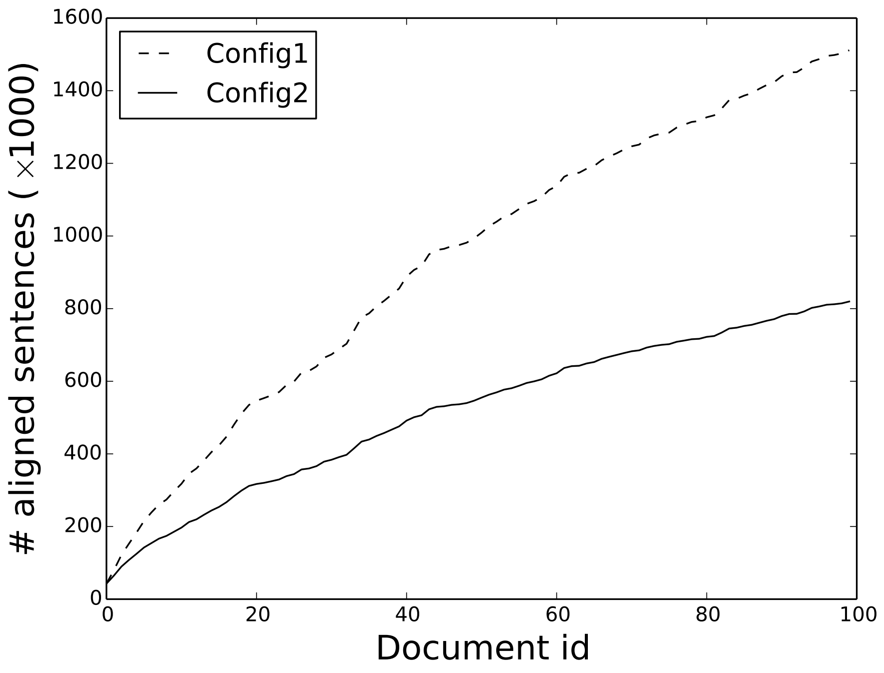


Figure 5.2 – Evolution of the cache size for a sequence of `Cochrane` documents.

on the remainder of **C**.

Algorithm 5 Estimation procedure for `Config2`

```
parallel train corpus $\mathbf{C}$, alignment cache $\mathbf{K}$
given an input document $\mathbf{d}_t$, sample size $M$
compute $\mathbf{\Sigma}[\mathbf{d}_t]$
for all $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}_t]$ do
    $\mathbf{S}[\bar{s}] = \texttt{rnd}(M, \mathbf{K}[\bar{s}], \bar{s}) \cup \texttt{rnd}(M - |\mathbf{K}[\bar{s}]|, \mathbf{C} \setminus \mathbf{K}, \bar{s})$
    $\mathbf{K} = \mathbf{K} \cup \texttt{owa}(\mathbf{S}[\bar{s}] \setminus \mathbf{K})$
    $\texttt{estimate}(\boldsymbol{\theta}_{\bar{s}}, \mathbf{S}[\bar{s}], \mathbf{K})$
end for
```

Experimental results (see `Config2` in Table 5.1) reveal that an additional x1.4 speed-up (wall clock time) is obtained as compared to `Config1` (or x36 as compared to `moses`), at the cost of a slight decrease in translation performance (-0.3 BP), which may be attributed to the smaller number of aligned sentences, resulting in poorer lexical models. The growth of the number of aligned sentences in `Config2` is shown in Figure 5.2.

To further analyze the effect of the cache, Figure 5.3 shows how the average per token processing time decreases as more and more documents from the same stream are translated. At the outset, estimation time per token decreases quickly as a result of the use of the cache; as more and more documents are translated, average estimation time continues to decrease, albeit at a slower pace.

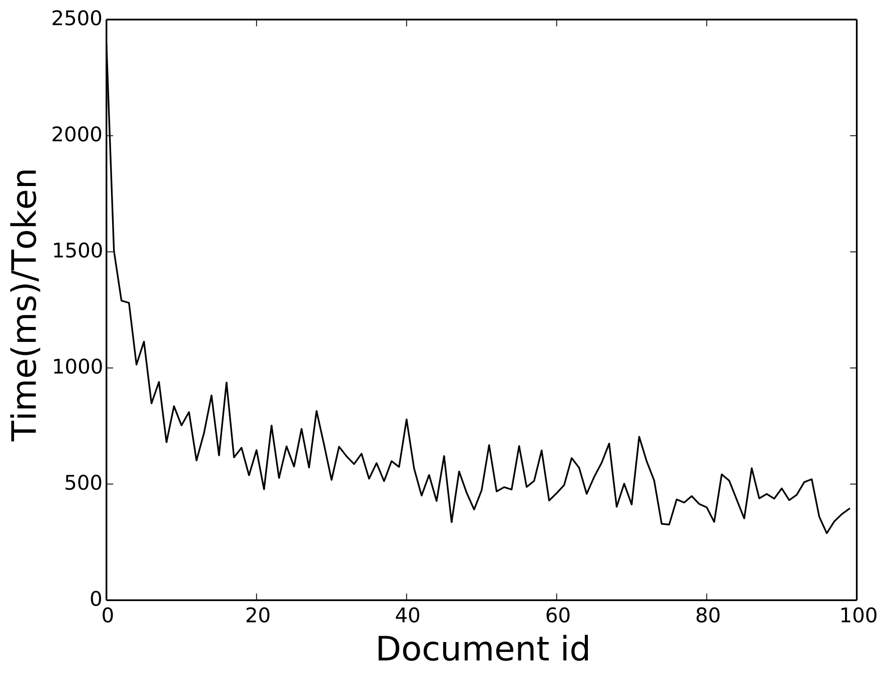


Figure 5.3 – Evolution of the average per token processing time for a sequence of documents.

5.1.4 Plug-and-play data integration (+`spec`)

We now consider the following *incremental training* regime: after each individual document is translated, the post-edited version of the document becomes available.[3] Our on-demand framework makes it natural and straightforward to readily integrate any newly available parallel data without any retraining.

In the following experiment, each newly available `Cochrane` document is aligned using `owa` and then added to a "specialized" corpus, denoted by `spec`. A separate phrase table is estimated from `spec`; considering the very small size of our specialized source (less than 1.8K sentence pairs), the corresponding phrase table, built from previous documents in the sequence $\{\mathbf{d}_i, i = 1 \dots t-1\}$, contains only two scores per phrase pair: the direct translation model score and the phrase penalty. As we still assume that no development set is available, the parameters for the new models are simply copied from the main table. Note that in this setting, the `spec` phrase table is used as a *back-off* table to the phrase table estimated from the main, static corpus. While this may seem counter-intuitive, we did this primarily because the `spec` translation model is comparatively poorly estimated, because of the small quantity of data used. However, for those domain-specific terms, phraseology or long phrases which usually only exist in the in-domain data, we could use the `spec` phrase table to translate them.

Results in Table 5.1 show that the additional table (+`spec`) helps to significantly improve translation quality over the raw `Config2` configuration (+3.7 BP), for a modest additional processing time of half an hour for aligning the content of the first 99 documents. Since the `spec` table for document $\mathbf{d}_t$ is estimated based on the previous $t-1$ documents, the quality of

[3]Actually, the `Cochrane` data used here is not a completely post-edited corpus: a large portion of the data was translated by human translator from scratch. We still use this data as a post-edited corpus in our experiments, although these two kinds of data is different. We believe this does not affect the experimental conclusions.

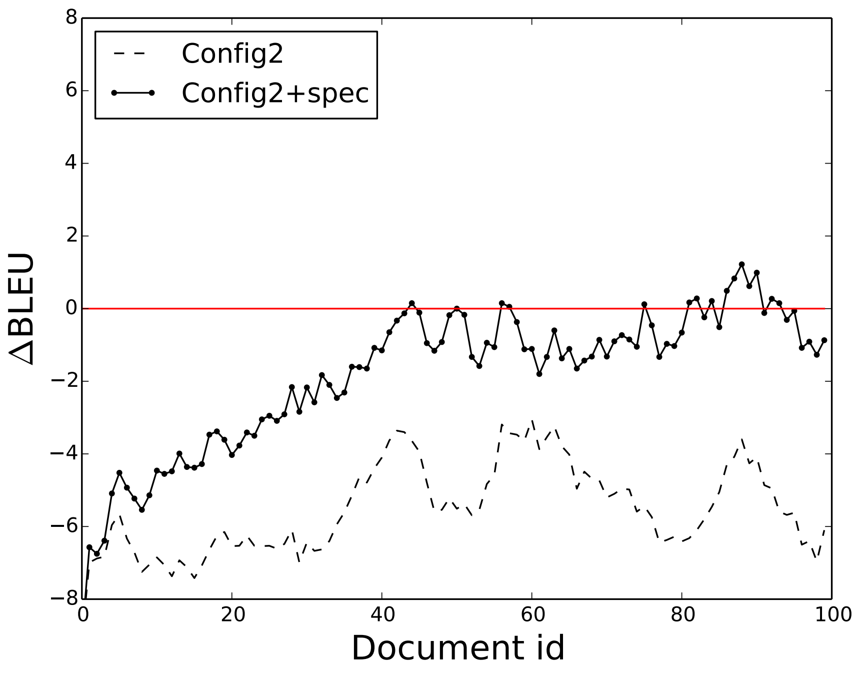


Figure 5.4 – Document-level comparison with the vanilla `moses` system. The y-axis represents the difference in BLEU score (ΔBLEU) between our systems and the vanilla `moses` system for each document in the sequence.

the phrase table improves over time.

