Incremental Development of Statistical Machine Translation Systems

Li Gong[1,2], Aurélien Max[1,2], François Yvon[1]

LIMSI-CNRS[1] & Univ. Paris-Sud[2]
rue John von Neumann, 91 403 Orsay, France
{firstname.lastname}@limsi.fr

Abstract

Statistical Machine Translation produces results that make it a competitive option in most machine-assisted translation scenarios. However, these good results often come at a very high computational cost and correspond to training regimes which are unfit to many practical contexts, where the ability to adapt to users and domains and to continuously integrate new data (eg. in post-edition contexts) are of primary importance. In this article, we show how these requirements can be met using a strategy for on-demand word alignment and model estimation. Most remarkably, our incremental system development framework is shown to deliver top quality translation performance even in the absence of tuning, and to surpass a strong baseline when performing online tuning. All these results obtained with great computational savings as compared to conventional systems.

1. Introduction

Statistical Machine Translation (SMT) has considerably matured in the past decade and is nowadays a competitive option in most practical machine-assisted translation scenarios. A notable fact about SMT technology is that the construction of high-performance systems is extremely expensive. Even if using appropriate computing resources and parallel programming techniques, building systems for very large data sets requires a significant processing time before any translation can be produced. If individual processing steps may be greatly accelerated, including e.g. word alignment [1] or system tuning [2], the requirement to process the entire 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 [3], this selection is itself time-consuming and is not justified when one only needs to translate a handful of documents or documents from multiple domains.

In addition, the trained translation models are *static*. In a state-of-the-art system, all models are extracted from a predefined parallel corpus, and are then used to translate any type of input text. However, 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 increase the n-gram coverage and to improve the parameter estimates of an existing system. 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.

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 [4, 5]. The main strength of these approaches is that they reduce the computation time of translation models and make it possible to extract translations from very large parallel data, even with arbitrarily long translation units. However, 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.

In this work, we propose to experiment with an architecture where word alignments are only computed on a per-need basis. This proposal naturally enables efficient, *plug-and-play* use of any newly available parallel data, as well as online learning of system parameters. This is similar to the objectives of stream-based SMT [6], but crucially does not require the *actual* alignment of all available data. This means that we are able to develop systems even faster: as our experiments show, immediate integration of newly translated documents, combined with online tuning, make it possible to dispense altogether with the development step. This pragmatic solution offers both the capacity to deliver translations to users much earlier, but also to quickly improve subsequent automatic translations.

The rest of the paper is organized as follows. In Section 2, we describe our framework for efficient on-demand development of SMT systems. We then present in Section 3 experiments designed to demonstrate the capabilities and flexibility of our framework. We finally conclude by reviewing related work in Section 4.

2. On-demand development of SMT systems

2.1. 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, based on small samples of parallel sentences. In our architecture, parallel sentence pairs are stored in a suffix array [7], enabling fast access to phrase instances.[1] At decoding time, the translation probabilities for all source phrases $\bar{s}$ (up to a given length) are computed based on a subset of occurrences of $\bar{s}$, where the sample size (denoted as M) enables to balance between speed and precision of estimates.

Previous approaches [4, 5] to sampling have resorted to *random deterministic sampling*, which picks a given number of examples by scanning the suffix array index at fixed intervals. The translation probability of a source phrase is then computed as:

$$p(\bar{t}|\bar{s}) = \frac{count(\bar{s},\bar{t})}{\sum_{\bar{t}'} count(\bar{s},\bar{t}')} \tag{1}$$

where $count(\cdot)$ is the number of occurrences of the given phrase pair in the sample, which may include occurrences where translation extraction was not possible (what Lopez [5] calls a *coherent* estimation of the translation model, which is found to generally improve performance).

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. If needed,[2] the following approximation can be used instead:

$$p(\bar{s}|\bar{t}) = min(1.0, \frac{p(\bar{t}|\bar{s}) \times freq(\bar{s})}{freq(\bar{t})}) \tag{2}$$

where $freq(\cdot)$ is the relative frequency of the given phrase in the entire corpus. The numerator ($p(\bar{t}|\bar{s}) \times freq(\bar{s})$) represents the predicted joint probability of $\bar{s}$ and $\bar{t}$.

