Online Learning for Statistical Machine Translation

Daniel Ortiz-Martínez*
Universitat Politècnica de València[1]

We present online learning techniques for statistical machine translation (SMT). The availability of large training data sets that grow constantly over time is becoming more and more frequent in the field of SMT—for example, in the context of translation agencies or the daily translation of government proceedings. When new knowledge is to be incorporated in the SMT models, the use of batch learning techniques require very time-consuming estimation processes over the whole training set that may take days or weeks to be executed. By means of the application of online learning, new training samples can be processed individually in real time. For this purpose, we define a state-of-the-art SMT model composed of a set of submodels, as well as a set of incremental update rules for each of these submodels. To test our techniques, we have studied two well-known SMT applications that can be used in translation agencies: post-editing and interactive machine translation. In both scenarios, the SMT system collaborates with the user to generate high-quality translations. These user-validated translations can be used to extend the SMT models by means of online learning. Empirical results in the two scenarios under consideration show the great impact of frequent updates in the system performance. The time cost of such updates was also measured, comparing the efficiency of a batch learning SMT system with that of an online learning system, showing that online learning is able to work in real time whereas the time cost of batch retraining soon becomes infeasible. Empirical results also showed that the performance of online learning is comparable to that of batch learning. Moreover, the proposed techniques were able to learn from previously estimated models or from scratch. We also propose two new measures to predict the effectiveness of online learning in SMT tasks. The translation system with online learning capabilities presented here is implemented in the open-source Thot toolkit for SMT.

1. Introduction

Multiplicity of languages is inherent to modern society. Phenomena such as globalization and technological development have extraordinarily increased the need for translating information from one language to another. One possibility to deal with this growing demand of translations is the use of machine translation (MT) techniques.

* PRHLT Research Center, Universitat Politècnica de València, 46071, València, Spain, E-mail: dortiz@prhlt.upv.es.
1 The author is now at Webinterpret.

Submission received: 2 May 2014; revised version received: 2 October 2015; accepted for publication: 20 November 2015.

doi:10.1162/COLI_a_00244

MT can be formalized under a statistical point of view as the process of finding the sentence of maximum probability in the target language given the source sentence. Statistical MT (SMT) requires the availability of parallel texts to estimate the statistical models involved in the translation. It is also important that such parallel texts belong to the same domain the system will be used for. These kinds of texts are referred to as **in-domain** corpora in the domain adaptation literature. However, in-domain corpora are often not available in real translation scenarios, forcing us to estimate the system models by means of large out-of-domain texts, such as Parliament proceedings. Unfortunately, this results in a significant degradation in the translation quality (Irvine et al. 2013).

There are many real translation scenarios in which new training data is inherently generated over time (e.g., translation agencies or the daily translation of government proceedings). The newly generated training data could be used to mitigate the problem of data scarcity. However, this situation poses new challenges in the SMT framework, because the vast majority of the SMT systems described in the literature makes use of the well-known **batch learning paradigm**. In the batch learning paradigm, the training of the SMT system and the translation process are carried out in separate stages. This implies that all training samples must be available before training takes place, preventing the statistical models to be extended when the system starts generating translations. To solve this problem, the **online learning** paradigm can be applied. Online learning is a machine learning task that is structured in a series of trials, where each trial has four steps: (1) the learning algorithm receives an instance, (2) a label for the instance is predicted, (3) the true label for the instance is presented, and (4) the learning algorithm uses the true label to update its parameters. In this paradigm, the training and prediction stages are no longer separated.

Online learning fits nicely in typical computer-assisted translation (CAT) applications used in translation agencies. This is because in such applications the system translation for each source sentence is validated by a human expert and thus can be used to produce new training pairs. One possible CAT implementation consists of post-editing (PE) the output of an MT system. In this implementation, the MT system generates an initial translation that is corrected by the user without further system intervention. Another instance of CAT is interactive machine translation (IMT), where the user generates each translation in a series of interactions with the system.

Scientific and commercial interest in CAT applications has greatly increased during recent years, capturing the attention of internationally renowned research groups and translation companies. A good example of this is the work carried out in the TransType (Foster, Isabelle, and Plamondon 1997) and TransType-II (SchlumbergerSema S.A. et al. 2001) research projects, where the IMT paradigm was developed, and the CasMaCat (Alabau et al. 2014) and MateCat (Federico et al. 2014) projects, where a substantial part of the effort was focused on developing adaptive learning techniques for CAT. Literature also offers demonstrations of CAT applications (Koehn 2009; Ortiz-Martínez et al. 2011) as well as studies involving real users showing the potential benefits of CAT (Green, Heer, and Manning 2013; Ortiz-Martínez et al. 2015).

In this work we propose online learning techniques for SMT useful to efficiently update the statistical models used by the system, avoiding the necessity of executing costly retraining processes. The properties of the proposed techniques will be tested in the two CAT scenarios we have mentioned. As noted earlier, in such scenarios there is a human translator that supervises each system translation. However, it is important to remark that our proposed techniques can also work in scenarios where there are no human experts involved. One example of this can be found in fully automatic translation tasks where the initial models can be extended from

new blocks of training data obtained from different sources, such as parliamentary proceedings.

The rest of the article is organized as follows: Section 2 describes the statistical foundations of SMT and its adaptation to the PE and IMT scenarios. Section 3 explains the online learning techniques proposed here, including the definition of a log-linear SMT model as well as a set of incremental update rules for each one of its components. The content of Section 3 is complemented by Appendix A, which presents an alternative incremental update rule for word-alignment models. Experimental results as well as their discussion are shown in Sections 4 and 5, respectively. Section 6 describes related work on online learning. The work conclusions are given in Section 7.

2. Statistical Framework

In this section we describe the details of the statistical framework adopted in the rest of the article. For this purpose, we briefly describe the statistical formulation of SMT as well as the required modifications for its use in two well-known applications of SMT, namely, post editing and interactive machine translation.

2.1 Statistical Machine Translation

In the statistical approach to MT, given a source sentence $f_1^J \equiv f_1 ... f_j ... f_J$ in the source language $\mathcal{F}$, we want to find its equivalent target sentence $e_1^I \equiv e_1 ... e_i ... e_I$ in the target language $\mathcal{E}$, where f_j and e_i note the ith word and the jth word of the sentences f_1^J and e_1^I, respectively. From the set of all possible sentences of the target language, we are interested in that with the highest probability according to the following equation:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I, e_1^I} \{Pr(e_1^I | f_1^J)\} \tag{1}$$

Early works on SMT decompose $Pr(e_1^I | f_1^J)$ applying Bayes' theorem and thus obtaining two new distributions, $Pr(e_1^I)$ and $Pr(f_1^J | e_1^I)$, which are approximated by means of parametric statistical models. Specifically, $Pr(e_1^I)$ is modeled by means of a *language model*, and $Pr(f_1^J | e_1^I)$ is modeled by means of a **translation model**.

Statistical language models are typically implemented with ***n*-gram language models**. Regarding the translation models, they are commonly implemented using the so-called **phrase-based models** (Koehn, Och, and Marcu 2003). The basic idea of phrase-based translation is to segment the source sentence into phrases, then to translate each source phrase into a target phrase, and finally to reorder the translated target phrases in order to compose the target sentence. The decisions made during the phrase-based translation process can be summarized by means of the hidden variable $\tilde{a}_1^K \equiv \tilde{a}_1 ... \tilde{a}_k ... \tilde{a}_K$, where $\tilde{a}_k$ denotes the index of the target phrase $\tilde{e}_k$ that is aligned with the kth source phrase $\tilde{f}_k$, determining a **bisegmentation** of the source and target sentences of length K.

Alternative formalizations to the one using Bayes' theorem have been proposed. Such formalizations are based on the direct modeling of the posterior probability, $Pr(e_1^I | f_1^J)$, by means of the so-called **log-linear models** for SMT (Och and Ney 2002). Log-linear models use a set of feature functions $h_r(f_1^J, e_1^I)$ each one with its corresponding weight λ_r, which are typically estimated by means of the well-known **minimum error rate training** (MERT) algorithm (Och 2003). Common log-linear model implementations

are strongly focused on these phrase-based models, obtaining the best alignment at phrase level:

$$\hat{e}_1^{\hat{I}} = \arg\max_{I, e_1^I} \left\{ \max_{K, \tilde{a}_1^K} \sum_{r=1}^{R} \lambda_r h_r(f_1^J, e_1^I, \tilde{a}_1^K) \right\} \quad (2)$$

where a total of R feature functions are assumed.

State-of-the-art decoders work by exploring the search space determined by Equation (2) using iterative algorithms that build partial target translations from left to right.

2.2 Post-Editing the Output of Statistical Machine Translation

Post-editing (PE) involves making corrections to machine generated translations (see TAUS-Project [2010] for a detailed study). PE is used when raw machine translation is not error-free, a common situation for current MT technology. PE tends to be carried out via tools built for editing human-generated translations, such as translation memories (some authors refer to this task as simply **editing**). Because in the PE scenario, the user only edits the output of the MT system without further intervention from the system, there are no differences in the way in which the MT system is designed and implemented. Hence, the statistical framework for MT described previously can be adopted without modifications in order to build the PE system.

2.3 Statistical Interactive Machine Translation

One alternative to the serial collaboration model adopted by PE is interactively combining the MT system with a human translator, constituting the interactive machine translation (IMT) paradigm (also referred to as interactive translation prediction). One possible IMT implementation uses SMT systems to produce target sentence hypotheses that can be partially or completely accepted and amended by a human translator (Barrachina et al. 2009). Each partially corrected text segment, or prefix, is then used by the SMT system as additional information to achieve improved suggestions.

Figure 1 illustrates a typical IMT session. In interaction-0, the system suggests a complete translation hypothesis, $\mathbf{e}_s$, given the source sentence, f_1^J. In interaction-1, the user moves the mouse to accept the prefix composed of the first eight characters *To view* (that is, the prefix of the sentence the user deems to be correct) and presses the **a** key (k), producing the prefix, $\mathbf{e}_p$. Then the system suggests completing the sentence with *list of resources* (a new $\mathbf{e}_s$), given the accepted and correct prefix. Interactions-2 and -3 are similar. In the final interaction, the user accepts the current translation suggestion.

In the IMT scenario we have to find an extension $\mathbf{e}_s$ for a user prefix $\mathbf{e}_p$:[2]

$$\hat{\mathbf{e}}_s = \arg\max_{\mathbf{e}_s} \{p(\mathbf{e}_s \,|\, f_1^J, \mathbf{e}_p)\} \quad (3)$$

If Bayes' theorem is applied, we then obtain two distributions, $p(\mathbf{e}_s \,|\, \mathbf{e}_p)$ and $p(f_1^J \,|\, \mathbf{e}_p, \mathbf{e}_s)$, that are very similar to those obtained for conventional SMT, since $\mathbf{e}_p\mathbf{e}_s \equiv e_1^I$. This allows us to use the same models if the search procedures are adequately

2 Note: in Figure 1, the prefix $\mathbf{e}_p$ also includes the keys, k, that are pressed by the user.

source(f_1^J): Para ver la lista de recursos
reference($\hat{e}_1^I$): To view a listing of resources

interaction-0	$\mathbf{e}_p$						
	$\mathbf{e}_s$	*To*	*view*	*the*	*resources*	*list*	
interaction-1	$\mathbf{e}_p$	To	view				
	k			**a**			
	$\mathbf{e}_s$				*list*	*of*	*resources*
interaction-2	$\mathbf{e}_p$	To	view	a	list		
	k				**i**		
	$\mathbf{e}_s$				*ng*	*resources*	
interaction-3	$\mathbf{e}_p$	To	view	a	listing		
	k					**o**	
	$\mathbf{e}_s$					*f*	*resources*
acceptance	$\mathbf{e}_p$	To	view	a	listing	of	resources

Figure 1
An example of an IMT session to translate a Spanish sentence into English.

modified. Specifically, the search is restricted to generate target sentences compatible with the user prefix.[3] Note that the statistical models are defined at the word level whereas the IMT interface described in Figure 1 works at the character level. This is not an important issue because the transformations that are required in the statistical models for their use at character level are trivial. Specifically, the compatibility with the user prefix is verified by comparing characters instead of words.