Figure 5.4 shows the document-level comparison between our systems (`Config2` and `Config2+spec`) and the vanilla `moses` system, where the curves represents the difference of performance (evaluated by BLEU) between `moses` and the corresponding system on each document in the stream. The parts above the horizontal line means the corresponding system is better than `moses`; otherwise, the corresponding system is worse. We first observe that the document-level gap between `Config2` and `Config2+spec` is much larger (around 5 BP) at the end of the document sequence than at the start, confirming that the quality of the `spec` phrase table improves over time. We also see that `Config2` systematically underperforms `moses` on all documents, which was expected given the gap in corpus-level performance. Interestingly, the use of the specialized phrase table, `Config2+spec`, yields fast improvements and matches the performance of `moses` after about 40 documents have been translated. We can conclude that the integration of such a specialized corpus allows our system to achieve nearly the same performance as the vanilla `moses` system but delivering translations a lot faster. Furthermore, these results are obtained without using a development set, a significant economy both in human translation time and in system development time. Although the obtained results strongly depend on the nature of the data used, the *plug-and-play* data integration feature of our framework is very useful to improve the translation performance when translating streams of related documents.

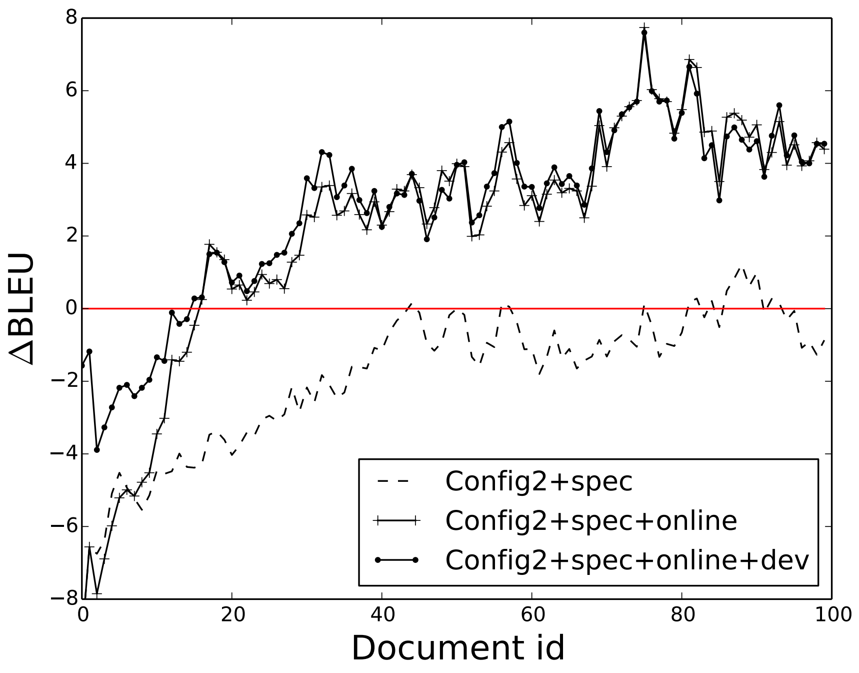


Figure 5.5 – Document-level comparison with the vanilla `moses` system with online tuning. Initialization either uses `moses` default values (`+spec+online`), or parameters tuned on a development set (`+dev`).

5.1.5 Simple online tuning (+`online`)

We have previously demonstrated that our on-demand framework allows us to seamlessly integrate newly available data, yielding systems that match a `moses` system trained in a conventional way after just 40 documents of our specific data source. Remarkably, these results were obtained *without any parameters tuning*. We now consider a simple online tuning strategy to further explore the potential of on-demand system development. In practice, the system's weights are retuned after each document has been translated (and post-edited) as follows. Taking the previous weights as the initial point, we run the parameter tuning process on the just translated and post-edited document; the resulting parameter values are then averaged with the parameter values of the previous documents, and then used for translating the next document.

As mentioned before, the quality of the `spec` phrase table improves over time. The parameters tuned on the document at the beginning of the sequence are consequently not optimal to translate documents at the end of the sequence, because the `spec` phrase table becomes more and more important in translation. Instead of averaging the parameter values of all previous documents, we only average the parameter values of the most recent documents, because we assume that the `spec` phrase tables of recent documents are of a similar quality to the current one. In our experiments, such average parameter values are computed based on the previous 10 documents. As for the tuning algorithm, considering the number of features used in the system, we chose `KBMIRA` in our experiments. Additionally, we also allow here the `spec` phrase table to compete with the phrase table estimated from the static corpus [**?**] instead of having the latter take precedence.

Results for this last configuration are given in Table 5.1 (`+online`). Our simple online

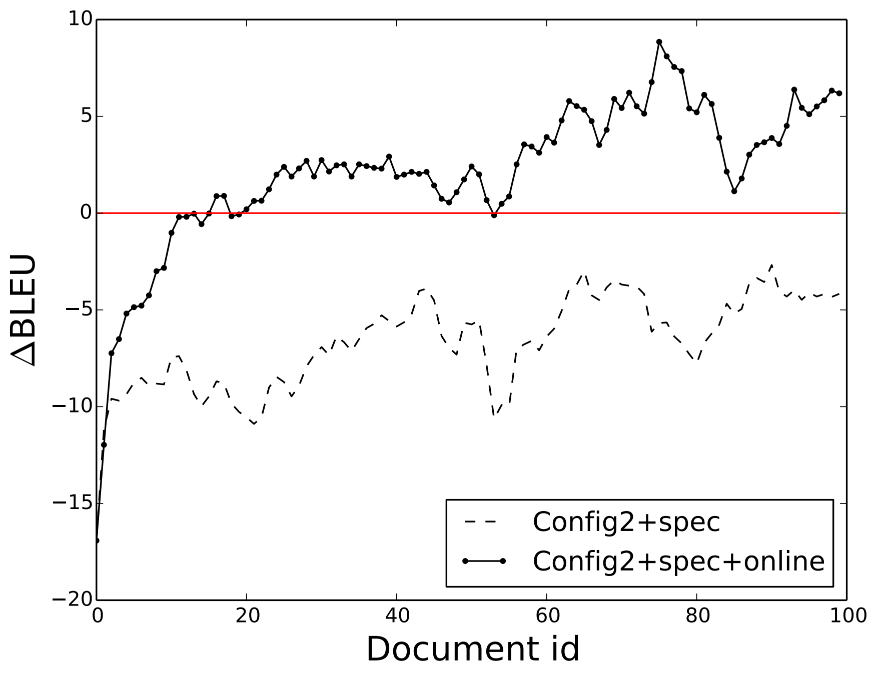


Figure 5.6 – Document-level comparison with the vanilla `moses` system with online tuning. Initialization either uses `moses` default values (`+spec+online`), or parameters tuned on a development set (`+dev`).

tuning yields a significant improvement (+4.1 BP) over the untuned `Config2+spec` configuration. Although the two configurations cannot be directly compared at the corpus-level, since our system integrates a growing set of in-domain data, while `moses` on its part greatly benefits from the in-domain development data, we still note that our framework now outperforms the `moses` baseline (+2.3 BP). More interestingly, comparison at the document-level (see Figure 5.5) demonstrates the strong potential of our framework: `moses` is systematically outperformed after fewer than 20 documents are translated. As for processing time, documents being very small, online tuning only has a small impact on the average processing time (3mn (wall clock time) on average for each document in this experiment)[4].

Our final experiment in the translation for communities scenario is designed to analyze the performance of our last configuration if it starts with conventionally tuned initial parameters. We thus first tuned the system on the development set, and then used the tuned parameters to initialize the starting parameters of this new configuration. The result is reported in Table 5.1 (`+dev`): using tuned parameters to initialize the system yields no significant change on translation quality. Comparing to the `Config2+spec+online` system, BLEU is worst by 0.2 points but TER is better by 0.3 points. The document-level comparison in Figure 5.5 shows, as expected, that initializing with parameters tuned on the development set yields better performance than `Config2+spec+online` at the beginning of the document sequence. However, after fewer than 20 documents have been processed, there is no visible difference between the two systems. We can thus conclude that the online tuning strategy implemented in our framework allows us to effectively dispense with the use of a development set.

[4]A control experiment using online tuning with one single (main) phrase table delivered performance on par with the `+spec` configuration (BLEU=32.30, TER=48.20)

We also performed these experiments on the French-to-English translation direction, and the document-level results are shown in Figure 5.6. First, for the `Config2+spec` system, we can observe that the performance of the system improves with the number of translated documents, although it is not as significant as the improvement observed in the English-to-French translation direction (as shown in Figure 5.5). This is probably related to the diversity of the language, where the 100 Cochrane bilingual documents contain 3 854 different English words and 4 398 different French words. Such larger vocabulary leads to low repetitiveness of the text, which makes the `+spec` less beneficial. By applying the online tuning, the system (`Config2+spec+online`) is improved very fast and outperforms again the `moses` baseline after fewer than 20 documents are translated.

A number of translation examples for our translation for communities experiments are provided in Appendix D.

5.2 Any-text translation with on-demand SMT

In the previous section, we have demonstrated that our framework could be used in a translation for communities scenario, where a stream of documents from the same domain needs to be translated. However, in a more realistic situation, for example a web-based MT service, the incoming documents could be of any domain. Maintaining a MT system for each domain is computationally prohibitive. Previous works on multiple domain translation [**??**] have resorted to domain identification. In these works, a sub-model is trained based on each domain-specific sub-corpus, and system parameters are also separately optimized based on the development set of each sub-domain. When translating, each input sentence or document is assigned to its corresponding sub-system based on domain identification. This approach is limited by the availability of domain specific corpora and development sets and lacks the ability to anticipate incoming data of new domains.

In this section, we consider a scenario that has been so far comparatively less studied while being more realistic, where the characteristics of the input text are completely unknown before translation. We thus make the following assumptions:

- Training data was collected opportunistically and no specific document metadata (e.g. genre, document boundaries) are available for the full data set.
- The input text corresponds to a coherent discourse (i.e. is not made by concatenating unrelated documents).
- The text can be from any arbitrary domain, which precludes any off-line adaptation using a predefined specific bilingual corpora; therefore, the only in-domain corpus is the input text itself.
- No adapted development set is available, which precludes the use of tuning techniques relying on a development corpus from the same data source or domain.

Since the input text is completely unknown and could be from any domain, we dub this translation scenario *any-text translation*.