2.2. On-demand word alignment

The second main peculiarity of our architecture is the ability to perform word alignment on demand for a subset of selected bi-sentences. Word and phrase alignments are required to compute Equation (1), and are obtained using our implementation of the sampling-based alignment method described in [8], which relies on ideas originally introduced in [9]. In this approach, the word alignment between a pair of parallel sentences is generated by a recursive binary segmentation process. Starting with a sentence-level alignment (explicitly available in the parallel corpus), segmentation is performed recursively to match smaller blocks until no block can be further segmented.

This process can be viewed as approximate top-down ITG parsing [10], where matching blocks are determined based on association scores between the words in the source and target sentences. In this study, association scores for the words in the source part of the bi-sentences of interest are generated by a sampling-based transpotting method, which also relies on a sampling strategy and is thus also quite fast. It is however worth noting that any kind of lexical score could be used to measure the strength of word associations.

2.3. System construction

As described before, our framework contains two main parts: on-the-fly model estimation with deterministic random sampling (denoted as `rnd`, henceforth) and on-demand word alignment (denoted as `owa`, henceforth).

The corresponding processing architecture is sketched in Algorithm 1. Given an input document $\mathbf{d}$ to translate, the system first extracts all possible source phrases, $\mathbf{\Sigma}[\mathbf{d}]$. Then, for each source phrase $\bar{s}$ in $\mathbf{\Sigma}[\mathbf{d}]$, we perform deterministic random sampling to select translation examples from the parallel corpus. We then obtain a translation sample of $\bar{s}$, $\mathbf{S}[\bar{s}]$. The sentence pairs in $\mathbf{S}[\bar{s}]$ are then aligned by our on-demand word alignment, 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.

Besides the translation models, the other models in our system are the same as in the default configuration of the `moses` system [11], including the lexical weighting and lexicalized reordering models. These models are also computed on-demand based on the computed word alignments.

Algorithm 1 On-demand development procedure

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

3. Experiments

In this section, we have chosen to illustrate two favorable use cases of our framework in order to demonstrate its capabilities and flexibility. The data used in this work is presented in Section 3.1. In Section 3.2, 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 take advantage of similarities between documents to outperform a strong baseline. In Section 3.3, another even more difficult use case, which we called *any-text translation*, will be studied.

3.1. Data

We selected English-French as our main language pair for this study, mostly because large quantities of parallel data

[1]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 can be constructed in $\mathcal{O}(|\mathbf{C}| \log(|\mathbf{C}|))$ time.

[2]Although this model has been shown to be non essential, we use it for the stability of our systems, especially when untuned systems are used.

Documents	# lines	#token$_{en}$	#token$_{fr}$	Domains
WMT	16.6M	396.9M	475.1M	Mixture
Cochrane(dev)	743	16.5K	21.4K	Medical
Cochrane(100 docs)	1.8K	38.6K	49.3K	Medical
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
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 1: Description of corpora used in our experiments.

are readily available for this language pair. Data from the Workshop on Statistical Machine Translation (WMT)[3] from a variety of domains were used, as well as additional data from various origins from the medical domain and used in the WMT'14 medical task.[4] This dataset, denoted as WMT, contains data from different domains, including News commentaries, parliamentary debates and medical texts.

In the "translation for communities" scenario, we used data of systematic summaries for specialists from the Cochrane collaboration.[5] The Cochrane dataset is made up of short documents typically containing one or two dozen of sentences. In the "any-text translation" scenario, we chose 8 documents from various domains: two entire transcriptions of TED Talks, two translated books, two movie subtitles and two technical manuals. Table 1 provides basic statistics regarding these corpora. Tokenization was performed using in-house tools.

3.2. Translation for communities

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 exists 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 thus constructed a vanilla moses system. We used mgiza++[6] to align the full bi-corpus and the moses scripts to extract a huge phrase table and a reordering table for the entire parallel corpus (respectively 20Gb and 7.5Gb com-

[3]http://www.statmt.org/wmt13
[4]http://www.statmt.org/wmt14
[5]http://summaries.cochrane.org
[6]http://www.kyloo.net/software/doku.php/mgiza:overview

Configs	Translation quality		PT construction time	
	BLEU	TER	user CPU	wall clock
moses	34.12	48.59	1,212h	252h
on-demand	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	14h

Table 2: Results for the owa system on a large-scale English-to-French translation task.

pressed on disk), which have to be filtered for each input text. The medical-domain LM was trained on the French side of WMT'14 medical data (containing 4.8M sentences and 78M tokens). The system was optimized with KBMIRA, a variant of the Margin Infused Relaxation Algorithm described in [12], on the Cochrane development set. Translations are computed with the moses phrase-based decoder. Results are reported using the BLEU [13] and TER [14] metrics.