3. Online Learning for Statistical Machine Translation

In this section we describe the concept of online learning and its application to SMT.

3.1 Definition of Online Learning

Online learning algorithms proceed in a sequence of trials. Each trial can be decomposed into four steps:

1. The learning algorithm receives an instance.
2. The learning algorithm predicts a label for the instance according to its current parameters.
3. The true label of the instance is presented to the learning algorithm.
4. The learning algorithm uses the true label to update its parameters.

The system uses the true label to measure the prediction error incurred by the learner and discarded afterwards. The ultimate goal of the online learning algorithm is to minimize the cumulative prediction error along its run by modifying its parameters.

More formally, given any sequence of training samples $\mathbf{x}_1, \mathbf{x}_2, \ldots$, an online learning algorithm produces a sequence of parameters: $\Theta^{(0)}, \Theta^{(1)}, \Theta^{(2)}, \ldots$, such that the algorithm parameters at trial t, $\Theta^{(t)}$, depends only on the previous parameters, $\Theta^{(t-1)}$, and the current sample $\mathbf{x}_t$.

3 This also includes the application of the log-linear approach given by Equation (2).

One important consequence of discarding the training samples after each trial is that the computational complexity of processing a new sample does not depend on the number of samples that has been previously seen. That is, the computational complexity of processing a new sample is constant.

The online learning algorithms that discard each new training sample after updating the learner are also referred to as **incremental learning algorithms** by some authors (see Anthony and Biggs [1992]). However, this constraint can be relaxed by using mini-batches (small sets of samples).

The online learning setting contrasts with the batch learning setting, in which all the training patterns are presented to the learner before learning takes place and the learner is no longer updated after the learning stage has concluded.

Batch learning algorithms are appropriate for their use in stationary environments. In a stationary environment, all instances are drawn from the same underlying probability distribution. By contrast, because online learning algorithms continually receive prediction feedback, they can be used in non-stationary environments.

The design of online learning algorithms raises issues not present in batch learning settings. Three of them are identified in Giraud-Carrier (2000): (1) *Chronology*: the order in which knowledge is acquired is an inherent aspect of online learning, (2) *Learning curve*: the learner may start from scratch and gain knowledge from examples given one at a time over time; as a result, it experiences a sort of learning curve, and (3) *Open-world assumption*: all the data relevant to the problem at hand is not available a priori.

Finally, online learning is also related to another learning paradigm: active learning. In this paradigm the system queries the user to obtain the true labels of specific instances, obtaining greater accuracy using less training data. Active learning can also be applied in online settings, where the capability of the system to learn in an online or incremental manner using techniques like those proposed here is crucial. One example of this is the work presented in González-Rubio, Ortiz-Martínez, and Casacuberta (2012), where active learning techniques for IMT are proposed.

3.2 Implementing Online Learning

The key aspect to be considered when implementing online learning algorithms is how to update the system parameters given the previous ones and the new training sample. If the online learning algorithm is based on statistical models, then we need to maintain a set of **sufficient statistics** for these models that can be incrementally updated. A sufficient statistic for a statistical model is a statistic that captures all the information that is relevant to estimate this model. If the estimation of the statistical model does not require the use of the expectation–maximization (EM) algorithm (e.g. n-gram language models), then it is generally easy to incrementally extend the model given a new training sample. By contrast, if the EM algorithm is required (e.g., word alignment models), the estimation procedure has to be modified, since conventional EM is designed for its use in batch learning scenarios. To solve this problem, an incremental version of the EM algorithm is required.

3.3 Predicting the Effectiveness of Online Learning in SMT Tasks

According to the work presented in Irvine et al. (2013), the presence of unknown source words and known source words with unknown translations explains the majority of the performance degradation when an SMT system is migrated to a new domain. The use of online learning can mitigate these two problems, because the system is

now able to efficiently learn translations for new or previously seen words. However, the benefits will only be significant when the document to be translated presents a high internal repetition rate, since this will allow the system to take advantage of the newly acquired knowledge. This should not be seen as a limitation specific to online learning. In batch learning scenarios the translation quality is strongly weakened if the training corpus is not representative of the text to be translated. When we move to an online setting, we still have the same requirement but now the training and translation stages are no longer separated. This is why we speak about repetitiveness instead of representativeness. In any case, sufficiently high repetition rates for test documents are common, according to the document-internal repetition property defined in Church and Gale (1995).

Bertoldi, Cettolo, and Federico (2013) propose an automatic measure for assessing the potential usefulness of online learning: the repetition rate (RR). In this section we will slightly modify the definition of RR and propose two additional measures.

The RR measure looks at the rate of non-singleton n-grams contained in a given text. More specifically, the rates of non-singleton n-grams from $n = 1$ to 4 are calculated and geometrically averaged, using a sliding window of $1,000$ words to make the rates comparable across different sized corpora. Here we use a slightly modified version in which the sliding window calculation is removed, because in real translation scenarios the text to be translated is available beforehand and should be completely translated. Thus, we define our modified RR (MRR) measure as follows:

$$\mathrm{MRR}(\mathcal{I}) = \left(\prod_{n=1}^{4} \frac{|\mathcal{I}_{n,1+}| - |\mathcal{I}_{n,1}|}{|\mathcal{I}_{n,1+}|} \right)^{1/4} \tag{4}$$

where $|\cdot|$ represents the length of a given set, $\mathcal{I}_{n,1+}$ represents the set of different n-grams contained in the in-domain corpus $\mathcal{I}$, and $\mathcal{I}_{n,1}$ represents the set of different n-grams occurring only once in $\mathcal{I}$.

The MRR measure does not take into account whether a specific n-gram is contained or not in the out-of-domain corpus that has been used to estimate the SMT models. According to Irvine et al. (2013), unseen events constitute a major cause of translation errors when migrating an existing SMT system to a new domain. Thus, it is interesting to restrict the calculation of the repetition rate to those n-grams that are not contained in the out-of-domain corpus. We will refer to this measure as the **restricted repetition rate** (RRR):

$$\mathrm{RRR}(\mathcal{I}, \mathcal{O}) = \left(\prod_{n=1}^{4} \frac{|\mathcal{I}_{n,1+} - \mathcal{O}_{n,1+}| - |\mathcal{I}_{n,1} - \mathcal{O}_{n,1+}|}{|\mathcal{I}_{n,1+} - \mathcal{O}_{n,1+}|} \right)^{1/4} \tag{5}$$

where $\mathcal{O}_{n,1+}$ represents the set of different n-grams contained in the out-of-domain corpus $\mathcal{O}$.

The RRR measure reflects how frequently unseen n-grams are repeated in the corpus to be translated. However, such unseen n-grams constitute only a fraction of the in-domain corpus. A high value of the RRR is not enough to predict good results

if the fraction of unseen n-grams is very low. To capture this corpus property, we define the **unseen n-gram fraction** (UNF) measure:

$$\mathrm{UNF}(\mathcal{I},\mathcal{O}) = \left(\prod_{n=1}^{4} \frac{\sum_{\mathbf{w}\in(\mathcal{I}_{n,1+}-\mathcal{O}_{n,1+})} c_{\mathcal{I}}(\mathbf{w})}{\sum_{\mathbf{w}\in\mathcal{I}_{n,1+}} c_{\mathcal{I}}(\mathbf{w})} \right)^{1/4} \tag{6}$$

where $c_{\mathcal{I}}(\mathbf{w})$ represents the count of n-gram $\mathbf{w}$ in corpus $\mathcal{I}$.

Here we propose to predict the potential usefulness of online learning, paying attention only to the RRR and the UNF measures. In spite of this, MRR will be also reported so as to compare the information provided by the three measures.

3.4 Statistical Phrase-Based Log-Linear Model for Online SMT

As stated in Section 2.1, log-linear models including phrase-based models as feature functions constitute the state-of-the-art in statistical machine translation. In this section we will describe the components of our log-linear model for SMT. Later, in Section 3.5, the update rules required to extend such components will be presented.

According to Equation (2), we introduce a set of seven feature functions in our log-linear model: an n-gram language model (h_1), a source sentence-length model (h_2), inverse and direct phrase-based models (h_3 and h_4, respectively), a target phrase-length model (h_5), a source phrase-length model (h_6), and a distortion (or phrase reordering) model (h_7). All these feature functions, with the exception of the one related to the direct phrase-based model (h_4), can be obtained from a proper decomposition of the distribution $Pr(e_1^I|f_1^J)$ (the decomposition is detailed in Ortiz-Martínez [2011]).

This set of feature functions is similar to those incorporated into other state-of-the-art SMT systems such as the Moses decoder (Koehn et al. 2007). The main difference of our proposal with existing works is that we have paid special attention to the formal justification of the features.

We next list the details for each feature function.

- **n-gram target language model (h_1)**[4]

$$h_1(e_1^I) = \log \left(\prod_{i=1}^{I+1} p(e_i|e_{i-n+1}^{i-1}) \right)$$

4 I is the length of e_1^I, e_0 is the **begin-of-sentence** symbol, $e_{|e|+1}$ is the **end-of-sentence** symbol, $e_i^j \equiv e_i ... e_j$

where $p(e_i|e_{i-n+1}^{i-1})$ is defined as follows:

$$p(e_i|e_{i-n+1}^{i-1}) = \frac{\max\{c_X(e_{i-n+1}^{i}) - D_n, 0\}}{c_X(e_{i-n+1}^{i-1})} + \frac{D_n}{c_X(e_{i-n+1}^{i-1})} N_{1+}(e_{i-n+1}^{i-1}\bullet) \cdot p(e_i|e_{i-n+2}^{i-1}) \quad (7)$$

where $D_n = \frac{c_{n,1}}{c_{n,1}+2c_{n,2}}$ is a fixed discount ($c_{n,1}$ and $c_{n,2}$ are the number of n-grams with one and two counts respectively), $N_{1+}(e_{i-n+1}^{i-1}\bullet)$ is the number of unique words that follows the history e_{i-n+1}^{i-1}, and $c_X(e_{i-n+1}^{i})$ is the count of the n-gram e_{i-n+1}^{i}, where $c_X(\cdot)$ can represent true counts $c_T(\cdot)$ or modified counts $c_M(\cdot)$.

True counts are used for the higher order n-grams and modified counts for the lower order n-grams. Given a certain n-gram, its modified count consists of the number of different words that precede this n-gram in the training corpus.

Equation (7) corresponds to the probability given by an n-gram language model with an interpolated version of the Kneser-Ney smoothing (Chen and Goodman 1996).

- **source sentence-length model (h_2)**

$$h_2(f_1^J, e_1^I) = \log(p(J \mid I)) = \log(\phi_I(J + 0.5) - \phi_I(J - 0.5))$$

where $\phi_I(\cdot)$ is the cumulative distribution function for the normal distribution (the cumulative distribution function is used to integrate the normal density function over an interval of length 1). A specific target sentence length I will be assigned during decoding time when a new empty hypothesis is created. After that, this hypothesis will be extended in successive trials, but it will be constrained to have I words when all the source words are covered. The sentence length model is introduced to avoid the generation of too short or too long target sentences, which negatively impact translation quality (other authors use models that simply penalize the number of target words).