Documents	# lines	# En tokens	# Fr tokens	Domains
`talk1`	232	4.2 K	4.3 K	TedTalk
`talk2`	249	5.2 K	5.9 K	TedTalk
`book1`	1093	22.5 K	23.8 K	Literature
`book2`	1604	35.1 K	37.8 K	Literature
`book3`	960	25.3 K	22.7 K	Literature
`subtitle1`	495	5.0 K	5.6 K	Open subtitle
`subtitle2`	528	4.8 K	5.2 K	Open subtitle
`php`	1000	11.6 K	12.5 K	Technical manual
`kdedoc`	995	11.8K	12.5 K	Technical manual

Table 5.2 – Description of the documents from various domains.

We chose 9 documents from various domains, as shown in Table 5.2: two entire transcriptions of TED[5] Talks, three translated books, two movie subtitles and two technical manuals.[6] We do not have the guarantee that there exists corresponding in-domain data in the system's training data for each domain of the input text. Each document is translated independently, sentence by sentence. Translation rules are extracted on-demand from the training corpus for each sentence using an adapted version of Algorithm 4, where each sentence is treated as a single document.

We also make the same assumption as in Section 5.1 that after each sentence has been automatically translated, it is post-edited by a human translator. These translated and post-edited data can be used to update both the models and parameters of the systems for the next sentences. In this case, each sentence is translated with two phrase tables: one is estimated based on the training data of the system, the other is estimated based on the previously translated sentences in the same document (denoted by `indoc`).

Again, we chose the large-scale corpus `WMT` (see Table 4.2) as the training data and the vanilla `moses` system as our baseline. Since no development set is available in this experiment, as in Section 5.1, we chose to use the decoder's default parameters as initial parameters for decoding. As for the target language model, we used the news-domain LM as described in Section 4.4.1. Although this language model was optimized on news-domain data, it was trained on very large quantities of data and could be considered as a reasonable general-domain language model.

Experimental results are presented in Table 5.3, where `moses` is the baseline system, `on-demand` represents our on-demand SMT system, and `+indoc` also represents our on-demand SMT system but also using the `indoc` phrase table in decoding.

First, by comparing the results of `moses` and `on-demand` systems, we find `moses` is better than `on-demand` on all documents on BLEU. On TER that `moses` is also better than `on-demand` on most documents (5 out of 9 documents). We attribute this result to the quality of the word alignments, where, as shown in Section 4.4.4, our on-demand word alignment approach is a little worse than `mgiza++` on large-scale corpora. Second, by adding the `indoc` phrase table, our `on-demand` systems (`+indoc`) are generally improved, except on `talk2`, and they are better than `moses` on most documents (6 out of 9 documents). Apparently, such

[5]TED (Technology, Entertainment, Design) is a global set of conferences owned by the private non-profit Sapling Foundation, under the slogan: "Ideas Worth Spreading". URL: `http://www.ted.com`

[6]More information about these 9 documents are provided in Appendix C.

Documents	Baseline		Systems			
	moses		on-demand		+indoc	
	BLEU	TER	BLEU	TER	BLEU	TER
talk1	27.84	56.99	27.27	57.34	**28.30**	56.53
talk2	**30.96**	50.20	29.13	50.88	29.08	50.94
book1	15.29	68.64	14.87	67.93	**17.12**	65.56
book2	14.71	69.21	13.84	69.39	**14.75**	68.23
book3	12.56	79.99	12.29	79.06	**13.25**	77.12
subtitle1	**25.10**	56.44	24.25	55.69	24.41	55.30
subtitle2	**29.79**	49.85	29.05	49.96	29.72	49.60
php	17.42	66.24	16.43	67.38	**25.17**	60.96
kdedoc	11.02	82.09	10.08	80.16	**13.43**	77.47

Table 5.3 – Any-text machine translation results for English-to-French translation.

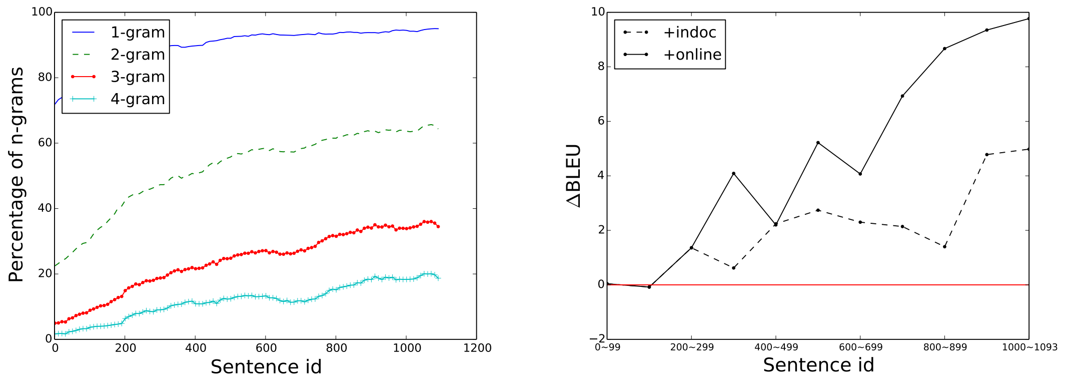
improvements depend on the repetitiveness and the length of documents. The more repetitive a document is and the longer it is, the more probable it becomes that the system could find useful information in previously translated sentences.

In this use case, it is also possible to perform parameter tuning during the translation of individual document. Unlike the situation in Section 5.1 where the translation unit was the document, here one sentence contains too little data to perform parameter tuning. Hence, instead, we chose to perform parameter tuning after each number of sentences are translated (100 in our experiments). In this experiment, sentences at the beginning of the document are always translated using the decoder's default parameters. After each group of 100 sentences have been translated, these 100 just translated sentences and their post-edited translations are used as a development set to tune the parameters with KBMIRA. The tuned parameters are then used to translate subsequent sentences. In order to show the effect of parameter tuning on the translation results, we only apply the tuning process to a few long documents ($>$ 1000 sentences): book1, book2 and php.

By applying parameter tuning after each group of 100 sentences on the +indoc system of book1, the translation result is further improved by $+2.4$ BP and -0.1 TP. On book2, the result is complex: the BLEU score is improved by $+0.3$ BP comparing to the +indoc system, but the TER score becomes worse by $+1.8$ TP. On the php document, a significant improvement from the +indoc is observed ($+9.2$ BP, -4.6 TP).

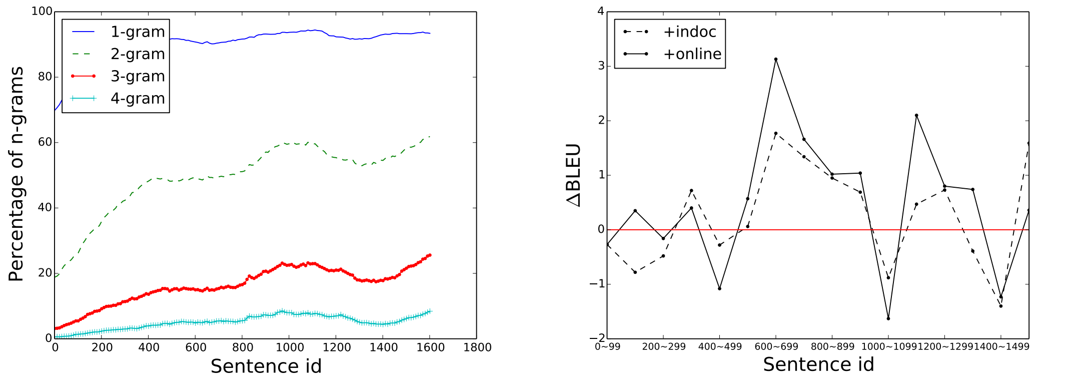
To better understand the behavior of our system, we also performed further analyses on these results. Figure 5.7a shows the percentage of n-grams occurring in sentence s_t that were also seen in the previous $t-1$ sentences $\{\mathrm{s}_i, i = 1 \ldots t-1\}$. For instance, for the sentences at the end of book1, about 20% of 4-grams (and nearly 40% of 3-grams) were found in the previous sentences of the document. Figure 5.7b shows the BLEU scores estimated on each group of 100 sentences. In the +indoc system, all sentences are decoded with the default parameters of moses, while in the +online system, the decoder parameters for each group of 100 sentences are tuned on the previous 100 sentences.[7] As shown in Figure 5.7b, the +indoc system takes advantage of the repetitiveness of the document and its performance is systematically better

[7]For example, the sentences 201 to 300 are decoded with the parameters which are tuned on the sentences 101 to 200.



(a) n-gram repetition (b) comparison with `moses`

Figure 5.7 – Experimental results on the `book1` document.



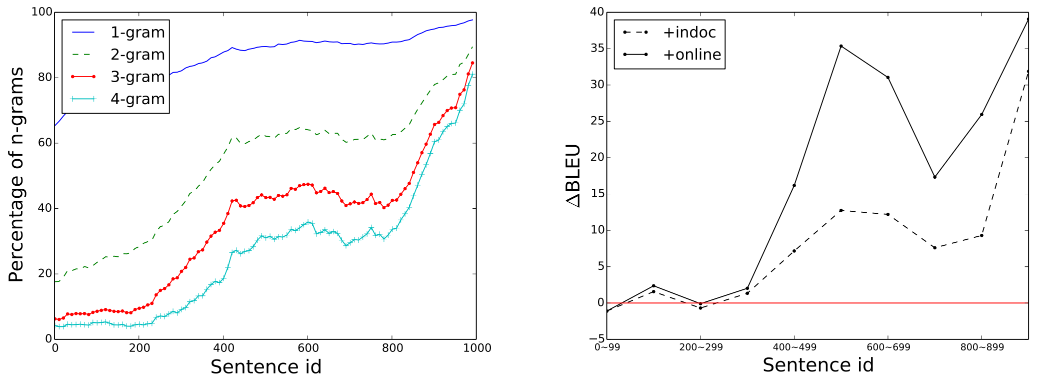
(a) n-gram repetition (b) comparison with `moses`

Figure 5.8 – Experimental results on the `book2` document.

than `moses` after translating 200 sentences. By applying parameter tuning on each group of 100 sentences, results are further improved, and to a larger extent (about 5 BP) at the end of the document.