In this first scenario, we consider a situation where a stream of documents needs to be translated. After each document has been automatically processed, 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 is illustrated using the Cochrane dataset, where we take the 100 documents constituting the test set (see Table 1) to simulate the document stream. In the following, we describe a series of increasingly rich configurations and show that our framework can deliver fast, yet competitive translations for these documents.

3.2.1. On-demand development of systems (on-demand)

In the first configuration, our system processes each input document separately in sequence, as described in Algorithm 1. Word alignments of previously aligned sentences will be cached and readily be available for subsequent documents. Each document-specific translation table is fed to the decoder[7], which uses the default values for all model parameters. In this configuration, no tuning is actually performed, which eliminates completely the need for a development corpus and allows us to obtain translation of documents almost instantly.[8]

Results for this untuned configuration (see on-demand in Table 2) are lower by 5.5 BLEU point (BP) than those of the conventionally tuned moses system, which can be mostly attributed to the absence of tuning. However, translations for the test set are delivered much faster, where our system is x36 times wall clock faster than moses.

[7]We used the moses decoder in our experiments, whose default parameters are: 0.3 for all 7 reordering features, including 6 lexical reordering features 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.

[8]In this work, the language model is still pre-trained. Future work will include the incremental / on-demand estimation of language models [15].

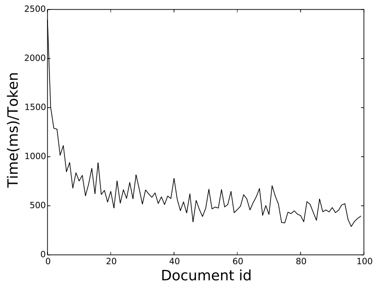


Figure 1: Evolution of the average per token processing time for a sequence of documents.

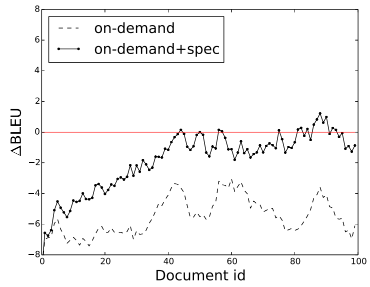


Figure 2: Document-level comparison with `moses` system in English-to-French translation direction. The y-axis represents the difference in BLEU score (ΔBLEU) between our systems and the vanilla `moses` system for each document in the sequence.

As mentioned before, the computed word alignments are cached and are available for translating subsequent documents. To further analyze the effect of the cache, Figure 1 shows how the average per token processing time decreases as more and more documents from the same flow 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, the average estimation time continues to decrease, albeit at a slower pace.

3.2.2. *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.[9] Our on-demand framework makes it natural and straightforward to integrate any newly available parallel data without any full retraining.

In the following experiment, each newly available `Cochrane` document is added to a "specialized" corpus, denoted by `spec`. A separate phrase table for each document is estimated from `spec` using Algorithm 1; considering the very small size of our specialized source, the corresponding phrase table, built from previous documents in the sequence $\{\mathbf{d}_i, i = 1 \ldots t-1\}$, 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 being thus 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.

[9] In fact, the `Cochrane` dataset used in this study is made of two parts: a large portion of the data was translated by human translator from scratch, while a smaller amount a document where actually produced through post-edition. We still use this data as a post-edited corpus in our experiments, although these two kinds of data are slightly different. We believe this does not affect our experimental conclusions [16].

Results in Table 2 show that the additional table (`+spec`) helps to significantly improve translation quality over the raw `on-demand` 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 the phrase table improves over time.

Figure 2 shows the document-level comparison between our systems (`on-demand` and `on-demand+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 `on-demand` and `on-demand+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 `on-demand` systematically underperforms `moses` on all documents, which was expected given the gap in corpus-level performance. Interestingly, the use of the specialized phrase table, `on-demand+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 much 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.

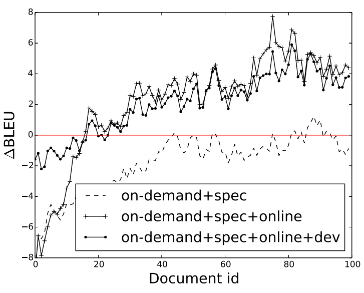


Figure 3: Document-level comparison with `moses` system in English-to-French translation direction. Initialization either uses `moses` default values (`+online`), or parameters tuned on a development set (`+dev`).