We use a specific normal distribution with mean μ_I and standard deviation σ_I for each target sentence length I.

- **inverse and direct phrase-based models (h_3, h_4)**

$$h_3(f_1^J, e_1^I, \tilde{a}_1^K) = \log\left(\prod_{k=1}^{K} p(\tilde{f}_k|\tilde{e}_{\tilde{a}_k})\right)$$

where $p(\tilde{f}_k|\tilde{e}_{\tilde{a}_k})$ is defined as follows:

$$p(\tilde{f}_k|\tilde{e}_{\tilde{a}_k}) = \beta \cdot p_{\text{phr}}(\tilde{f}_k|\tilde{e}_{\tilde{a}_k}) + (1 - \beta) \cdot p_{\text{hmm}}(\tilde{f}_k|\tilde{e}_{\tilde{a}_k}) \quad (8)$$

In Equation (8), $p_{\text{phr}}(\tilde{f}_k|\tilde{e}_{\tilde{a}_k})$ denotes the probability given by a statistical phrase-based dictionary used in regular phrase-based models.

$p_{\text{hmm}}(\tilde{f}_k|\tilde{e}_{\tilde{a}_k})$ is the probability given by a hidden Markov model (HMM)-based (intra-phrase) alignment model (see Vogel, Ney, and Tillmann 1996):

$$p_{\text{hmm}}(\tilde{f}|\tilde{e}) = \epsilon \sum_{a_1^{|\tilde{f}|}} \prod_{j=1}^{|\tilde{f}|} p(\tilde{f}_j|\tilde{e}_{a_j}) \cdot p(a_j|a_{j-1}, |\tilde{e}|) \tag{9}$$

The HMM-based alignment model probability is used here for smoothing purposes. Analogously, h_4 is defined as:

$$h_4(f_1^J, e_1^I, \tilde{a}_1^K) = \log\left(\prod_{k=1}^{K} p(\tilde{e}_{\tilde{a}_k}|\tilde{f}_k)\right)$$

- **target phrase-length model (h_5)**

$$h_5(f_1^J, e_1^I, \tilde{a}_1^K) = \log\left(\prod_{k=1}^{K} p(|\tilde{e}_k|)\right)$$

where $p(|\tilde{e}_k|) = \delta(1-\delta)^{|\tilde{e}_k|}$.

h_5 implements a target phrase-length model by means of a geometric distribution with probability of success on each trial δ.

The use of a geometric distribution penalizes the length of target phrases.

- **source phrase-length model (h_6)**

$$h_6(f_1^J, e_1^I, \tilde{a}_1^K) = \log\left(\prod_{k=1}^{K} p(|\tilde{f}_k| \mid |\tilde{e}_{\tilde{a}_k}|)\right),$$

where $p(|\tilde{f}_k| \mid |\tilde{e}_{\tilde{a}_k}|) = \frac{1}{1+\tau}\delta(1-\delta)^{\text{abs}(|\tilde{f}_k|-|\tilde{e}_{\tilde{a}_k}|)}$, $\tau = \sum_{i=1}^{|\tilde{e}_{\tilde{a}_k}|-1} \delta(1-\delta)^i$, and abs$(\cdot)$ is the absolute value function.

A geometric distribution (with scaling factor $\frac{1}{1+\tau}$) is used to model this feature (it penalizes the difference between the source and target phrase lengths).

- **distortion model (h_7)**

$$h_7(\tilde{a}_1^K) = \log\left(\prod_{k=1}^{K} p(\tilde{a}_k|\tilde{a}_{k-1})\right)$$

where $p(\tilde{a}_k|\tilde{a}_{k-1}) = \frac{1}{2-\delta}\delta(1-\delta)^{\text{abs}(b_{\tilde{a}_k} - l_{\tilde{a}_{k-1}})}$, $b_{\tilde{a}_k}$ denotes the beginning position of the source phrase covered by $\tilde{a}_k$ and $l_{\tilde{a}_{k-1}}$ denotes the last position of the source phrase covered by $\tilde{a}_{k-1}$.

A geometric distribution (with scaling factor $\frac{1}{2-\delta}$) is used to model this feature (it penalizes the reorderings).

3.5 Online Update Rules

After translating a source sentence f_1^J, a new sentence pair (f_1^J, e_1^I) is available to feed the SMT system. To do this, a set of sufficient statistics that can be incrementally updated is maintained for the statistical models that implement each feature function $h_r(\cdot)$.

In the following sections, we present the set of sufficient statistics for each model. Regarding the weights of the log-linear combination, they are not modified because of the presentation of a new sentence pair to the system. These weights can be adjusted offline by means of a development corpus and well-known optimization techniques, such as the Powell algorithm or the downhill simplex algorithm, which are commonly used in a typical MERT procedure.

3.5.1 Language Model (h_1). Feature function h_1 implements a language model. According to Equation (7), the following data are to be maintained: $c_{k,1}$ and $c_{k,2}$ given any order k, $N_{1+}(\cdot)$, and $c_X(\cdot)$ (see Section 3.4 for the meaning of each symbol).

Given a new sentence e_1^I, and for each k-gram e_{i-k+1}^i of e_1^I, where $1 \leq k \leq n$ and $1 \leq i \leq I+1$, the set of sufficient statistics is modified, as is shown in Algorithm 1. The algorithm checks the changes in the counts of the k-grams to update the set of sufficient statistics. For a given k-gram, e_{i-k+1}^i, its true count and the corresponding normalizer are updated at lines 13 and 14, respectively. The modified count of the $(k-1)$-gram and its normalizer are updated at lines 7 and 8, respectively, only when the k-gram e_{i-k+1}^i appears for the first time (condition checked at line 2). The value of the $N_{1+}(\cdot)$ statistic for e_{i-k+1}^{i-1} and e_{i-k+2}^{i-1} is updated at lines 10 and 6, respectively, only if the word e_i has been seen for the first time following these contexts. Finally, sufficient statistics for D_k

Algorithm 1 Pseudocode for the `update_suff_stats_lm` algorithm. This algorithm is used to incrementally update the sufficient statistics of a language model with Kneser-Ney smoothing. The meaning of the different symbols is explained in Section 3.5.1.

input : n (higher order), e_{i-k+1}^i (k-gram),
$\mathcal{S} = \{\forall j(c_{j,1}, c_{j,2}), N_{1+}(\cdot), c_X(\cdot)\}$ (current set of sufficient statistics)
output : $\mathcal{S}$ (updated set of sufficient statistics)

1 **begin**
2 **if** $c_T(e_{i-k+1}^i) = 0$ **then**
3 **if** $k - 1 \geq 1$ **then**
4 updD($\mathcal{S}$,k-1,$c_M(e_{i-k+2}^i)$,$c_M(e_{i-k+2}^i) + 1$)
5 **if** $c_M(e_{i-k+2}^i) = 0$ **then**
6 $N_{1+}(e_{i-k+2}^{i-1}) := N_{1+}(e_{i-k+2}^{i-1}) + 1$
7 $c_M(e_{i-k+2}^i) := c_M(e_{i-k+2}^i) + 1$
8 $c_M(e_{i-k+2}^{i-1}) := c_M(e_{i-k+2}^{i-1}) + 1$
9 **if** $k = n$ **then**
10 $N_{1+}(e_{i-k+1}^{i-1}) := N_{1+}(e_{i-k+1}^{i-1}) + 1$
11 **if** $k = n$ **then**
12 updD($\mathcal{S}$,k,$c_T(e_{i-k+1}^i)$,$c_T(e_{i-k+1}^i) + 1$)
13 $c_T(e_{i-k+1}^i) := c_T(e_{i-k+1}^i) + 1$
14 $c_T(e_{i-k+1}^{i-1}) := c_T(e_{i-k+1}^{i-1}) + 1$

Algorithm 2 Pseudocode for the `updD` algorithm. This algorithm is used internally by the `update_suff_stats_lm` algorithm to update the value of the D_k statistic involved in the generation of language model probabilities.

```
  input  : S (current set of sufficient statistics),k (order), c (current count),
           c' (new count)
  output : (c_{k,1}, c_{k,2}) (updated sufficient statistics)
1 begin
2   if c = 0 then
3     if c' = 1 then c_{k,1} := c_{k,1} + 1
4     if c' = 2 then c_{k,2} := c_{k,2} + 1
5   if c = 1 then
6     c_{k,1} := c_{k,1} − 1
7     if c' = 2 then c_{k,2} := c_{k,2} + 1
8   if c = 2 then c_{k,2} := c_{k,2} − 1
```

are updated at lines 12 (for higher order n-grams) and 4 (for lower order n-grams), following the auxiliary procedure shown in Algorithm 2.

3.5.2 Source Sentence Length Model (h_2). Feature function h_2 implements a source sentence length model. h_2 requires the incremental calculation of the mean μ_I and the standard deviation σ_I of the normal distribution associated with a target sentence length I. For this purpose, the procedure described in Knuth (1981) can be used. In this procedure, two quantities are maintained for each normal distribution: μ_I and S_I, where S_I is an auxiliary quantity from which the standard deviation can be obtained, as is explained subsequently. Given the training sample (f_1^J, e_1^I) at trial t, the two quantities are updated according to the following equations:

$$\mu_{i:i\neq I}^{(t)} = \mu_i^{(t-1)} \tag{10}$$

$$\mu_I^{(t)} = \mu_I^{(t-1)} + (J - \mu_I^{(t-1)})/c(I) \tag{11}$$

$$S_{i:i\neq I}^{(t)} = S_i^{(t-1)} \tag{12}$$

$$S_I^{(t)} = S_I^{(t-1)} + (J - \mu_I^{(t-1)}) \cdot (J - \mu_I^{(t)}) \tag{13}$$

where $c(I)$ is the count of the number of sentences of length I that have been seen so far, and $\mu_I^{(t-1)}$ and $S_I^{(t-1)}$ are the quantities previously stored ($\mu_I^{(0)}$ is initialized to the source sentence length of the first sample and $S_I^{(0)}$ is initialized to zero). Finally, the standard deviation can be obtained from $S_I^{(t)}$ as follows: $\sigma_I^{(t)} = \sqrt{S_I^{(t)}/(c(I)^{(t)} - 1)}$.

3.5.3 Inverse and Direct Phrase-Based Models (h_3 and h_4). Feature functions h_3 and h_4 implement inverse and direct phrase-based models, respectively. These phrase-based models are combined with HMM-based alignment models via linear interpolation. In this work we have not studied how to incrementally update the weights of the interpolation. Instead, these weights can be estimated from a development corpus.

Because phrase-based models are symmetric models, only an inverse phrase-based model is maintained. The inverse phrase model probabilities, $p(\tilde{f}|\tilde{e})$, are estimated from phrase counts, c($\tilde{f}$,$\tilde{e}$), as follows:

$$p(\tilde{f}|\tilde{e}) = \frac{c(\tilde{f},\tilde{e})}{\sum_{\tilde{f}'} c(\tilde{f}',\tilde{e})}$$

According to the previous equation, the set of sufficient statistics to be stored for the inverse phrase model consists of a set of phrase counts, $c(\tilde{f},\tilde{e})$.