Now turning to `book2`, we find that the results are very different than for `book1`. First, as shown in Figure 5.8a, the n-gram repetition rate is lower than that of `book1`, especially for 3-grams and 4-grams. For instance, for the sentences at the end of the document, less than 10% of 4-grams were found in the previous sentences of the document. The effect of the low repetitiveness of the document is also reflected on the corpus-level evaluation (see Table 5.3), where adding the `indoc` phrase table only improves performance by $+0.9$ BP, while on `book1`, the improvement is much larger ($+2.2$ BP). As shown in Figure 5.8b, parameter tuning could not always improve the translation performance (only in 11 out of 15 sentence groups), and sometimes the performance declines with the tuned parameters. This result may be related to the overfitting issues or the document itself.

Finally, on `php`, results are much clearer. As shown in Figure 5.9a, the `php` document has a very high repetition rate. The effect of such a high repetition rate is directly reflected on the translation results shown in Figure 5.9b, where the `+indoc` system improves very quickly



(a) n-gram repetition (b) comparison with `moses`

Figure 5.9 – Experimental results on the `php` document.

along with the number of translated sentences, and the improvement is very large. With tuned parameters, the system could better take advantage of the `indoc` phrase table, and the results are further improved.

As shown in these experiments, our framework can quickly construct SMT systems and incrementally adapt them to the target domain, even though the input texts are completely unknown. Its on-demand training character makes it possible to immediately produce translation output, even though the translation quality at the beginning is not very competitive. Also, its incremental adaptation scheme quickly improves its performance, especially on long and repetitive documents.

5.3 Summary

This chapter has addressed the issue of how the computationally expensive cost of the development of high-performance SMT systems, which typically exploit very large quantities of data, can be significantly reduced. By using our on-demand strategies, reductions of computation time to up to 36 times were obtained relative to a state-of-the-art system trained in a traditional fashion. Fast integration of newly available data in conjunction with online tuning allowed us to quickly reach the same performance as a strong baseline.

In addition to its implications as regards the development of SMT systems, our framework provides an innovative methodology that is also suitable for interactive MT: we measured wall clock times of less than 1 minute (*before any cache is available*) to build translation tables for individual sentences[8], making it practical to integrate system development within interactive human post-editing. And the experimental results provide empirical justification for online training and adaptation: users (e.g. human post-editors) can obtain better translations (according to BLEU) for most documents, and globally much faster, considering that no development set was ever required. Actually, this is also a significant advantage of our system, since preparing a development set for a specific translation task is also very expensive.[9]

[8]Note that this result could be largely improved with an optimized implementation or more powerful hardware.
[9]Considering that human translation speed depends on the difficulty of the text and characteristics of the trans-

We lastly want to underline that scenarios based on the `+spec` characteristic make simpler assumptions than traditional interactive MT (e.g. [**???**]), as parameter updates are synced to the stream of incoming documents. In addition, as illustrated in Section 5.2, the on-demand strategy is also capable to perform the more fine-grained scenario of interactive MT, with the distinguishing characteristics that the MT system does *not even need to exist* before its actual use.

In the experiments of this chapter, adaptation was performed by using a growing specialized phrase table. However, the phrase tables estimated from the main, large corpus were not contextually adapted. The training corpus used in our experiments is a mixture that consists of data from many different domains, origins, and quality, however, the examples used for model estimation are randomly selected from the training corpus with a uniform distribution. A possible continuation of this work is that it is possible to improve the sampling strategy that is used by our framework. In the next chapter, we will resort to sampling strategies that take contextual information into consideration, so as to perform some type of domain adaptation by preferring examples that appear more consistent with each input document.

lator, in average, a professional translator can translate about $3,000$ words per working day, amounting to roughly 1 week of work to prepare a typical `Newstest` or `Cochrane` development set.

6 Contextual Adaptation for On-demand SMT

The development of high-performance SMT systems usually requires large quantities of training data. Using as much data as is available for a given language pair is necessary to alleviate the data sparseness issue through better coverage. However, in typical settings, bilingual corpora used to train SMT systems are collected opportunistically; as a result, the amount of out-of-domain data largely outweights that of the in-domain data. Adding more bilingual data also increases the possibility of encountering new translations, and makes the translation of phrases more ambiguous, sometimes in a detrimental way, since not all corresponding translations (or senses) are appropriate for the input text. Hence, using all the available data as if all parts were equally relevant for the task at hand is not an optimal way of developing SMT systems.

A practical solution is to perform data selection to use only the most useful portion of the available data, depending on its relevance to the input text, to train a system. For instance, **?** performed data selection based on infrequent n-grams in the baseline system. This approach increases the lexical coverage of the system, but many other useful in-domain data from the full corpus are not exploited. **?** used the TF-IDF metric to select the top n similar sentences for each sentence in the test set. The selected sentences form an adapted training corpus, which is then used to train translation models. However, this approach may still face the lexical coverage issue.

In our previous experiments, translation models were estimated based on uniformly selected random samples. Random sampling was performed for each source phrase, ensuring good lexical coverage for our systems, yet no contextual information relative to the input text was used. In this chapter, we implement a contextual sampling strategy that aims to select the most appropriate translation examples for the estimation of document specific models. Note that the adaptation problem discussed in this work is limited here to *adapting translation models*, so that the issues of adapting target language models [**???**] will not be discussed.

The rest of the chapter is organized as follows. In Section 6.1, we will propose two contextual sampling strategies which enable to adapt translation models to the test domain. In Section 6.2, a confidence estimation of adapted models is described. We then present in Section 6.3 experiments designed to assess the performance of the contextual sampling strategies and the confidence model. In Section 6.4, we will apply the contextual sampling strategy in our on-demand SMT framework.

6.1 Contextual sampling strategies

In our systems, the model estimation for each phrase $\bar{s}$ in the input test data is performed on-demand and based on a selected subset of examples of $\bar{s}$ in the training corpus. Instead of selecting such a sample using uniform random sampling, we now want to bias sampling so as to prefer the examples that are most relevant to the input test data, in an attempt to perform some *contextual adaptation* of the translation and reordering models. As a definition for relevancy between one sentence in the training corpus and the input test data, we will use the average similarity between the sentence and each sentence in the input test data. Thus, the expected adaptation should be more thematic than based on local features (see Section 3.3.1). A number of text similarity metrics have been used in previous works, including `TF-IDF` [**??**], language model `perplexity` [**???**] and N-gram precision [**?**]. Considering the efficiency of computation and the compatibility with our on-demand framework, we selected N-gram precision and `TF-IDF` as relevancy metrics in this study. We will now describe both in turn.

6.1.1 N-gram precision

N-gram precision (NGP henceforth) is used by [**?**] to compute the similarity between two sentences for translation memory retrieval. It is defined as follows:

$$\texttt{Sim}(\mathrm{s}, \mathrm{s}^d) = \sum_{n=1}^{N} \frac{1}{N} p_n(\mathrm{s}, \mathrm{s}^d) \tag{6.1}$$

where s^d represents a sentence in the input test document and s a sentence in the training corpus, N defines the maximum length of n-grams considered, and $p_n(\mathrm{s}, \mathrm{s}^d)$ is the n-gram precision of the order n.

NGP forms a fundamental subcomputation of the corpus-level MT evaluation metric BLEU score (see Equation 2.15). In the definition of BLEU, the n-gram precision is defined as the ratio of clipped correct n-grams of order n in relation to the total number of n-grams of the same order in the translation hypothesis. By nature, using this term alone would tend to prefer shorter translation candidates. A *brevity penalty* term was hence introduced to correct this problem at the corpus-level. Similarly, using this metric alone for data selection will also bias the selection to prefer very long or very short sentences. Unfortunately, the brevity penalty, intended to be used at the corpus-level, is not applicable in the situation of data selection where the metric is applied at the sentence-level. In [**?**], a modified n-gram precision is defined as:

$$p_n(\mathrm{s}, \mathrm{s}^d) = \frac{|\mathrm{s}_{n\text{-grams}} \cap \mathrm{s}^d{}_{n\text{-grams}}|}{(1 - Z) * |\mathrm{s}_{n\text{-grams}}| + Z|\mathrm{s}^d{}_{n\text{-grams}}|} \tag{6.2}$$

where $\mathrm{s}^d{}_{n\text{-grams}}$ and $\mathrm{s}_{n\text{-grams}}$ are the set of n-grams in s^d and s, respectively, and Z is a penalty factor on the length of s. Larger Z values correspond to having a smaller penalty on the length of s and tend to prefer longer s; smaller Z values penalize more on the length of s and tend to prefer shorter s. The optimal value for Z can differ for different translation tasks, based on e.g. languages and corpora. Empirically, we found that larger Z values generally lead to better results than smaller ones. We make the hypothesis that long sentences containing more co-occurrences of the phrases in the input test document are more useful than short sentences, because the latter contain less contextual information. Empirically, we chose $Z = 0.75$ as the default value in our experiments.

In summary, NGP measures the surface string similarity between two sentences and gives higher preference to sentences s sharing the more n-gram phrases with s^d in the input test document. Note that n-grams are equally weighted by the metric.

6.1.2 TF-IDF

`TF-IDF`, is a similarity measure from Information Retrieval which has been used for data selection purposes in SMT [**??**]. Each document is represented by a vector $\mathbf{v_s} = (w_1 \ldots w_j \ldots w_n)$, where n is the size of the vocabulary. Each entry w_j in the vector is computed as:

$$w_j = tf_j \times \log(idf_j) \tag{6.3}$$

where, tf_j is the number of occurrences in the document of the j-th vocabulary word, and idf_j is the so-called *inverse document frequency* of the j-th word. We used the following definition:

$$idf_j = \frac{\#\text{ documents in corpus}}{\#\text{ documents containing the } j\text{-th vocabulary word}} \tag{6.4}$$

The similarity between two documents is then defined as the distance between the two corresponding vectors.