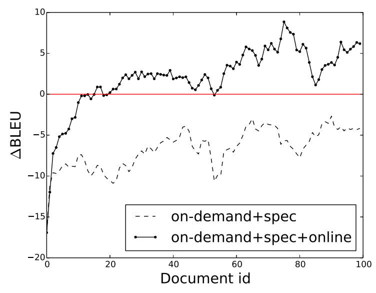


Figure 4: Document-level comparison with `moses` system in French-to-English translation direction.

3.2.3. *Simple online tuning (`+online`)*

We have previously shown 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 (here `KBMIRA`) on the just translated and post-edited document; the resulting parameter values are then averaged with the parameter values of the 10 previous documents[10], and then used for translating the next document. Additionally, in order to leverage the `spec` table, we also allow here the `spec` phrase table to compete with the phrase table estimated from the static corpus [17] instead of having the latter take precedence.

Results for this last configuration are given in Table 2 (`+online`). Our simple online tuning yields a significant improvement (+4.1 BP) over the untuned `on-demand+spec` configuration. Even though 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 3) 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 takes 3mn (wall clock time) on average for each document in this experiment.

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 2 (`+dev`): using tuned parameters to initialize the system yields no significant change on translation quality. Comparing to the `on-demand+spec+online` system, BLEU by 0.2 points but TER is better by 0.3 points. The document-level comparison in Figure 3 shows, as expected, that initializing with parameters tuned on the development set yields better performance than `on-demand+spec+online` at the start 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.

Finally, we also performed these experiments on the French-to-English translation direction, and the corresponding document-level results are shown in Figure 4. First, for the `on-demand+spec` system, we 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 3). This is probably related to the diversity of the language: indeed, the 100 `Cochrane` bilingual documents contain 3 854 unique English words and 4 398 unique French words. A larger vocabulary implies a lower repetition rate, which makes `+spec` less beneficial. When applying online tuning, our best system (`on-demand+spec+online`) again improves quickly and outperforms the `moses` baseline after less than 20 documents have been translated.

3.3. Any-text translation

In this section, we consider a comparatively less studied, albeit somewhat more realistic, scenario, 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

[10]We restrict to the more recent documents to make tuning more reactive to changes in the quality of the `spec` table.

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.
- 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 scenario *any-text translation*.

As presented above, experiments are performed on 8 documents from various domains (see Table 1). Each document is translated independently, sentence by sentence. Translation rules are extracted from the training corpus for each sentence using an adapted version of Algorithm 1, where each sentence is treated as a single document.

We also make the same assumption as in Section 3.2 that after each sentence has been automatically translated, a reference translation is made available by a human translator (simulating a post-edition scenario, even though the documents used in this section have not been post-edited). These translated and reference data are used to update both the models and parameters for the next sentences. In this study, 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`[11]).

Again, we chose the large-scale corpus `WMT` (see Table 1) as the training data and the vanilla `moses` system as our baseline. Since no development set is available, we chose to use the decoder's default parameters as initial parameters for decoding. As for the target language model, a general-domain LM was used which was trained on the `WMT` corpus. Since the `WMT` corpus contains very large quantities of data from different domains, this LM could be considered as a reasonable general-domain LM.

Experimental results are presented in Table 3, where `moses` is the baseline system, `on-demand` represents our on-demand SMT system, and `+indoc` 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, `moses` is also better than `on-demand` on most documents (5 out of 8 documents). We attribute this result to the effect of sampling and the differences in word alignments: our models are estimated based on a subset of translation examples while the models in `moses` system are estimated based on all examples in the corpus and our on-demand word alignments are probably a little worse than the `mgiza++` word alignments 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` for BLEU on most documents (6 out of 8 documents). Apparently, such improvements depend on the repetitiveness and the length of documents.

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
`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

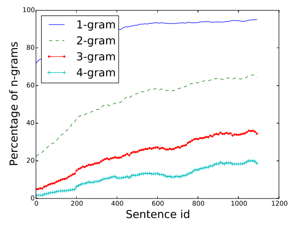
Table 3: Any-text machine translation results for English-to-French translation.

In this use case, it is also possible to perform parameter tuning during the translation of individual documents. Unlike the situation in Section 3.2, where the translation unit was the document, here one sentence contains too little information to perform parameter tuning. Hence, instead, we chose to perform parameter tuning after small batches (of size 100 in our experiments) have been translated. In this experiment, the first sentences of a document are always translated using the decoder's default parameters. After each has batch been translated, the corresponding references are made available and used as a development set to tune the parameters, again with `KBMIRA`. The updated parameters are then used to translate subsequent sentences. In order to assess the effect of parameter tuning on translation results, we only apply the tuning process to a few long documents ($>$ 1000 sentences): `book1`, `book2` and `php`.