When processing a new sentence pair (f_1^J, e_1^I), the standard phrase-based model estimation method (see Zens, Och, and Ney [2002] and Koehn, Och, and Marcu [2003] for a detailed explanation) uses a word alignment matrix, A, between f_1^J and e_1^I to extract the set of phrase pairs that are *consistent* with the word alignment matrix: $\mathcal{BP}(f_1^J, e_1^I)$. This consistency relation is formally defined as follows:

$$\mathcal{BP}(f_1^J, e_1^I, A) = \{(f_j^{j+r}, e_i^{i+s}) | \forall (i',j') \in A : j \leq j' \leq j+r \iff i \leq i' \leq i+s\} \quad (14)$$

Hence, the set of consistent phrase pairs is constituted by those bilingual phrases where all the words within the source phrase are only aligned to the words of the target phrase and vice versa.

Given the training pair (f_1^J, e_1^I) at trial t, and after obtaining the set of consistent phrase pairs, $\mathcal{BP}(f_1^J, e_1^I, A)$, the phrase counts are updated as follows:

$$c(\tilde{f},\tilde{e})^{(t)} = c(\tilde{f},\tilde{e})^{(t-1)} + c(\tilde{f},\tilde{e} \,|\, \mathcal{BP}(f_1^J, e_1^I, A)) \quad (15)$$

where $c(\tilde{f},\tilde{e})^{(t)}$ is the current count of the phrase pair $(\tilde{f},\tilde{e})$, $c(\tilde{f},\tilde{e})^{(t-1)}$ is the previous count, and $c(\tilde{f},\tilde{e} \,|\, \mathcal{BP}(f_1^J, e_1^I, A))$ is the count of $(\tilde{f},\tilde{e})$ in $\mathcal{BP}(f_1^J, e_1^I, A)$.

After updating the phrase counts, we need to efficiently compute the phrase translation probabilities. For this purpose, we maintain in memory both the current phrase counts and their normalizers.

One problem to be solved when updating the phrase model parameters is the need to generate word alignment matrices. To solve this problem, we use the direct and inverse HMM-based alignment models that are included in the formulation of the IMT system. Specifically, these models are used to obtain word alignments in both translation directions. The resulting direct and inverse word alignment matrices are combined by means of the **symmetrization** alignment operation (Och and Ney 2003) before extracting the set of consistent phrase pairs.

In order to obtain an IMT system able to robustly learn from user feedback, we also need to incrementally update the HMM-based alignment models. In the following section we show how to efficiently incorporate new knowledge into these models.

3.5.4 Inverse and Direct HMM-Based Alignment Models (h_3 and h_4). HMM-based alignment models play a crucial role in log-linear components h_3 and h_4 because they are used to smooth phrase-based models and to generate word alignment matrices. HMM-based alignment models were chosen here because, according to Och and Ney (2003) and Toutanova, Ilhan, and Manning (2002), they outperform IBM 1 to IBM 4 alignment

models while still allowing the exact calculation of the likelihood. However, our proposal is not restricted to the use of HMM-based alignment models.

The standard estimation procedure for HMM-based alignment models is carried out by means of the EM algorithm. However, the standard EM algorithm is not appropriate to incrementally extend our HMM-based alignment models because it is designed to work in batch training scenarios. To solve this problem, the incremental view of the EM algorithm (Neal and Hinton 1998) can be applied.

Model Definition. HMM-based alignment models are a class of single-word alignment models. Single-word alignment models are based on the concept of alignment between word positions of the source and the target sentences f_1^J and e_1^I. Specifically, the alignment is defined as a function $a : \{1 \cdots J\} \rightarrow \{0 \cdots I\}$, where $a_j = i$ if the jth source position is aligned with the ith target position. Additionally, $a_j = 0$ notes that the word position j of f_1^J has not been aligned with any word position e_1^I (or that it has been aligned with the **null word** e_0). Letting $\mathcal{A}(f_1^J, e_1^I)$ be the set of all possible alignments between e_1^I and f_1^J, we formulate $Pr(f_1^J|e_1^I)$ in terms of the alignment variable as follows:

$$Pr(f_1^J|e_1^I) = \sum_{a_1^J \in \mathcal{A}(f_1^J, e_1^I)} Pr(f_1^J, a_1^J|e_1^I) \tag{16}$$

Under a generative point of view, $Pr(f_1^J, a_1^J|e_1^I)$ can be decomposed without loss of generality as follows:

$$Pr(f_1^J, a_1^J|e_1^I) = Pr(J|e_1^I) \cdot \prod_{j=1}^{J} Pr(f_j|f_1^{j-1}, a_1^j, e_1^I) \cdot Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) \tag{17}$$

HMM-based alignment models are very similar to IBM models (Brown et al. 1993), specifically, they only differ in the assumptions made over the alignment probabilities. HMM-based alignment models use a first-order alignment model $p(a_j|a_{j-1}, I)$ to approximate the distribution $Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$ and a word-to-word lexical model $p(f_j|e_{a_j})$ to approximate the distribution $Pr(f_j|f_1^{j-1}, a_1^j, e_1^I)$, resulting in the expression

$$p(f_1^J, a_1^J|e_1^I, \Theta) = \prod_{j=1}^{J} p(f_j|e_{a_j}) \cdot p(a_j|a_{j-1}, I) \tag{18}$$

where we assume that a_0 is equal to zero and

$$\Theta = \begin{cases} p(f|e) & \forall f \in \mathcal{F} \text{ and } e \in \mathcal{E} \\ p(i|i', I) & 1 \leq i \leq I,\ 0 \leq i' \leq I \text{ and } \forall\, I \end{cases} \tag{19}$$

is the set of hidden parameters.

Incremental EM Algorithm. The incremental EM algorithm was introduced by Neal and Hinton (1998) in batch learning settings. In such settings, the set of training samples is known before the training process takes place (see Section 3.1). Here we will

instantiate the incremental EM algorithm for a batch learning translation task with a given set of training pairs, $\{(\mathbf{f}_1, \mathbf{e}_1), \ldots, (\mathbf{f}_m, \mathbf{e}_m), \ldots, (\mathbf{f}_M, \mathbf{e}_M)\}$. After that, we will present the application of the algorithm in an online learning setting.

It can be demonstrated that $s(\mathbf{f}, \mathbf{e}, \mathbf{a}) = \sum_m s_m(\mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)$ constitutes a vector of sufficient statistics for the model parameters, where $s_m(\mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)$ is the vector of sufficient statistics for data item m:

$$s_m(\mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m) = \begin{cases} c(f|e; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m) & \forall f \in \mathcal{F} \text{ and } e \in \mathcal{E} \\ c(i|i', I; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m) & 1 \leq i \leq I,\ 0 \leq i' \leq I \text{ and } \forall\, I \end{cases} \tag{20}$$

with $c(f|e; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)$ being the number of times that the word e is aligned to the word f for the sentence pair $(\mathbf{f}_m, \mathbf{e}_m)$; and $c(i|i', I; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)$ being the number of times that the alignment i has been seen after the previous alignment i', given a source sentence composed of I words for the sentence pair $(\mathbf{f}_m, \mathbf{e}_m)$.

To implement the E step of the incremental EM algorithm, we need to obtain the expected value at trial t of the sufficient statistics, given the probability distribution of the hidden alignment variable: $\tilde{s}_m^{(t)}$, where counts are replaced by expected counts, $c(f|e; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}$ and $c(i|i', I; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}$. If data item m is chosen to update the model at trial t, then the E step requires the following operations:

$$\left.\begin{array}{l} \text{Set } \tilde{s}_i^{(t)} = s_i^{(t-1)} \text{ for } i \neq m \\ \text{Set } \tilde{s}_m^{(t)} = \{c(f|e; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}, c(i|i', I; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}\} \\ \text{Set } \tilde{s}^{(t)} = \tilde{s}^{(t-1)} - \tilde{s}_m^{(t-1)} + \tilde{s}_m^{(t)} \end{array}\right\} \tag{21}$$

Regarding the M step, we have to obtain the set of parameters that maximizes the likelihood of the complete data given the expected values of the sufficient statistics, obtaining the following update equations:

$$p(f|e)^{(t)} = \frac{\sum_{m=1}^{M} c(f|e; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}}{\sum_{f' \in \mathcal{F}} \sum_{m=1}^{M} c(f'|e; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}} \tag{22}$$

$$p(i|i', I)^{(t)} = \frac{\sum_{m=1}^{M} c(i|i', I; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}}{\sum_{i''=1}^{I} \sum_{m=1}^{M} c(i''|i', I; \mathbf{f}_m, \mathbf{e}_m, \mathbf{a}_m)^{(t)}} \tag{23}$$

In the previous equations, the numerator values constitute the cumulative sufficient statistics $\tilde{s}^{(t)} = \sum_m \tilde{s}_m^{(t)}$ for the model parameters.

Application to an Online Setting. The previous instantiation of the incremental EM algorithm works in a batch learning setting where the set of training samples is given a priori. By contrast, in the online learning paradigm the training samples are not available a priori but become available over time—specifically, one at a time. Given

the training sample (f_1^J, e_1^I) at trial t, the incremental update equation for the cumulative sufficient statistics, $\tilde{s}^{(t)}$, is given by the following expression:

$$\tilde{s}^{(t)} = \tilde{s}^{(t-1)} + \tilde{s}_t^{(t)} \tag{24}$$

where $\tilde{s}_t^{(t)}$ represents the sufficient statistics for sample at trial t.

It should be noted that now we no longer need to subtract the previous sufficient statistics for individual samples at trial $t-1$: $\tilde{s}_m^{(t-1)}$, as it is requested in the batch learning case (compare the previous equation with Equation (21)), since the training samples are discarded after being processed. This implies that only one training epoch is executed over the training samples, and also that the previous sufficient statistics for individual samples, $\tilde{s}_m^{(t-1)}$, should not be stored, allowing us to save memory in a substantial manner (the memory requirements may become prohibitive for large sample sets). The execution of only one training epoch can be seen as a limitation with respect to batch training, but empirical results in Section 4.4 show that the online update rule is competitive with the batch update rule due to the faster convergence of incremental EM. Additionally, this online update rule can be easily modified to execute multiple epochs while storing a reduced quantity of sufficient statistics for the last samples, as it is explained in Appendix A.

On the other hand, it is also worth mentioning that the sufficient statistics for a given sentence pair are nonzero for a small fraction of its components. As a result, the time required to update the parameters of the HMM-based alignment model depends only on the number of nonzero components.

After updating the sufficient statistics, $\tilde{s}^{(t)}$, we need to efficiently compute the model parameters. For this purpose, the normalizer factors for $\tilde{s}^{(t)}$ are also maintained.

The parameters of the direct HMM-based alignment model are estimated analogously to those of the inverse model.

3.5.5 Source Phrase Length, Target Phrase Length, and Distortion Models (h_5, h_6, and h_7). The δ parameters of the geometric distributions associated with the feature functions h_5, h_6, and h_7 are left fixed. Because of this, there are no sufficient statistics to store for these feature functions.

4. Experimental Results

This section describes the experiments that we carried out to test our proposed online learning techniques. Our experiments were focused on the PE and IMT scenarios, because they fit nicely into the online learning paradigm.

Our experiments use the log-linear SMT model with online learning capabilities described in Section 3.4. The IMT experiments reported here combine this log-linear SMT model with stochastic error correction models following the technique introduced in Ortiz-Martínez (2011). This technique uses word graphs to avoid retranslating the source sentence at each interaction of the IMT process. The incremental language and phrase-based models involved in the interactive translation process were generated and accessed by means of the open source Thot toolkit (Ortiz, García-Varea, and Casacuberta 2005; Ortiz-Martínez and Casacuberta 2014). The specific functionality used in this

Table 1
XRCE corpus statistics for three different language pairs.