In this work, since we do not have the document boundary information in the training corpus, each source sentence in the training corpus s is treated as a document. Each sentence in the input text s^d is used as a separate query. The similarity between two sentences (s, s^d), following common usage, is defined as the cosine between the two vectors, given by:

$$\texttt{Sim}(\mathbf{s}, \mathbf{s}^d) = \cos(\mathbf{v_s}, \mathbf{v_{s^d}}) = \frac{\sum_j w_{\mathbf{s},j} w_{\mathbf{s}^d,j}}{||\mathbf{v_s}|| \, ||\mathbf{v_{s^d}}||} \tag{6.5}$$

In the contextual sampling process, each example candidate is ranked by its average similarity score with respect to all sentences in the input test document, and the top M (the sample size) highest-scoring portion is selected as our new sample.

In summary, `TF-IDF` measures some thematic similarity between two sentences by focusing on important words.

6.1.3 On-demand system construction with contextual sampling

The incorporation of the proposed sampling strategies into our on-demand SMT framework is straightforward. For each phrase $\bar{s}$ in the input test document $\mathbf{d}$, all its instances in the training corpus ($\mathbf{C}[\bar{s}]$) are found using the source language corpus suffix array. Instances in $\mathbf{C}[\bar{s}]$ are then ranked according to their similarity with the input test data. The best portion is selected as the sample to estimate translation models of $\bar{s}$. Note that for now we do not consider modifying the size M of translation samples, in order to allow us to make a direct comparison to results obtained by the random sampling strategy using the same sample size. The whole process is sketched in Algorithm 6.

Algorithm 6 On-the-fly model estimation with a contextual sampling strategy

```
parallel train corpus $\mathbf{C}$
given an input document $\mathbf{d}$, sample size $M$
compute $\mathbf{\Sigma}[\mathbf{d}]$
for all $\bar{s} \in \mathbf{\Sigma}[\mathbf{d}]$ do
    $\mathbf{C}[\bar{s}] = \texttt{find}(\mathbf{C}, \bar{s})$ // find all its occurrences
    for all $\mathbf{s} \in \mathbf{C}[\bar{s}]$ do
        $\mathbf{s}_{scores} = \texttt{Avg}(\texttt{Sim}(\mathbf{s}, \mathbf{s}^d), \forall \mathbf{s}^d \in \mathbf{d})$
    end for
    $\mathbf{S}[\bar{s}] = M$-best in $\mathbf{C}[\bar{s}]$
    if $\mathbf{A}_{\mathbf{S}[\bar{s}]}$ not exists then
        $\mathbf{A}_{\mathbf{S}[\bar{s}]} = \texttt{owa}(\mathbf{S}[\bar{s}])$
    end if
    $\texttt{estimate}(\boldsymbol{\theta}_{\bar{s}}, \mathbf{S}[\bar{s}], \mathbf{A}_{\mathbf{S}[\bar{s}]})$
end for
```

This approach is, unsurprisingly, more computationally expensive than the random sampling strategy (`rnd`). Unlike `rnd`, which scans the suffix array at fixed intervals (at most M operations), contextual sampling strategies in our current implementation require to retrieve all instances of a given source phrase and then to compute the similarity score for each of them with the input document. Its complexity thus depends on phrase frequencies and on the complexity of the computation of the similarity score used. By replacing the random sampling strategy `rnd` with the proposed contextual sampling strategies, we hope to select translation examples which are relevant for translating the input test document so as to perform contextual adaptation of translation and reordering models.

6.2 Confidence estimation of adapted models

Phrase scoring strategies used in conventional phrase-based SMT systems are based on simple count ratios and can thus be criticized on the following grounds:

1. A source phrase occurring rarely will result in its translations being over-estimated[1].
2. A majority of *inappropriate* examples for a given source phrase will probably make noisy translations being equally or more likely than correct translations, as well as increase model ambiguity.

The contextual sampling strategies presented in Section 6.1 allow us to assign a similarity score to each individual example and to select the most appropriate examples for estimation. In some sense, this contextual selection should reflect the confidence that the associated translations are thematically appropriate. By definition, an example only loosely matching the context

[1]Inverse translation models and lexical weighting are in a way meant to compensate for this.

of the input document should not be used for estimation. However, as presented in Section 6.1, if a source phrase has fewer than M occurrences in the training corpus, all its examples are selected for estimation independently of their similarity to the input sentence.[2] No contextual sampling is applied to these phrases. These phrases have too few occurrences in the training corpus and their translation probabilities may be over-estimated.

In addition to the relevancy of the examples used, their number of occurrences should participate in estimating the confidence in a translation distribution. Given a particular number of examples for a source phrase, the least informative, or least *committing*, situation would be one in which all translation examples are different, yielding the following conditional entropy:

$$H_{unif}(\bar{s}) = -\sum_{\bar{t}} p(\bar{t}|\bar{s}) \log p(\bar{t}|\bar{s}) = \log c(\bar{s}) \tag{6.6}$$

Intuitively, the better the examples used for contextual estimation of a phrase's translations, the more the conditional entropy for that phrase should be reduced, as translation alternatives should be restricted to a few synonymous translations. The information gain measured as a difference of entropy values between the previous situation and the more informative situation of a given model provides some account of how much confidence should be put in the collective contribution of the selected examples. We thus used the following as a new feature in our experiments involving adapted translation models:

$$\begin{aligned} h_{conf}(\bar{s}) &= \exp\left(H_{unif}(\bar{s}) - H(\bar{s})\right) \\ &= \exp(\log c(\bar{s}) + \sum_{\bar{t}} p(\bar{t}|\bar{s}) \log p(\bar{t}|\bar{s})) \end{aligned} \tag{6.7}$$

The computation of this feature is straightforward: $c(\bar{s})$ is efficiently obtained from the suffix array, and the entropy of translation probabilities ($\sum_{\bar{t}} p(\bar{t}|\bar{s}) \log p(\bar{t}|\bar{s})$) can be computed on-the-fly with uniform random sampling.

This value increases when either the number of examples for $\bar{s}$ is high or when the entropy of the adapted translation distribution is low. This score models the quality of source phrases. Larger numbers of examples usually lead to a better estimation of the translation distribution, and a lower entropy of the adapted translation distribution indicates that a phrase is less ambiguous. Actually, this score implicitly models the preference on the choice of phrases used by the decoder, where phrases of high frequency and low entropy of the adapted translation distribution should be used more often in translation than other less frequent and more ambiguous phrases.

6.3 Experimental validation

In this section, we design several experiments intended to assess our contextual sampling strategies and confidence model. Experiments are performed on the data sets described in Table 4.2. The vanilla `moses` systems in Section 4.4 was used again for our baseline systems. As alternative baselines, we also reuse our on-the-fly systems built in Section 4.4 with deterministic random sampling (`rnd`) using the same sample size.

[2]It is also possible to use instance weighting schemes in this situation.

	Newstest		WMT'14-med		Cochrane	
	BLEU	TER	BLEU	TER	BLEU	TER
	English → French					
moses-vanilla	35.84±0.05	46.14±0.10	40.84±0.11	42.23±0.09	34.12±0.10	48.59±0.22
rnd	35.64±0.15	46.31±0.08	40.87±0.12	42.23±0.04	34.08±0.05	48.48±0.27
NGP	35.53±0.09	46.46±0.03	41.48±0.13*	41.75±0.13	33.92±0.08	48.11±0.20
TF-IDF	35.67±0.08	46.34±0.03	41.91±0.05*	41.54±0.10	34.45±0.02*	47.12±0.16
+conf	**35.75±0.07**	46.46±0.08	**41.97±0.02***	41.55±0.15	**34.47±0.07***	47.23±0.16
	French → English					
moses-vanilla	37.45±0.03	44.29±0.04	39.85±0.05	39.77±0.04	37.11±0.08	42.81±0.08
rnd	36.64±0.05	45.04±0.13	40.21±0.01	39.84±0.05	37.48±0.14	42.54±0.04
NGP	36.59±0.16*	45.06±0.32	39.97±0.01	39.89±0.07	**37.99±0.08***	42.11±0.07
TF-IDF	36.84±0.09*	44.71±0.05	**40.44±0.08***	39.45±0.10	37.82±0.05*	42.06±0.02
+conf	**36.91±0.10***	44.55±0.12	**40.44±0.13***	39.45±0.10	37.94±0.02*	42.05±0.00

Table 6.1 – Experimental results for three MT tasks, in which all tasks use 100 as sample size except the WMT'14-med (English → French) task that uses 500 as sample size. (* significant at $p < 0.01$ level)

We developed various adapted systems using Algorithm 6. The purpose of these experiments is to validate the performance of the proposed sampling strategies and confidence model, so the same mgiza++ word alignments and language models as for our baseline systems were used. Experiments were performed on the three test data sets and in both translation directions. As in our previous experiments, all systems were optimized with KBMIRA; translations were computed with moses; and results are reported using BLEU and TER metrics, using the test set average of 3 optimization runs.

6.3.1 Results

Experimental results for the contextual sampling strategies and confidence model are presented in Table 6.1. Looking at the English-to-French translation results, on the Newstest translation task, we find that NGP is a little worse than random sampling rnd (-0.1 BP, $+0.1$ TP), while the TF-IDF sampling achieves the same performance as rnd. Both of them slightly underperform the moses system. Although translation performance is not improved by our contextual sampling strategies, the obtained results are not discouraging. Indeed, in their analysis of translation errors caused by shifting domain on the news commentary data, **?** found little difference between the baseline system and their error-corrected systems. In other words, the expected effects of a better thematic modeling for such a domain are modest.

By contrast, on the WMT'14-med translation task, both NGP and TF-IDF sampling significantly outperform rnd as well as the moses system. TF-IDF itself outperforms NGP ($+0.4$ BP, -0.2 TP). On the Cochrane translation task, NGP is a little worse than rnd on BLEU (-0.2 BP) while it is again better than rnd and moses on TER. TF-IDF outperforms both rnd and moses (0.4 BP and -0.14 TP).