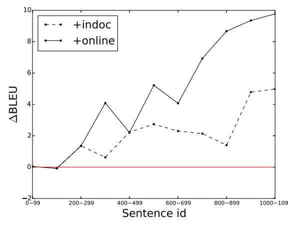
For `book1`, applying parameter tuning after each group of 100 sentences for the `+indoc` system yields a further improvement of $+2.4$ BP and -0.1 TP. On `book2`, the result is less clear: the BLEU score is improved by $+0.3$ BP comparing to the `+indoc` system, but the TER score becomes worse by $+1.8$ 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 document-level analyses on these results. Figure 5a shows the percentage of n-grams occurring in sentence $\mathbf{s}_t$ that were also seen in the previous $t-1$ sentences $\{\mathbf{s}_i, i = 1 \ldots t-1\}$. 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 5b 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

[11] Actually, `indoc` is similar to previous spec phrase table, but `indoc` is estimated based on translated data in the same document.

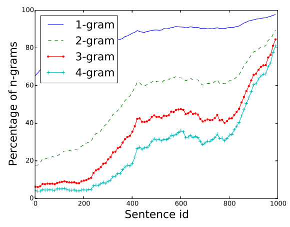


(a) n-gram repetition

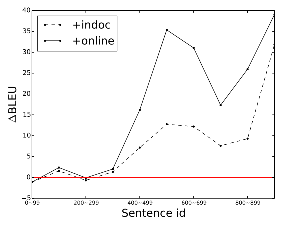


(b) comparison with `moses`

Figure 5: Experimental results on `book1`.

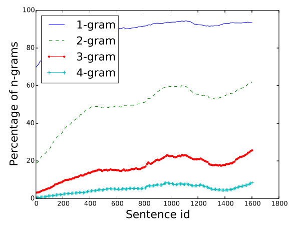


(a) n-gram repetition

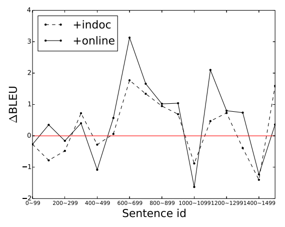


(b) comparison with `moses`

Figure 7: Experimental results on the `php` document.



(a) n-gram repetition



(b) comparison with `moses`

Figure 6: Experimental results on `book2`.

of 100 sentences are tuned on the previous 100 sentences.[12] As shown in Figure 5b, the `+indoc` system takes advantage of the repetitiveness of the document and its performance is systematically better 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 6a, the n-gram repetition rate is lower than that of `book1`, especially for 3-grams and 4-grams. For instance, less than 10% of the 4-grams occurring in sentences at the end of the document, were seen in previous passages. The effect of the low repetitiveness of the document is also reflected on the corpus-level evaluation (see Table 3), where adding the `indoc` phrase table only improves performance by $+0.9$ BP, which compares poorly with the ($+2.2$ BP) improvement observed for `book1`. In this situation, parameter tuning does not always improve translation performance (only in 11 out of 15 sentence groups), and sometimes even proves detrimental to translation quality (see Figure 6b). This result may be related to overfitting issues and suggests to use more sophisticated online adaptation strategies.

Finally, on `php`, the results are much clearer. As shown in Figure 7a, 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 7b, where the `+indoc` system improves very quickly 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.

In this series of experiments, we have demonstrated that 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.

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

4. Related Work

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, making it practical to integrate system development within interactive human post-editing.

Interactive Machine Translation (IMT) was pioneered by projects such as TransType [18], where an 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 [19, 20] and to integrate the result of the interaction to influence future choices of the system. More recently, online learning was introduced in the IMT framework [21] to improve the exploitation of the translator's feedback.

A similar idea was also presented in [22]. In this work, the input document is processed sentence by sentence. After the translation of each sentence, the MT output and the post-edited translation are analyzed and used to extract post-editing rules. These rules are then used to automatically process the MT output so as to improve the quality of output translations.

5. Conclusion

This work 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 incremental strategies, reductions of computation time up to 36 times were obtained relative to a state-of-the-art system trained in a conventional fashion. Fast integration of newly available data in conjunction with online tuning allowed us to quickly reach the same performance as a strong baseline.