		English	Spanish	English	French	English	German
Training	Sentences	55,761		52,844		49,376	
	Running words	571,960	657,172	542,762	573,170	506,877	440,682
	Vocabulary	25,627	29,565	24,958	27,399	24,899	37,338
Dev	Sentences	1,012		994		964	
	Running words	12,111	13,808	9,480	9,801	9,162	8,283
Test	Sentences	1,125		984		996	
	Running words	7,634	9,358	9,572	9,805	10,792	9,823
	MRR	19.8	24.4	25.9	26.9	26.9	22.8
	RRR	13.4	14.2	16.9	17.3	15.5	14.8
	UNF	18.0	14.9	18.8	16.5	13.7	24.6

experimentation has been made freely available in a new version of toolkit.[5] We did not consider the use of the Moses decoder (Koehn et al. 2007) in our experiments because it is not prepared to work in the IMT framework and it does not implement the incremental version of the EM algorithm (it implements the stepwise version, which is unstable in online learning settings, according to Blain, Schwenk, and Senellart [2012]). However, translation quality results reported in Ortiz-Martínez (2011) show that Thot is competitive with Moses for corpora of different complexities.

4.1 Corpora

The experiments were performed using the XRCE, the Europarl, and the EMEA corpora. The XRCE corpus (SchlumbergerSema S.A. et al. 2001) consists of translations of XRCE printer manuals from English to three different languages—namely, Spanish, French, and German. Table 1 shows the main figures of the XRCE corpora for training, development, and test partitions. The XRCE corpus is included here because it has been extensively used in the literature to report SMT and IMT results (a complete set of experiments with this corpus is shown in Barrachina et al. [2009]). This feature will allow us to compare the results of our proposed system with those obtained by state-of-the-art systems.

Table 1 also shows three different measures to predict the effectiveness of online learning for this particular translation task, including the modified repetition rate (MRR), the restricted repetition rate (RRR), and the unseen n-gram fraction (UNF) (see Section 3.3). In this work we propose to use only RRR and UNF measures to assess the usefulness of online learning. However, MRR will also be reported so as to give a better idea of the accuracy of these two measures. The three XRCE test sets present moderately high values for the three measures. Slight drops of the RRR measure with respect to the MRR measure are observed (that is, there are repeated n-grams in the test corpus that were already seen in the training set), suggesting that the repetition rate present in the test sets cannot be fully exploited by online learning.

The Europarl corpus (Koehn 2005) is extracted from the proceedings of the European Parliament, which are written in the different languages of the European Union.

5 It can be downloaded at `https://github.com/daormar/thot`.

Table 2
Europarl corpus statistics for three different language pairs.

		English	Spanish	English	French	English	German
Training	Sentences	1,547,596		1,525,315		1,601,936	
	Running words	33,125 K	34,116 K	31,923 K	34,571 K	36,185 K	34,070 K
	Vocabulary	96,741	146,288	95,232	114,417	101,113	318,475
Dev	Sentences	3,003		3,003		3,003	
	Running words	72,988	78,888	72,988	81,800	72,988	72,603
Test	Sentences	3,000		3,000		3,000	
	Running words	64,809	70,562	64,809	73,664	64,809	63,411
	MRR	10.7	11.4	10.7	12.5	10.7	8.0
	RRR	4.3	4.4	4.3	4.8	4.2	3.1
	UNF	20.7	19.6	20.8	18.2	20.0	26.0

In our experiments we used the version created for the shared task of the ACL 2013 Workshop on Statistical Machine Translation (Bojar et al. 2013). To simplify the experiments, all those sentences whose length in words was greater than 40 were removed from the training set. Regarding the language pairs under consideration, again, we will translate from the English language to Spanish, French, and German. Table 2 shows the main figures of training, development, and test sets. The Europarl corpus constitutes one good example of a complex, real-world translation task that is also very well known in the MT scientific community. Regarding the measures to predict the effectiveness of online learning, it should be noted that the MRR measure is much lower than that observed for the XRCE corpora (see Table 1). Moreover, a significant drop in the RRR measure with respect to the MRR is observed, indicating that the vast majority of the repeated n-grams in the test corpus has already been seen in the training corpus. Therefore, we expect a limited effectiveness of online learning for this task.

Finally, we also carried out experiments with the EMEA corpus. The EMEA corpus consists of documents from the European Medicines Agency, made available with the OPUS corpora collection (Tiedemann 2009). In this work we extracted specific test sets of 3,000 sentences from the whole set of parallel sentences. Before doing this, we first removed the duplicate sentence pairs contained in this corpus (they represent a very high percentage of the total number of sentence pairs). Table 3 contains some statistics of the resulting corpora. The main interest of the EMEA corpus in our proposed experimentation is that it constitutes an example of an in-domain translation task. The models of the SMT system can be estimated from the out-of-domain Europarl corpus and then used to translate the EMEA corpus, simulating a non-stationary translation task. As it can be seen in Table 3, MRR, RRR, and UNF measures clearly suggest the potential usefulness of online learning in this task (RRR and UNF were calculated using the Europarl training corpus as the out-of-domain corpus).

4.2 Assessment Criteria

We evaluated our SMT system with online learning using three evaluation measures: WER, BLEU, and KSMR. System performance was assessed by comparing the system

Table 3
Statistics for a subset of the EMEA corpus selected for testing purposes. Figures are shown for three different language pairs. RRR and UNF measures have been calculated using the Europarl training set as the out-of-domain corpus.

		English	Spanish	English	French	English	German
	Sentences	3,000		3,000		3,000	
Test	Running words	46,311	50,410	45,260	53,787	46,319	43,887
	Vocabulary	4,716	5,329	4,659	5,135	4,772	5,809
	MRR	34.2	33.5	34.5	34.7	34.6	29.1
	RRR	30.3	29.3	30.8	31.1	30.6	25.3
	UNF	43.5	39.9	44.0	45.5	43.7	46.5

translations with the corresponding target language references of the test set. WER and BLEU measures are intended for its use in the evaluation of the PE scenario:

- **Word Error Rate** (WER): the system hypothesis is compared to the reference translation by computing the minimum number of edit distance operations (substitutions, insertions and deletions) between the hypothesis and the reference translation, divided by the number of reference words.
- **Bilingual evaluation understudy** (BLEU): The BLEU score (Papineni et al. 2001) computes the geometric mean of the precision of n-grams of various lengths between a hypothesis and a set of reference translations multiplied by a factor that penalizes short sentences.

Because we want to evaluate the performance of our proposed SMT system in an IMT scenario, we need to estimate the effort required by the user to produce correct translations using the system. To this end, we use the target references to simulate the translations that the user has in mind. The first translation hypothesis for each given source sentence is compared with a single reference translation and the longest common character prefix (LCCP) is obtained. The first non-matching character is replaced by the corresponding reference character and then a new system translation is produced. This process is iterated until a full match with the reference is obtained. Each computation of the LCCP would correspond to the user looking for the next error and moving the pointer to the corresponding position of the translation hypothesis. We refer to a pointer movement as a **mouse-action**. On the other hand, each character replacement would correspond to a **keystroke** of the user. If the first non-matching character is the first character of the new system hypothesis in a given interaction, no LCCP computation is needed; that is, no pointer movement would be made by the user. Bearing this in mind, we define the following IMT evaluation measure:

- **Keystroke and mouse-action ratio** (KSMR): KSMR (Barrachina et al. 2009) is the number of keystrokes plus the number of mouse-actions divided by the total number of reference characters.

It is worthy of note that KSMR assumes that both keystrokes and mouse-actions require the same effort from the user. This constitutes an approximation, since these two actions are different and require different types of effort (Macklovitch 2006).

In addition to the WER, BLEU, and KSMR measures, in our experimentation we additionally report the learning time in seconds after each training sample presentation. The learning time is important to assess the ability of the learning algorithms to work in a real time scenario. All the experiments were executed on a Windows PC with a 2.00 Ghz Intel Xeon processor with 1GB of memory.

4.3 Experimentation Protocol

We evaluated our techniques by simulating real users. Because the different corpora used in the experiments contained source and target translations, we used the latter to simulate the reference translations that the user has in mind for each source sentence.

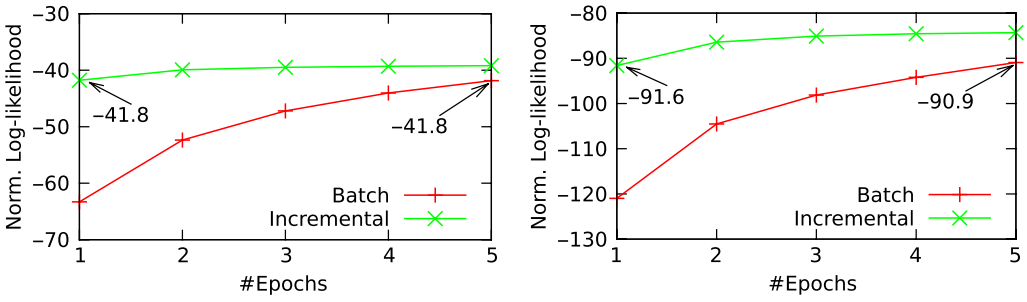
This paper studies the application of the online learning paradigm to SMT; therefore the experimentation follows the learning process structured as a sequence of trials that was described in Section 3.1. During this sequence of trials, online learning systems experience some sort of learning curve as they gain knowledge after each training sample presentation. Given that such a learning curve is an important issue when designing online learning algorithms, some of the results reported here include plots with the evolution of cumulative error measures.

It is also interesting to clarify the way in which the different corpora described in Section 4.1 have been used throughout the experimentation. One factor that has influenced the decisions in this regard is the high computational cost of batch retraining. Batch retraining is present in different experiments reported in this work because it provides a valuable reference when assessing the performance of online learning. In such experiments, we have defined specific subsets of the training corpora in order to speed up the experiments. More specifically, the first 10,000 sentences of the XRCE and Europarl corpora have been used.

The decisions regarding the use of the different corpora in the experiments can be summarized as follows. The training sets of the XRCE and Europarl corpora were used to measure the convergence properties of the incremental EM algorithm (Section 4.4). The above-mentioned subset of the training corpora was used to study the impact of the update frequency in the results (Section 4.5), to compare the performance of batch and online learning (Section 4.6), and to analyze the influence of sentence ordering in the system performance (Section 4.7). Finally, in the experiments to test the capability of our online learning techniques to learn from previously estimated models (Section 4.8), we used the training and development sets of the XRCE and Europarl corpora to initialize the system models, and the test sets to obtain translation results. For the system trained with the Europarl corpus, the experimentation is complemented with translation results using the in-domain EMEA corpus.

4.4 EM Algorithm Convergence Experiments

The standard estimation procedure for current phrase-based models relies on the generation of word alignment matrices. As it was explained in Section 3.5, in our proposal such alignment matrices are generated by means of HMM-based word alignment models that are incrementally updated from user feedback. For this purpose, we need to replace the batch EM algorithm by the incremental EM algorithm. Given the great importance of generating word alignments in the estimation of phrase-based models (see Section 3.5.3), we carried out experiments to compare the convergence rates of batch and incremental EM algorithms for HMM-based word alignment models.



(a) XRCE English–Spanish (b) Europarl English–Spanish

Figure 2
EM convergence experiment comparing the normalized log-likelihood obtained when executing five training epochs of the batch and incremental versions of the EM algorithm. The experiments were executed on the XRCE and Europarl English–Spanish training corpora.