In the reverse translation direction, French-to-English, a similar observation can be made on the Newstest translation task, NGP being slightly worse than rnd, and TF-IDF being a little better than rnd ($+0.2$ BP and -0.3 TP). Both of them, NGP and TF-IDF, are outperformed by the moses system by -0.3 BP and -0.2 BP, respectively. On the WMT'14-med translation task, NGP is a little worse than rnd (-0.2 BP), while TF-IDF performs a little better than rnd

	Newstest		WMT'14-med		Cochrane	
	Avg. # transl.	Entropy	Avg. # transl.	Entropy	Avg. # transl.	Entropy
	English → French					
rnd	47.11	3.16	157.33*	3.61	43.57	3.01
TF-IDF	44.07	3.08	141.41*	3.51	40.20	2.93
	French → English					
rnd	31.19	2.66	26.56	2.39	27.92	2.49
TF-IDF	30.80	2.66	22.93	2.22	25.41	2.41

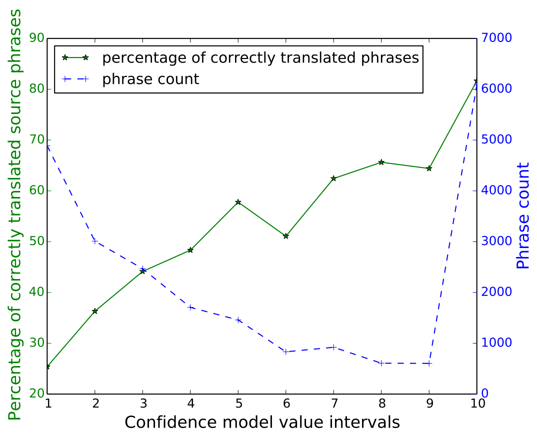
Table 6.2 – Average number of translations and average entropy of source phrases in the translation tables (considering only source phrases with a frequency higher than M). Experiments marked by * use $M = 500$, all the others experiments use $M = 100$.

($+0.2$ BP and -0.4 TP). On the Cochrane translation task, both NGP and TF-IDF perform better than rnd and moses.

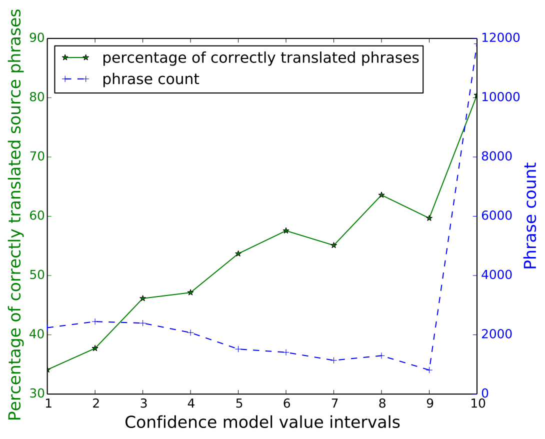
In general, TF-IDF performs better than NGP on most of translation tasks and it is systematically at least as good as rnd, and better on medical translation tasks. The performance of NGP is not quite stable. TF-IDF estimates some thematic similarity between sentences and aims to select sentences that are about the same topic as the input test document, which should mostly play a role in *word translation disambiguation*. NGP computes some string similarity that considers units larger than simple words, without any regard for the significance of the considered n-grams nor for their size. Consequently, NGP may be more useful for retrieving translation examples when the test data is of a repetitive, stereotypical nature, which is precisely the case in the Cochrane scenario.

Our experiments have shown that the contextual sampling strategy can yield improvements over the baseline systems. Note that on all translation tasks, TF-IDF and rnd use the same sample size M, the only difference between them lying in the selected examples which are used for estimating translation models. By analyzing the phrase tables, we also find that the phrase tables extracted using TF-IDF are less diverse than the phrase tables extracted using rnd, suggesting than fewer, more coherent translations were extracted. As shown in Table 6.2, the TF-IDF phrase tables for our test sets contain significantly fewer translations, in average for each source phrase, and their average entropy of translation model ($p(\bar{t}|\bar{s})$) is lower than that of the rnd phrase table. On the one hand, the contextually adapted phrase tables have less noisy translations, which also reduces the decoding complexity. On the other hand, the lower average entropy of the contextually adapted phrase tables makes the choices of translation more sure.

As TF-IDF yields more stable and usually better performance than NGP, we decided to assess our confidence model on systems using TF-IDF. By adding the confidence model (+conf) into the TF-IDF systems, we find that translation performance is systematically improved over the corresponding TF-IDF variant, although the resulting improvements are always modest (see Table 6.1). We then assessed whether our confidence model (Section 6.2) is a good predictor of translation quality. Figure 6.1 plots the percentage of correctly translated source phrases in the trace of the decoder (counted as such when their target phrase matches the reference translation) against score intervals of the models. For the TF-IDF+conf systems on the Newstest task in both translation directions, we observe a clear tendency to provide better translations for test phrases with higher confidence. This result clearly shows that our proposed confidence model is a good indicator of useful phrases in the phrase table, although its impact on the automatic translation tasks has been found to be modest.



(a) English to French



(b) French to English

Figure 6.1 – Percentage of correctly translated source phrases in the trace of the decoder of the `TF-IDF+conf` systems against score value intervals of the confidence model (`conf`) in the `Newstest` translation task. The dashed line represents the number of phrases in each confidence value interval.

6.4 Contextual adaptation of on-demand SMT systems

In Section 5.1, we have presented a series of increasingly richer configurations of our on-demand SMT development framework, and have shown that our system can deliver fast, yet competitive translations for these documents. In these experiments, the input was a sequence of documents, and translation models and word alignments were computed on-the-fly. Recall that in `Config0`, each input document is processed in isolation, which is computationally expensive since much information is re-computed for each input document. In `Config1`, all computed information are cached in the system and could be used directly if needed. This caching scheme largely reduces the computational burden of system development. In order to reduce it further, in `Config2` sampling is biased by preferring to select previously aligned sentences whose word alignments are already in the cache of the system. However, no contextual information was considered so far.

In the previous section, we have described our contextual sampling strategies and found that the one based on `TF-IDF` leads to better performance. The similarity metric used can be computed efficiently using the available suffix array, and is fully compatible with our on-demand SMT development framework. In this section, we now incorporate contextual sampling into our incremental SMT development framework described in Section 5.1

`TF-IDF` was used as the sampling selector. In the system, input documents were again processed one by one, but the sampling was adapted to each document based on the `TF-IDF` similarity between each training examples and the input document. The selected examples were then aligned by our on-demand word alignment method and used for model estimation. The whole procedure is thus the same as the one described in Algorithm 4, except for the sampling strategy.

Results for these experiments are provided in Table 6.3, where `Config_ctx` represents systems constructed using the `TF-IDF` contextual sampling strategy. `Config1`, `Config2`

Configs	Translation quality		PT construction time	
	BLEU	TER	user CPU	wall clock
`Config1`	28.87	49.40	111h	10h
`Config2`	28.58	49.54	76h	7h
`+spec`	32.33	46.42	76h (+0.5h)	7h
`+online`	36.41	46.44	-	-
`Config_ctx`	28.84	49.78	95h	9h
`+spec`	32.54	46.73	95h (+0.5h)	9h
`+online`	36.67	46.02	-	-
`+conf`	36.67	45.84	-	-

Table 6.3 – Incremental development of SMT systems with contextual sampling (`TF-IDF`).

and `Config2+X` correspond to the same systems as in Chapter 5 and are used as baseline systems for these new experiments.

First, by comparing `Config_ctx` with `Config1` and `Config2`, we find that these three systems yield very close results, `Config_ctx` being slightly better than `Config2` and slightly worse than `Config1`. Since no parameter tuning was performed so far, such observations were expected.

Since the word alignment is by far the most time-consuming process in our SMT system development framework, the processing time directly reflects the number of sentences which have to be aligned. `Config_ctx` is faster than `Config1`, which means that it aligns fewer sentences. In other words, the `TF-IDF` sampling strategy re-selects more often the aligned sentences which are already selected for previous documents than `rnd`, a consequence of the thematic adaptation that is performed. Since all documents are from the same source, a sentence example which is considered useful and selected by `TF-IDF` for a document in the sequence will probably also be considered useful for subsequent documents and be selected anew. Hence, the `TF-IDF` sampling strategy may more often reuse the selected and aligned sentences than `Config1`, which explains why `Config_ctx` system is faster than `Config1` system. `Config_ctx` is slower than `Config2`, since `Config2` was, by contrast, designed to reuse as much as possible selected sentences so as to reduce as much as possible the need for aligning new sentences.

Adding the `spec` phrase table into the `Config_ctx` system yields an improvement of about $+3.7$ BP and -3.1 TP. Applying our online tuning (`+online`) strategy yields an additional improvement of $+4.1$ BP and -0.7 TP relative to the `Config_ctx+spec` system. Comparing to the `Config2+X` systems, `Config_ctx+spec` is a little better than the `Config2+spec` system, and such an improvement also exists in the `+online` systems. Note that the only difference between `Config_ctx+X` and `Config2+X` systems is their main phrase tables where in `Config_ctx+X`, the main phrase tables are estimated on the large-scale training corpus using `TF-IDF` sampling strategy while in `Config2+X`, the main phrase tables are estimated on the same training corpus but using the biased random sampling strategy described in Section 5.1.3.

We also reproduce the document-level analysis that was performed in Section 5.1. The comparison between the systems with online tuning (`Config2+spec+online` and `Config_ctx+spec+online`) and the `moses` system is shown in Figure 6.2.