We lastly want to underline that scenarios based on the `+spec` characteristic make simpler assumptions than traditional interactive MT (e.g. [18, 19, 21]), as parameter updates are synced to the stream of incoming documents. In addition, as illustrated in Section 3.3, 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.

6. References

[1] C. Dyer, V. Chahuneau, and N. A. Smith, "A simple, fast, and effective reparameterization of IBM model 2," in *Proceedings of NAACL-HLT*, Atlanta, USA, 2013, pp. 644–648.

[2] S. Green, S. Wang, D. Cer, and C. D. Manning, "Fast and adaptive online training of feature-rich translation models," in *Proceedings of ACL*, Sofia, Bulgaria, 2013, pp. 311–321.

[3] G. Gascó, M.-A. Rocha, G. Sanchis-Trilles, J. Andrés-Ferrer, and F. Casacuberta, "Does more data always yield better translations?" in *Proceedings of EACL*, Avignon, France, 2012, pp. 152–161.

[4] C. Callison-Burch, C. Bannard, and J. Schroeder, "Scaling phrase-based statistical machine translation to larger corpora and longer phrases," in *Proceedings of ACL*, Ann Arbor, USA, 2005, pp. 255–262.

[5] A. Lopez, "Tera-scale translation models via pattern matching," in *Proceedings of COLING*, Manchester, UK, 2008, pp. 505–512.

[6] A. Levenberg, C. Callison-Burch, and M. Osborne, "Stream-based translation models for statistical machine translation," in *Proceedings of HLT-NAACL*, Los Angeles, USA, 2010, pp. 394–402.

[7] U. Manber and G. Myers, "Suffix arrays: A new method for on-line string searches," in *Proceedings of SODA*, Philadelphia, USA, 1990, pp. 319–327.

[8] L. Gong, A. Max, and F. Yvon, "Improving bilingual sub-sentential alignment by sampling-based transpotting," in *Proceedings of IWSLT*, Heidelberg, Germany, 2013.

[9] A. Lardilleux, F. Yvon, and Y. Lepage, "Hierarchical Sub-sentential Alignment with Anymalign," in *Proceedings of EAMT*, Trento, Italy, 2012, pp. 280–286.

[10] D. Wu, "Stochastic inversion transduction grammars and bilingual parsing of parallel corpora," *Computational Linguistics*, vol. 23, no. 3, pp. 377–403, 1997.

[11] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of ACL*, Prague, Czech Republic, 2007, pp. 177–180.

[12] C. Cherry and G. Foster, "Batch tuning strategies for statistical machine translation," in *Proceedings of NAACL-HLT*, Montréal, Canada, 2012, pp. 427–436.

[13] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *Proceedings of ACL*, Philadelphia, USA, 2002, pp. 311–318.

[14] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of AMTA*, Boston, USA, 2006, pp. 223–231.

[15] A. Levenberg and M. Osborne, "Stream-based randomised language models for SMT," in *Proceedings of EMNLP*, Singapore, 2009, pp. 756–764.

[16] M. Denkowski, C. Dyer, and A. Lavie, "Learning from post-editing: Online model adaptation for statistical machine translation," in *Proceedings of EACL*, Gothenburg, Sweden, 2014, pp. 395–404.

[17] P. Koehn and J. Schroeder, "Experiments in domain adaptation for statistical machine translation," in *Proceedings of WMT*, Prague, Czech Republic, 2007, pp. 224–227.

[18] P. Langlais, G. Foster, and G. Lapalme, "TransType: a computer-aided translation typing system," in *Proceedings of NAACL-ANLP*, Seattle, USA, 2000, pp. 46–51.

[19] S. Barrachina, O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. Lagarda, H. Ney, J. Tomás, E. Vidal, and J.-M. Vilar, "Statistical approaches to computer-assisted translation," *Computational Linguistics*, vol. 35, no. 1, pp. 3–28, Mar. 2009.

[20] P. Koehn, "A web-based interactive computer aided translation tool," in *Proceedings of the ACL-IJCNLP Software Demonstrations*, Singapore, 2009, pp. 17–20.

[21] D. Ortiz-Martínez, I. García-Varea, and F. Casacuberta, "Online learning for interactive statistical machine translation," in *Proceedings of HLT-NAACL*, Los Angeles, USA, 2010, pp. 546–554.

[22] M. Simard and G. Foster, "Pepr: Postedit propagation using phrase-based statistical machine translation," in *Proceedings of MT Summit*, Nice, France, 2013, pp. 191–198.