Figure 2 shows the normalized log-likelihood that is obtained when executing up to five training epochs[6] of the batch and incremental versions of the EM algorithm (common training schemes in state-of-the-art SMT systems frequently execute five EM training epochs to train the different word-alignment models). Plots were obtained for the XRCE and Europarl training corpora and the three translation directions (from English to Spanish, French, and German). However, in the figure only the XRCE English to Spanish (Figure 2a) and the Europarl English to Spanish (Figure 2b) results are reported[7] (very similar results were obtained for the other language pairs).

According to the results presented in Figure 2, the incremental EM algorithm is able to obtain a greater normalized log-likelihood than that obtained by the batch EM algorithm for the two corpora under consideration. In addition to this, such a greater log-likelihood can be obtained with fewer EM training epochs. These observed results are due to the fact that the incremental EM algorithm executes complete E and M steps for each training sample, resulting in a much greater rate of model updates per each training epoch (Neal and Hinton 1998).

Note that, according to Equation (24), only one training epoch of the incremental EM algorithm is performed when training HMM-based alignment models (i.e., each training sample is processed only once by the learning algorithm and discarded afterwards). This contrasts with the conventional batch training scheme, in which a few training epochs (typically five) are executed. Hence, to fairly compare batch learning with our proposed online learning strategy, we should observe the relationship between the normalized log-likelihood of the incremental EM algorithm at the first training epoch and that of the fifth training epoch of the batch algorithm. According to the values shown in Figures 2a and 2b, we can appreciate a very small degradation in the log-likelihood (<1% for the Europarl corpus) or no degradation at all.

Because the difference in the log-likelihood between batch and incremental EM algorithms is negligible, we consider that the update rule for HMM-based alignment models given by Equation (24) is able to obtain word-alignment models comparable to those that can be obtained using batch learning. This claim will be supported with additional empirical evidence in Section 4.6.

6 An epoch is a single presentation of all samples in the training set.
7 To speed up the experiments, we took the first 100,000 sentences of the Europarl training corpus.

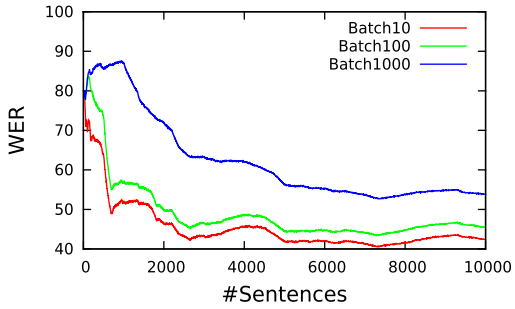
Nevertheless, it is possible to take advantage of the better convergence properties of the incremental EM algorithm by slightly modifying the conditions imposed by the online learning framework adopted in this paper (see Section 3.1). In such a framework, only the last sample presented so far to the learning algorithm can be used to modify the model parameters at each trial. This constraint can be slightly relaxed, allowing us to define alternative update rules for the HMM alignment models that execute more than one EM algorithm iteration over each sample. One example of such alternative update rules is described in Appendix A. Empirical results also given in the same appendix show that the obtained log-likelihood and the evaluation measures can be marginally improved with respect to the strict observation of the online learning framework.

Finally, it is worth pointing out that according to the results presented in Figure 2, incremental EM could be suitable to replace batch EM in a batch-learning scenario. However, one disadvantage of applying incremental EM to a batch-learning task is the necessity of storing the sufficient statistics for the whole data set: $s_1, s_2, \ldots, s_M$. For large data sets, the sufficient statistics may not fit in memory. Nevertheless, this information can be stored on disk and accessed efficiently, because the algorithm reads the data in a sequential manner. By contrast, this disadvantage is totally removed when incremental EM is applied in an online learning scenario, since the sufficient statistics for each training sample are discarded at the end of each trial, or after a finite number of trials for the alternative update rule described in Appendix A.

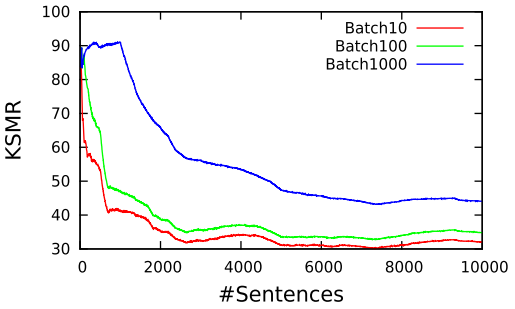
4.5 Impact of Update Frequency

One important aspect to be clarified when designing PE or IMT systems is the influence of the system update frequency on the obtained performance. It is expected that updating the system in a sentence-wise manner will produce the best results. However, this updating strategy poses efficiency problems because of the necessity of executing model updates in real time. This problem can be alleviated by defining an alternative update strategy in which the training process is delayed until a certain number of samples have been gathered. Delaying model updates may cause performance degradation, but it also constitutes one way to reduce the strong time requirements of a sentence-wise updating strategy. Specifically, if the time between updates is sufficiently high, the use of batch learning techniques could be appropriate (e.g., the training process can be executed overnight), removing the necessity of implementing online learning.

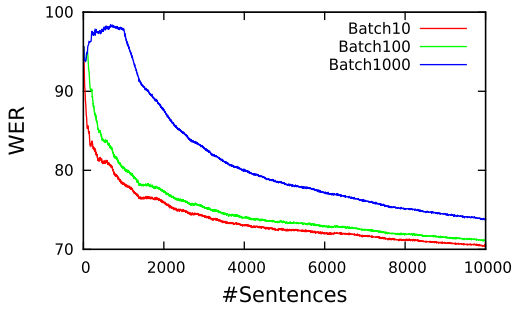
To test the impact of update frequency on system performance, we carried out PE and IMT experiments using the XRCE and the Europarl corpora in the three different language pairs (from English to Spanish, French, and German). In the experiments, the first 10,000 sentences of the different training sets were translated, using cumulative WER and KSMR to measure the user effort in the PE and IMT scenarios, respectively. The system was initialized with empty models, and after that such models were extended from the user validated translations using three different update frequencies: every 10, 100, and 1,000 sentences. Five training epochs were executed in all cases. We did not consider sentence-by-sentence updating because of the huge computational cost of the retraining. Model updates were performed by means of conventional batch-learning techniques, that is, the whole set of training samples seen so far is batch-retrained whenever the model is updated. Additionally, we adopted default values for the weights of the log-linear model. The results of the experiments are shown in Figure 3. Again, we only report results for the English–Spanish XRCE corpus (Figures 3a and 3b) and for the English–Spanish Europarl corpus (Figures 3c and 3d). Very similar results were obtained for the other language pairs.



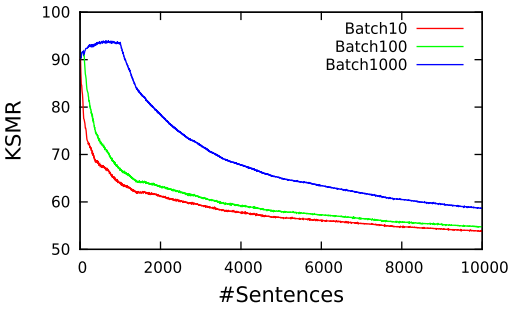
(a) Cumulative WER Results for XRCE English–Spanish



(b) Cumulative KSMR Results for XRCE English–Spanish



(c) Cumulative WER Results for Europarl English–Spanish



(d) Cumulative KSMR for Europarl English–Spanish

Figure 3
Impact of update frequency when translating the first 10,000 sentences of the English–Spanish language pair of the XRCE and Europarl corpora. A conventional SMT system executed five batch-training epochs every 10, 100, and 1,000 sentences. The system was initialized with empty models and default values for the weights of the log-linear model. Plots show the evolution of the user effort in PE and IMT scenarios measured in terms of cumulative WER and KSMR, respectively.

As it can be observed in Figure 3, the user effort in terms of WER and KSMR was lower when the update frequency was increased. More specifically, batch retraining every 10 samples (Batch10) consistently outperformed the rest of the systems in all cases and retraining every 100 samples (Batch100) was also consistently better than retraining every 1,000 sentences (Batch1000). Sharper curves were obtained when translating the XRCE corpora, probably reflecting that in this corpus, there are groups of sentences with highly different translation difficulties from the system point of view.

Note that, in all cases, the initial WER and KSMR measures are not equal to 100%. This is because of the fact that the system copies to the output all those unknown words contained in the input. In some cases such copied words (names, dates, etc.) are correct words contained in the reference translations.

Additionally, it is illustrative to consider the time cost of batch retraining for each system. Figure 4 shows the time cost in seconds of batch retrainings when we increase the number of training samples presented to the system. Results are shown for an update frequency equal to 1,000 when translating the XRCE (XRCE Batch1000) and the Europarl corpora (Europarl Batch1000). Higher update frequencies produced exactly the same results but with a higher number of points in the plots. As it was expected, the training times increase linearly with the number of training samples presented to the system. Time costs were higher for the more complex Europarl corpus. After processing

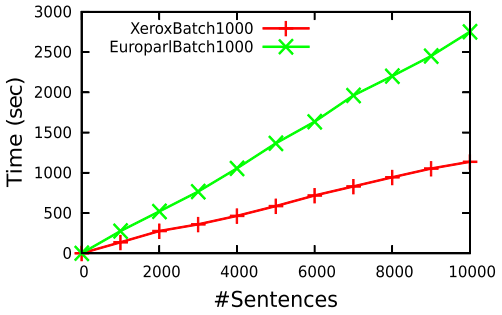


Figure 4
Batch retraining time in seconds as a function of the number of samples presented to the system. Results are shown for the first 10,000 sentences of the XRCE and Europarl training sets. Systems were retrained every 1,000 sentences executing five epochs. Time costs are given in seconds.

10,000 sentences, batch retraining took 19 minutes for the XRCE corpus and 45 minutes for the Europarl corpora. This gives a clear idea of the infeasibility of batch retraining in a sentence-wise updating strategy. Moreover, time costs of batch retraining soon become unaffordable because of their linear growth with the number of translated sentences.

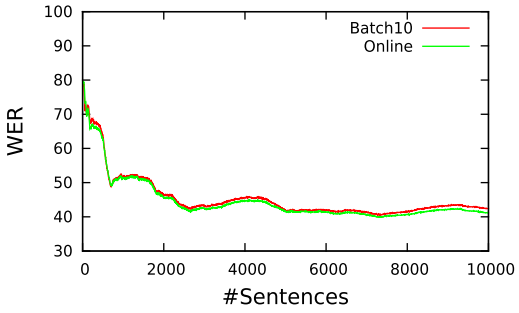
Turning back to the question made at the beginning of this section, experimental results clearly show that it is not possible to obtain the performance of a sentence-wise update strategy if the update frequency is decreased. This constitutes a strong argument in favor of the application of our proposed online learning techniques, which are specifically designed to learn from individual training samples. By contrast, batch learning requires the execution of expensive retraining processes whenever a new sample is presented to the learner. These findings are further supported in the next section, where the performance of batch and online learning systems are compared.