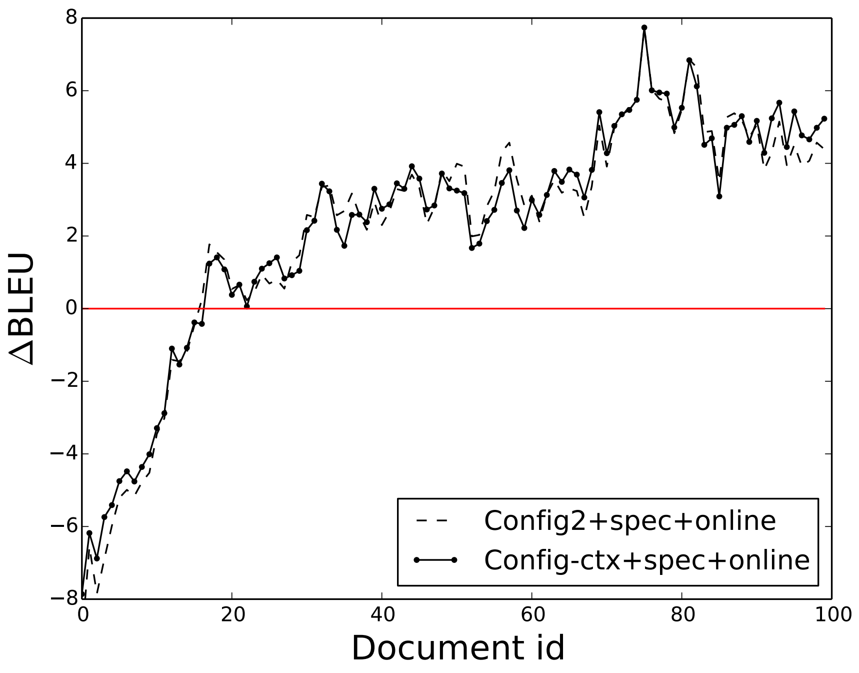


Figure 6.2 – Document-level comparison with the `moses` system with online tuning.

We find that the two configurations perform very close to each other, which confirms their similar results at the corpus-level. We also find that the contextually adapted system `Config_ctx+spec+online` performs systematically better than `Config2+spec+online` at the beginning of the document sequence, which would mostly explain the corpus-level BLEU difference. Unfortunately, in this experiment the effect of adapted model in the main phrase table is dominated by the effect the small in-domain `spec` table. At the beginning of the document sequence, the quantity of data used to extract the `spec` table is small and the adapted model in the main phrase table possibly help to improve the translation of documents. However, as the system progresses in the document sequence, more and more in-domain data are used to extract the `spec` table which plays a growing part in subsequent translations.

Finally, as was illustrated in Figure 6.1, the confidence model was found to be a reasonably good indicator of the reliability of the phrases in the phrase tables. Our last experiment in this section is designed to assess if the conclusions on the confidence model obtained in the previous section still hold for our on-demand SMT system with incremental training.

We reused the same configuration as `Config_ctx+spec+online` and added the confidence model into the main phrase table of each document.[3] Looking at the results (see `+conf` Table 6.3), we do not observe any improvement on the BLEU metric, and only a small improvement on TER (-0.2 TP) relative to the `Config_ctx+spec+online` system. This result is very similar to the results obtained in the experiments of the previous section using the same data set. Analyzing the percentage of correctly translated source phrases in each confidence model value intervals as shown in Figure 6.1, we observe the similar tendency. According to these results, the confidence model does not improve translation here. The infor-

[3]The confidence model was not added into the `spec` phrase table because this table is estimated based on too few data and there are little differences between $H_{unif}(\bar{s})$ and $H(\bar{s})$ (cf. Equation (6.7)).

mation contained in this model is possibly implicitly contained in other models of the phrase table because it is computed directly based on the other models in the phrase table. However, this model is a good and direct indicator for us to know which phrases are well estimated and which are not and should deserve more attention, possibly resorting to human intervention and to supplementary resources.

6.5 Summary

This chapter has addressed the issues of the contextual adaptation of the systems developed in our framework. Adaptation was performed through contextual sampling, where two similarity metrics were considered: n-gram precision and `TF-IDF`. `TF-IDF` was found to perform at least as well, and often better than random sampling which was used through Chapters 5 and 4. We also integrated our contextual sampling strategy into our on-demand SMT system using `owa` alignments, and obtained results on par with those obtained by the baseline `moses` system. We also proposed a method to estimate the confidence of adapted models which was added into the phrase table as an additional model. Experimental results and further analyses have shown that this model may not be a strong predictor of translation quality but that it provides us with useful information as to which phrases should be better estimated.

The contextual sampling strategies used in this chapter only exploit information in the source language. The experiments reported in Chapter 5 clearly showed that the specialized phrase tables (`spec`) extracted from previously translated documents can significantly improve translation quality. In fact, these translated documents could also be used to improve the contextual sampling by taking advantage of the contextual information also in the target language. This feature could be used in an interactive translation framework: after each sentence has been post-edited by a human translator, the new data could be used readily and instantly in our framework to update the translation models, but also to bias subsequent samplings to select translation examples that match the style and choices made by the human translator.

Conclusion

Contributions

The very large computational cost associated with the development of high performance MT systems has long been acknowledged, and is problematic for several reasons. First, this cost seriously constrains the number of experiments that can actually be run on very large datasets and the frequency of update of systems when new data become available. It also impedes the portability of high performance SMT systems on light-weight computational devices. Finally, it constitutes a barrier to entry for newcomers in the field, who need to invest an increasingly high amount of resources to keep up with the state-of-the-art.

On-the-fly model estimation, previously studied in [**??**], has been shown to be a fast yet competitive approach to extract resources for data-based MT. In these works, phrase pair inventories need no longer to be pre-computed, and are only computed for the phrases occurring in the text to translate, thereby largely reducing the computational burden of model extraction. This burden can be further reduced by resorting to sampling, which makes use of a limited number of examples for model estimation without significantly altering translation quality. Nevertheless, previous studies only answer part of the question, as they still require to pre-compute all word alignments, which remains, by an large the most time consuming step in large scale SMT system development.

The first main contribution of this thesis was hence devoted to a new on-demand word alignment method that aligns training sentence pairs in isolation. This property was necessary to enable the development of on-demand SMT systems, yielding significant additional gains in processing time, in particular when few texts have to be translated. With now a situation where all the necessary information for translation model estimation are computed on-demand, pre-translation processing time mainly depends on the size of the input text rather than on the available training data. As a concrete illustration, our framework can build translation tables for individual sentences in less than 1 minute with a parallel corpus comprising about 400 million source tokens.

Another crucial property is the seamless integration of any newly available data. This enables to make use of any periodically available new data, but also to immediately exploit manual post-editions, so that the system can avoid making repetitive errors and adapt to a specific domain and style in an interactive scenario.

Another important issue that we identified in the previous works of **?** and **?** is that the translation examples used were randomly selected from the training corpus. Although experimental results showed that this approach can achieve competitive translation performance, it is not optimal, especially when the majority of the training data is out-of-domain relative to the input text.

The second main contribution of this thesis was to integrate contextual sampling strategies to select translation examples from the large-scale corpus that are similar to the input text, and hence build adapted phrase tables. Similarly to the traditional data selection approaches, our contextual sampling selects the most relevant sub-parts of the training corpus. However, in contrast to most data selection approaches, our approach is performed at the level of phrases. This enables to both exploit the entire training corpus, and to ensure the best attainable lexical coverage. Additionally, our implementation fits our on-demand development scheme, where contextual information is computed only for phrases occurring in the input text and where the similarity metrics used are cheap to compute.

Future work

We believe that our work has addressed important issues in SMT system development, and that significant contributions were proposed and validated empirically. Nevertheless, we envision a number of ways in which on-demand system development can be further improved, and list the most important ones in the remainder of this concluding chapter.

On-demand estimation of adapted language models We have focused in the present work on the estimation and adaption issues of translation models and have not addressed these issues for the target language model. Numerous works have shown the potential of language model adaptation, and our framework could straightforwardly accomodate a dynamic language model, for instance by making use of a suffix array of the target language corpus [**?**].

Allowing unaligned words Our experiments have shown that our on-demand word alignment approach can yield results that are comparable to a state-of-the-art baseline on small-scale data sets (cf. Table 4.6). However, it was found to be systematically inferior on large-scale data sets (cf. Table 4.10). One possible reason is that our on-demand word alignment does not allow unaligned words. As illustrated in Section 4.1.3, this constraint largely reduces the number of phrase pairs that can be extracted, and forces many function words to be extracted jointly with content words. On the source language side, this increases the risks of phrase extraction failure; on the target language side, it decreases the diversity of phrases that will be presented to the language model. Studying how out top-down alignment algorithm could accommodate unaligned words would then constitute a promising continuation of our work.

Adaptation of translation example sampling Contextual sampling strategies have been integrated in our framework to bias sampling of examples so as to perform some type of adaptation. However, contextual information was so far only exploited on the source side of the training corpus. In a situation similar to interactive MT, we have measured significant improvements when post-edited translation examples were immediately integrated into the system by means of an additional, *specialized* phrase table (see Chapter 5). In fact, post-edited examples could also be useful to better select future translation examples on the basis of their coherence in terms of phrasal translation choices. We thus envision that sampling could be adapted based on both source- and target-side contextual information. Furthermore, the sample size could be dynamically adapted for each individual phrase, stopping sampling as early as possible when stable translation distributions are obtained, or conversely sampling larger numbers of examples when more diversity should be proposed to the decoder for ambiguous phrases. We note that, in the extreme, our SMT framework can be configured to work as an hybrid SMT-EBMT system, in particular if only the most similar example of each phrase is retrieved and used.

Improvements in automatic translation diagnosis Our framework particularly lends itself to tracing how a specific automatic translation was generated: from the bi-phrases used to compose it, it is straightforward to go as far back as the individual training examples that contained the corresponding information and to inspect how and why the word alignment was produced. In fact, every aspects of model estimation could be run on-the-fly for inspection; for instance, a probabilistic phrasal dictionary could be obtained in a nick of time, with each translation choice linked to its examples in the parallel corpus. Such capacities could also be put to use for pedagogical purposes to teach the basics of SMT or to provide assistance in the analysis or quality control of parallel corpora.