4.6 Batch versus Online Learning, Learning from Scratch

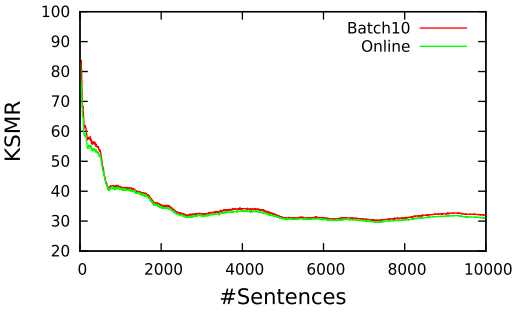
The great impact of frequent updates in the system performance demonstrated in the previous section poses the question of the necessity of replacing conventional batch learning techniques by online learning techniques. EM convergence experiments provided in Section 4.4 showed that the log-likelihood of HMM-based word alignment models using the incremental version of the EM algorithm is competitive with that obtained by using the conventional version. However, it is still unclear if the use of online learning will cause a degradation in the quality of the translations with respect to the use of batch learning.

Figure 5 shows the experiments we carried out to demonstrate the effectiveness of online learning. For this purpose, we compared the performance of a batch system executing five training epochs every 10 sentences (Batch10) with that of an online system (Online). Plots show the evolution of the user effort required to obtain correct translations. This effort is measured in terms of cumulative WER and KSMR for the PE and IMT scenarios, respectively. Initial models were empty in all cases. We report the results obtained when translating the first 10,000 sentences of the English–Spanish XRCE (Figures 5a and 5b) and Europarl training corpora (Figures 5c and 5d). Very similar results were obtained for English–French and English–German language pairs.

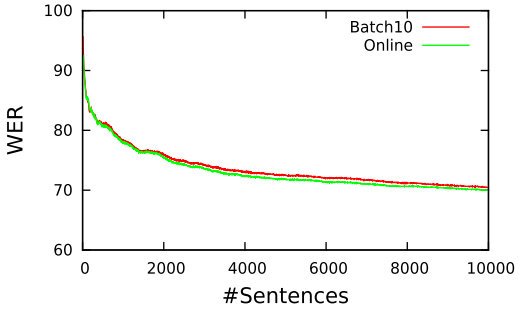
As it can be seen in Figure 5, the performance of online learning is slightly better than that of batch learning for both corpora and for the two scenarios under consideration: PE and IMT. We think that the reason for this slight improvement is the



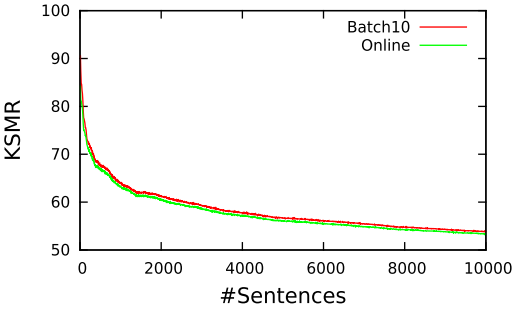
(a) Cumulative WER for XRCE English–Spanish



(b) Cumulative KSMR for XRCE English–Spanish



(c) Cumulative WER for Europarl English–Spanish



(d) Cumulative KSMR for Europarl English–Spanish

Figure 5
Comparison between batch and online learning when translating the first 10,000 sentences of the English–Spanish language pair of the XRCE and Europarl corpora. The batch learning system executed five training epochs every 10 sentences. The system was initialized with empty models and default values for the log-linear weights. Plots show the evolution of the user effort in the PE and IMT scenarios measured in terms of cumulative WER and KSMR, respectively.

higher update frequency of online learning, since the SMT models are extended for each individual training pair. It should be noted that the shape of the curves obtained with online learning is very similar to that of batch learning. This implies that incremental EM presents a stable behavior, which contrasts with the instability of the stepwise EM algorithm reported in Blain, Schwenk, and Senellart (2012). Finally, it is also worthy of note that the results also show that the system is able to learn from scratch.

It is also illustrative to carry out a descriptive analysis of the learning times per training sample that were obtained during this experiment. Figure 6 shows a boxplot summarizing the main statistics of the learning times for the XRCE and Europarl corpora. As it can be seen, the boxplot clearly show the small time cost of the learning process for the two different corpora under consideration. Specifically, the learning time was never greater than 1 second, and the median times were 0.03 and 0.16 seconds for the XRCE and the Europarl corpora, respectively. The learning time was greater for the Europarl corpus because of the greater length in words of the sentence pairs with respect to that of the XRCE corpus.

4.7 Ordering Effects

The order in which knowledge is acquired is an important issue in online learning tasks (see Section 3.1). When the label of a new sample is presented to the learning algorithm,

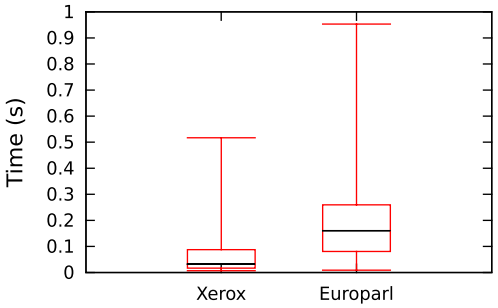


Figure 6
Boxplots of the time cost per each individual training sample required to train the first 10,000 sentences of the XRCE and Europarl training corpora using our proposed online learning techniques. Times are measured in seconds.

its parameters are modified to minimize cumulative prediction error. Hopefully, this modification will allow the system to provide more accurate predictions for similar samples. However, modifying parameters may also produce **lateral effects**. A lateral effect can cause the system to generate a wrong prediction for a given sample because of undesired changes in the learning algorithm parameters. One possible way to minimize the number of lateral effects is by processing similar samples in consecutive trials.

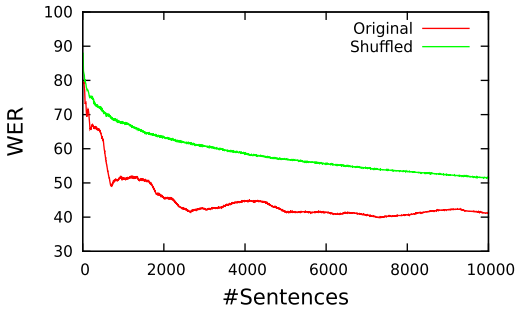
Figure 7 shows the experiments we executed to test the influence of corpus ordering in both WER and KSMR results when translating the first 10,000 sentences of the English–Spanish XRCE and Europarl training corpora by means of an online SMT system. For both tasks we translated the original portion of the training corpus and the same portion after being randomly shuffled.

As it can be seen in Figure 7, the obtained results were generally better for the original corpora than for the shuffled ones. The reason for the improved results is due to the fact that, in the original corpora, similar sentences appear more or less contiguously (because of the organization of the contents of the printer manuals for the XRCE corpus or to the chronological order of the parliamentary sessions for the Europarl corpus). This circumstance increases the accuracy of online learning, since with the original corpora the number of lateral effects occurred between the translation of similar sentences is decreased. By contrast, the accuracy was worse for shuffled corpora. Shuffling causes similar sentences to no longer appear contiguously and thus, the number of lateral effects that may occur between the translation of similar sentences is increased.

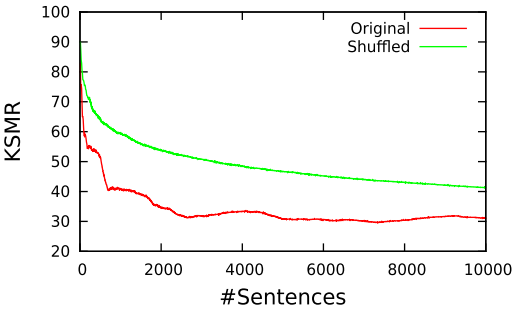
The differences between the results of original and shuffled corpora were much greater for the XRCE corpus than for the Europarl corpus. One possible explanation for this phenomenon is the lower repetition rate of the latter corpus. For low repetition rates, the number of lateral effects between the translation of similar sentences will be lower, since such sentences appear in a small number.

4.8 Learning from Previously Estimated Models

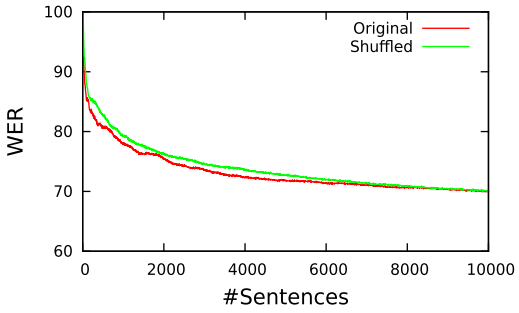
In the previous sections, we have shown empirical results where the models used by the SMT system were initially empty. In this section we show experiments in an alternative learning scenario where the SMT systems learn from previously estimated models. Under these circumstances, we compared the performance of a conventional SMT system with that of an online SMT system. More specifically, the conventional



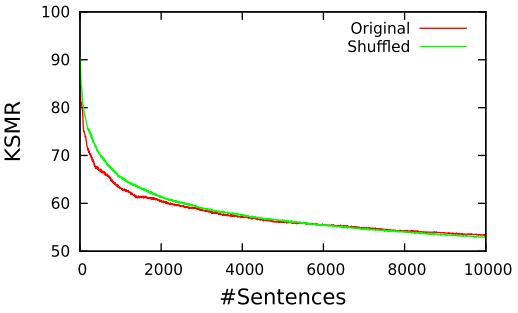
(a) Cumulative WER for XRCE English–Spanish



(b) Cumulative KSMR for XRCE English–Spanish



(c) Cumulative WER for Europarl English–Spanish



(d) Cumulative KSMR for Europarl English–Spanish

Figure 7
Impact of sentence ordering when translating the first 10,000 sentences of the English–Spanish language pair of the XRCE and Europarl corpora. Such sentences were presented in their original order or randomly shuffled. The system was initialized with empty models and default values for the weights of the log-linear model. The different plots show the evolution of the user effort in the PE and IMT scenarios measured in terms of cumulative WER and KSMR, respectively.

SMT system is a system that is not able to take advantage of user feedback after each translation, whereas the online SMT system uses the new sentence pairs provided by the user to revise the statistical models. Both systems used log-linear models trained in batch mode by means of the XRCE or the Europarl training corpora (five training epochs were executed). The weights of the log-linear model were adjusted for the corresponding development corpora via MERT.

4.8.1 XRCE Experiments. Table 4 shows the obtained results when translating the XRCE test corpora from English to Spanish, French, and German using conventional (batch learning without retraining) and online SMT systems. The table shows the BLEU, WER, and KSMR measures for both systems (95% confidence intervals are shown in all cases). The table also shows the average online learning time (LT) for each new sample presented to the system. All the improvements obtained with the online SMT system for the different measures were statistically significant. Greater improvements were obtained when translating to French and German. Such improvements could not be accurately predicted by means of the MRR measure (see Table 1), because, for instance, English to German presented a lower MRR value than English to Spanish. However, RRR and UNF measures were lower for English to Spanish, demonstrating the utility of such measures

Table 4
BLEU, WER, and KSMR results for the XRCE test corpora using conventional (batch learning without retraining) and online SMT systems. Both systems used MERT to adjust log-linear weights. The average online learning time (LT) in seconds is shown for the online system.

Corpus	SMT system	BLEU	WER	KSMR	LT (sec)
Eng–Spa	conventional	58.3±2.4	32.5±1.9	19.3±1.2	-
	online	64.0±2.4	28.1±1.7	16.6±1.1	0.06
Eng–Fre	conventional	32.2±2.2	63.7±2.2	36.7±1.2	-
	online	43.7±2.3	48.4±2.2	30.3±1.2	0.09
Eng–Ger	conventional	20.5±1.9	72.4±1.9	42.9±1.1	-
	online	28.9±2.1	61.7±2.1	37.0±1.3	0.07

to obtain refined predictions of the impact of online learning in the results. The average learning times allow the system to be used in a real-time scenario.