Exploiting comparable corpora Another interesting direction would be to accommodate within our framework large collections of *not so parallel* sentence pairs. Comparable corpora have proven to be useful resource for MT systems [**?**]. The information mining techniques used on comparable corpora could be integrated into our framework in a on-demand development scheme which yields a situation where both parallel and comparable bilingual corpora may be used seamlessly.

Appendices

A
Abbreviation

- BLEU: Bilingual Evaluation Understudy
- BP : BLEUPoint
- BCED : Bilingual Cross-Entropy Difference
- CAT : Computer-Assisted Translation
- CED : Cross-Entropy Difference
- DL : Discontinuous Left
- DR : Discontinuous Right
- EBMT : Example-Based Machine Translation
- EM : Expectation-Maximization (algorithm)
- EMEA : European Medicine Agency
- HMM : Hidden Markov Model
- HTER : Human Translation Error Rate
- IDF : Inverse Document Frequency
- IMT : Interactive Machine Translation
- IOB : Inside, Outside, Begin (chunking tags)
- IR : Information Retrieval
- ITG : Inverse Transduction Grammars
- KBMIRA : K-Best MIRA
- LIMSI : Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur (Computer Sciences Laboratory for Mechanics and Engineering Sciences)
- LM : Language Model
- MERT : Minimum Error Rate Training
- METEOR : Metric for Evaluation of Translation with Explicit Ordering
- ML : Monotone Left
- MR : Monotone Right
- MT : Machine Translation
- NGP : N-gram Precision
- NLP : Natural Language Processing
- OOV : Out-Of-Vocabulary
- POS : Part-Of-Speech
- PP : perplexity

- RBMT : Rule-based Machine Translation
- SL : Swap Left
- SMT : Statistical Machine Translation
- SR : Swap Right
- TER : Translation Error Rate
- TF : Term Frequency
- TP : TER Point
- VSM : Vector Space Model
- WMT : Workshop on Statistical Machine Translation
- WSD : Word Sense Disambiguation

B

Extracts of Data

B.1 `Newstest`

Labour defends budget tax rises
ministers have defended the tax rises and spending cuts announced in the pre-Budget report against criticism from the opposition , business and unions .
the Tories said Alistair Darling " blew " an opportunity to show he was serious about cutting the deficit by delaying decisions until after the election .
the chancellor also came under fire for hitting low and middle income workers .
among his headline proposals are a 0.5 % rise in National Insurance and a 1 % cap on public pay settlements from 2011 .
National Insurance anger
unions have protested that low-paid workers are being penalised for a recession not of their making and warned of " problems " ahead .
the National Insurance increase - which will raise about £3bn a year - has angered the business community , which says it is a tax on jobs when the focus should be on economic recovery .
the increase , limited to those earning more than £20,000 a year , will hit about 10 million workers .

B.2 `WMT'14-med`

the aim of this study was to investigate the effect of in vivo inhibition of factor XI and TAFI in an experimental thrombosis model in rabbits .
cardiac arrests are sometimes referred to as cardiopulmonary arrest , cardiorespiratory arrest , or circulatory arrest .
it's a long , hollow tube at the end of your digestive tract where your body makes and stores stool .
about 5 percent of people with ulcerative colitis develop colon cancer .
post-transplant cancers which are not virus-inducted can be relied to genetic factors of the transplanted patient and / or the transplant donor .
soft tissue injectables and fillers are a non-surgical option for facial rejuvenation that address the loss of

volume that accompanies facial aging .
we will investigate if there is a change in your lung inflammatory cells after the endotoxin challenge when you take the gT versus when you take a placebo .
patients with type 1 and type 2 diabetes mellitus (DM) will be included .
the first 6 months will be a wash in period and participants will be randomised (1/1) to their treatments at the 6 month visit .
polyp-like varices are shown here in the gastric cardia , seen on retroflexion of the endoscope .

B.3 Cochrane

antipsychotic medication versus placebo for people with both schizophrenia and learning disability .
antipsychotic medication is the standard treatment for people with learning disability and schizophrenia .
to determine the effects of any antipsychotic medication compared with placebo for treating people with a dual diagnosis of learning disability and schizophrenia .
for this update we searched the Cochrane Schizophrenia Group's Register of trials (July 2004) , relevant reference lists and sought unpublished data from pharmaceutical companies .
we included all randomised clinical trials of longer than one month's duration , involving people with both schizophrenia and learning disability (a measured IQ of 70 or less) that evaluated antipsychotic medication versus placebo .
we reliably selected and assessed studies for methodological quality .
two reviewers , working independently , extracted data .
we would have analysed dichotomous data on an intention-to-treat basis and presented continuous data with 65 % completion rate .
for dichotomous outcomes , our intention was to estimate a fixed effect relative risk (RR) with the 95 % confidence interval (CI) together with the number needed to treat / harm (NNT / H) .
we found only one relevant randomised trial using our search method and this had to be excluded .
this study included four people with a dual diagnosis of schizophrenia and learning disability , but results were only available for two of the participants .
it was unclear as to which groups the other two people were allocated .
in order to display the data , we would have had to have made too many assumptions about these two people and any results would be uninformative and potentially misleading .
using the methods described we found no randomised controlled trial evidence to guide the use of antipsychotic medication for people with both learning disability and schizophrenia .
until the urgent need for randomised controlled trials is met , clinical practice will continue to be guided by extrapolation of evidence from randomised controlled trials involving people with schizophrenia , but without learning disability , and non-randomised trials of those with learning disability and schizophrenia .

C

Documents in Any-text Translation

The documents used in the Any-text translation scenario (see Section 5.2) are obtained from `OPUS`[1].

- `talk1`: Jane McGonigal: Gaming can make a better world (Filmed February 2010 at TED2010).
- `talk2`: Bill Gates: Innovating to zero! (Filmed February 2010 at TED2010).
- `book1`: Alice's Adventures in Wonderland (Author: Lewis Carroll).
- `book2`: The Great Shadow (Author: Arthur Conan Doyle).
- `book3`: Candide (Author: Voltaire).
- `subtitle1`: Breakout Kings (Season 2 Episode 7 - Ain't Love (50) Grand; Air data April 14, 2012).
- `subtitle2`: Jane by Design (Season 1 Episode 11 - The Replacement Original; Air date June 5, 2012).
- `php`: A parallel corpus originally extracted from PHP manual[2].
- `kdedoc`: A parallel corpus of KDE manuals.

[1]`http://opus.lingfil.uu.se/`
[2]`http://se.php.net/download-docs.php`

D Translation for communities: translation examples

English input 1: the main outcome measure was abstinence from smoking after at least six months follow up .

French reference: le principal critère de jugement était l' abstinence tabagique après un suivi d' au moins six mois .

`moses`: la principale mesure des résultats a été l' abstinence de fumer après au moins six mois de suivi .

`on-demand`: le critère principal était l' abstinence du tabac après au moins six mois de suivi .

`+spec`: le critère principal était l' abstinence tabagique après au moins six mois de suivi .

`+online`: le principal critère de jugement était l' abstinence tabagique après au moins six mois de suivi .

English input 2: we updated this search on 30 June 2010 and added the results to the awaiting classification section of the review .

French reference: nous avons mis à jour ces recherches le 30 juin 2010 et ajouté les résultats à la section de classification en attente de la revue .

`moses`: nous avons mis à jour cette perquisition le 30 juin 2010 et ajouté les résultats à la section de classification en attente de l' examen .

`on-demand`: on a mis à jour cette perquisition le 30 juin 2010 et ajouté de l' attente de la section de l' examen .

`+spec`: nous avons mis à jour ces recherches le 30 juin 2010 et ajouté les résultats à la section de classification de l' examen .

`+online`: nous avons mis à jour ces recherches le 30 juin 2010 et ajouté les résultats à la section en attente de classification de la revue .

E

Publications By the Author

2014

- **Li Gong**, Aurélien Max, François Yvon. Vers un développement plus efficace des systèmes de traduction statistique: un peu de vert dans un monde de BLEU. In *Proceedings of Traitement Automatique du Langage Naturel 2014 (TALN)*, Marseille, France, 2014.

- **Li Gong**, Aurélien Max, François Yvon. Construction (très) rapide de tables de traduction à partir de grands bi-text. In *Proceedings of Traitement Automatique du Langage Naturel 2014 (TALN)*, Marseille, France, 2014.

- Nicolas Pécheux, **Li Gong**, Quoc Khanh Do, Benjamin Marie, Yulia Ivanishcheva, Alexandre Allauzen, Thomas Lavergne, Jan NieHues, Aurélien Max, François Yvon. LIMSI @ WMT'14 Medical Translation Task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, Baltimore, Maryland USA, 2014.

- Hélène Bonneau-Maynard, Natalia Segal, Eric Bilinski, Jean-Luc Gauvain, **Li Gong**, Lori Lamel, Antoine Laurent, François Yvon, Julien Despres, Yvan Josse, Viet Bac Le. Traduction de la parole dans le projet RAPMAT. In *Proceedings of the 30th édition des Journées d'Etudes sur la Parole*, Le Mans, France, 2014.

2013

- **Li Gong**. La traduction automatique statistique, comment ça marche ? In *Interstices.info*, 2013 (https://interstices.info/traduction-automatique-statistique).

- **Li Gong**, Aurélien Max, François Yvon. Improving Bilingual Sub-sentential Alignment by Sampling-based Transpotting. In *Proceedings of the 10th International Workshop on Spoken Language Translation (IWSLT)*, Heidelberg, Germany, 2013.

2012

- Hai-Son Le, Thomas Lavergne, Alexandre Allauzen, Marianna Apidianaki, **Li Gong**, Aurélien, Max, Artem Sokolov, Guillaume Wisniewski, François Yvon. LIMSI @ WMT12. In *Proceedings of the Seventh Workshop on Statistical Machine Translation (WMT)*, Montréal, Canada, 2012.
- **Li Gong**, Aurélien Max, François Yvon. Towards Contextual Adaptation for Any-text Translation. In *Proceedings of the 9th International Workshop on Spoken Language Translation (IWSLT)*, Hong Kong, China, 2012.