Additionally, in Table 5 we show a comparison of the KSMR results obtained by our proposed online SMT system, with those obtained by different state-of-the-art IMT systems described in the literature. These IMT systems are based on different translation approaches, including the alignment templates (AT), the stochastic finite-state transducer (SFST), and the phrase-based (PB) approaches to IMT (see Barrachina et al. [2009] for more details). AT and SFST systems follow the word graph-based approach to generate the IMT suffixes, whereas the PB system retranslates the source sentence at each interaction of the IMT process. Experiments reported in Barrachina et al. (2009) showed that word graph–based systems are much faster than systems that retranslate the source sentence at each interaction, but obtain slightly worse results. Because quick response times are critical in an IMT scenario, the majority of the IMT systems reported in the literature, as well as the one proposed here, follow a word graph–based implementation strategy. Our system significantly outperformed the results obtained by the state-of-the-art systems, except those of the PB system for English to Spanish. Even in this case, our system obtained slightly better results.

4.8.2 Europarl Experiments. Table 6 shows the translation results from English to Spanish, French, and German for the Europarl corpus when using conventional and online SMT systems. Again, BLEU, WER, and KSMR measures for conventional and online SMT

Table 5
KSMR results of the comparison of our system with online learning and three different state-of-the-art IMT conventional systems. The experiments were executed on the XRCE corpora. Best results are shown in bold.

Corpus	AT	PB	SFST	Online
Eng–Spa	23.2±1.3	16.7±1.2	21.8±1.4	**16.6±1.1**
Eng–Fre	40.4±1.4	35.8±1.3	43.8±1.6	**30.3±1.2**
Eng–Ger	44.7±1.2	40.1±1.2	45.7±1.4	**37.0±1.2**

Table 6
BLEU, WER, and KSMR results for the Europarl test corpora using conventional (batch learning without retraining) and online SMT systems. Log-linear weights were adjusted via MERT. The average online learning time (LT) in seconds is shown for the online system.

Corpus	SMT system	BLEU	WER	KSMR	LT (sec)
Eng-Spa	conventional	21.0±0.5	65.4±0.7	45.9±0.4	-
	online	22.5±0.6	63.4±0.7	44.7±0.4	0.2
Eng-Fre	conventional	21.2±0.6	64.4±0.7	44.4±0.5	-
	online	22.6±0.6	63.2±0.7	43.2±0.5	0.2
Eng-Ger	conventional	13.1±0.5	73.8±0.7	49.2±0.4	-
	online	14.1±0.5	72.8±0.7	48.0±0.4	0.2

systems are shown. The table also reports the average LT for the system with online learning.

As seen in Table 6, online learning allowed us to obtain around one point of improvement in the three measures under consideration with respect to the conventional system (without retraining). However, the improvements were not statistically significant in some cases (WER for English to French, BLEU and WER for English to German). These smaller improvements with respect to those observed for the XRCE task could be predicted from the lower repetition rates that the Europarl corpus present (see Table 2), especially for the RRR measure, which reflects how frequently unseen *n*-grams are repeated in the corpus to be translated.

The average online learning time was greater than that for the XRCE corpus shown in Table 4. Despite this, it was small enough for its use in a real-time scenario.

4.8.3 EMEA Experiments. Table 7 shows the obtained results measured in terms of BLEU, WER, and KSMR when translating the EMEA test corpora from English to Spanish, French, and German. Conventional and online SMT systems with models estimated from the Europarl training corpora were used. In this experimentation, we also considered a third SMT system that used online learning from scratch. The average online LT for each new sample presented to the system is also reported.

As seen in Table 7, the online SMT system, whether it learned from scratch or from previously estimated models, significantly outperformed the results obtained by the conventional SMT system (batch learning without retraining) for the three evaluation measures. The magnitude of the improvements were greater than that observed for the XRCE and Europarl corpora, as could be predicted from the MRR, RRR, and UNF measures provided in Table 3. Note that, although the values of the MRR measure of EMEA were similar to that obtained for the XRCE corpus, the improvements were greater because of the higher number of unseen events (explained by the RRR) and their more-frequent presence in the EMEA corpora (explained by means of the UNF). The online learning system using previously estimated models consistently produced improvements of more than 10 points for the three evaluation measures and for each of the language pairs. The improvements were smaller for the online learning system from scratch. Despite this, it is worth noting that for this task, it would be better to use a system with online learning, even if its models were initially empty, than to use a conventional SMT system with models trained on out-of-domain corpus.

Table 7
BLEU, WER, and KSMR for the EMEA test corpora using conventional (batch learning without retraining), online, and online from scratch SMT systems, with models estimated by means of the Europarl training corpora. Log-linear weights were trained via MERT. The average online learning time (LT) in seconds is shown for the online system.

Corpus	SMT system	BLEU	WER	KSMR	LT (sec)
Eng-Spa	conventional	24.0±0.8	64.0±0.9	49.1±0.6	-
	online	45.6±1.2	49.6±1.4	32.3±0.8	0.1
	online scratch	36.0±1.2	52.2±1.0	35.6±0.8	0.1
Eng-Fre	conventional	18.3±0.7	69.4±0.9	52.3±0.6	-
	online	39.7±1.3	53.9±1.3	33.7±0.8	0.1
	online scratch	30.0±0.2	59.5±1.1	41.2±0.8	0.1
Eng-Ger	conventional	17.7±0.8	77.8±1.3	55.3±0.6	-
	online	35.7±1.3	61.8±1.5	37.2±0.8	0.1
	online scratch	27.4±1.2	65.5±1.2	43.8±0.9	0.1

Regarding the average learning times per each new training pair, again, they were small enough to allow the use of online learning in real-time scenarios.

5. Discussion

The set of experiments presented in the previous section validates the use of incremental EM to design online learning algorithms for SMT. One common criticism of incremental EM is its great memory requirements due to the necessity of storing the set of sufficient statistics for each individual sample. However, this criticism was initially made in the context of batch learning (see Liang and Klein 2009), and there were no studies on its application to online learning tasks, with the exception of the work presented in Ortiz-Martínez, García-Varea, and Casacuberta (2010). Here we have proposed two update rules that present constant (and very small) memory requirements while maintaining the same or even better performance than batch retraining. The first proposed update rule (see Equation (24)) allows us to execute one epoch over the training samples by storing the sufficient statistics for the last one. This significantly differs from the memory requirements of incremental EM applied in a batch learning context, where the sufficient statistics for all of the samples seen so far need to be stored (for a more detailed explanation, see Equation (24) and the subsequent discussion). The second update rule (see Appendix A) is a generalization of the first one, executing several epochs over the training samples by storing the sufficient statistics of only a fixed quantity of the last samples.

Furthermore, incremental EM presents convergence rates very similar to that of batch retraining, resulting in a very stable learning algorithm whose behavior contrasts with the stability problems reported in other works for stepwise EM (Blain, Schwenk, and Senellart 2012).

The reported experiments also measured the impact of frequent updates in the PE and IMT scenarios. In both cases, system performance was greatly improved when the update frequency was increased. This constitutes a strong argument in favor of using online learning, because of the prohibitive time cost of frequent batch retrainings.

The experimentation also showed a clear manifestation of one inherent feature of online learning: the ordering effects of training samples in system performance. We compared the results that are obtained when the sentences to be translated were chronologically ordered with those obtained with a randomly ordered corpus. The benefits of online learning are favored by the former situation, since similar sentences appear more or less contiguously. This would be an example of the document internal repetition phenomenon mentioned by some authors. Moreover, this is the expected situation in real translation scenarios.

The online learning techniques proposed in this article allows us to achieve the goal of learning from scratch in an efficient manner. Learning from scratch is not only a theoretical scenario that can be proposed in an SMT research context, but a technique with potential utility in real domain adaptation tasks. It is generally acknowledged that in-domain corpora are difficult to obtain, and, as a result, SMT system models are initialized by means of out-of-domain texts (Irvine et al. 2013). Empirical results presented here show that online learning from scratch can produce significantly better results than a conventional SMT system with models estimated from out-of-domain corpora. This would not be the only possible application of online learning in this scenario. Indeed, online learning from scratch has already been applied to build automatic post-editing systems designed to work in situations where in-domain corpora are not available (Lagarda et al. 2015). Another possibility would be to linearly combine a model trained from out-of-domain data with an initially empty and separate online learning model. This online learning model could be useful both for learning new translations as well as for giving preference to the in-domain data. For example, Mirking et al. (2013) demonstrate the utility of a similar system, but implemented with batch retraining.

Finally, the presented results also demonstrate that our proposed implementation of online learning is able to learn from previously existing models. We compared the obtained BLEU, WER, and KSMR measures of a conventional SMT system (batch learning without retraining) with that of an online system. The improvements were significant in almost all cases, and very strong for specific tasks and language pairs (their magnitude was greater than 10 points for the different measures under consideration).

6. Related Work

Online learning has been a main topic of research in the field of machine learning. However, in the SMT framework, the vast majority of the work has been devoted to the study of the batch-learning setting. The application of online learning to SMT has been mostly centered on estimating the feature weights of a log-linear model by means of discriminative training techniques. Examples of this kind of work can be found in Och and Ney (2002), Liang et al. (2006), Watanabe et al. (2007), Chiang, Marton, and Resnik (2008) and Martínez-Gómez, Sanchis-Trilles, and Casacuberta (2012). These works differ from the one presented here in that we apply online learning techniques to train the features of the log-linear model instead of their weights.

To our knowledge, the first work on online learning for SMT focused on training model features is Cesa-Bianchi, Reverberi, and Szedmak (2008). That paper presents a very constrained version of online learning applied to a CAT scenario, where the translation model cannot be extended because of the high computational cost of retraining the whole model for a new training pair. The literature on online learning for SMT has tried to solve or alleviate this problem in different ways, as it is discussed in the following sections, where we identify four different approaches.

6.1 Online Learning Constrained by Previously Existing Models

The first approach accounts for work that relies on previously estimated models as a measure to avoid full model retraining. An early attempt can be found in Nepveu et al. (2004), where dynamic adaptation of an IMT system via cache-based model extensions to language and translation models is proposed. That work constitutes a domain adaptation technique and not an online learning technique, since the proposed cache components require pre-existent models estimated in batch mode. As pointed out by the authors, one of the most important limitations of their proposal is the inability to process words that were not seen during the estimation of the pre-existent models. In addition to this, their IMT system does not use state-of-the-art models. The work presented in Hardt and Elming (2010) applies a similar strategy to a modern phrase-based SMT system, using heuristic IBM4-based word alignment techniques to add new phrase pairs to a local phrase table. Their technique shares similar limitations with the work presented in Nepveu et al. (2004), since it requires pre-existent models estimated in batch mode. In addition to this, according to the empirical results that are reported, their proposal is slow (average learning times per sentence of up to 1 minute) and unable to obtain the same results as conventional batch retraining because of the heuristic decisions that are made to incrementally train the phrase models. Bertoldi, Cettolo, and Federico (2013) present an online learning technique based on cache components, which is also strongly inspired by the work by Nepveu et al. (2004). In spite of the fact that their proposed technique is able to extend the translation model by obtaining alignments at the phrase level, the phrase model that is used to generate such alignments is not updated from user feedback. Again, this makes the system dependent on previously existent models estimated in batch mode and presumably less reliable when learning from new sentence pairs that contain poorly represented or unseen events during the initial training stage. Finally, Wäeschle et al. (2013) present a work very similar to Bertoldi, Cettolo, and Federico (2013) that also includes discriminative training methods.

Mathur, Cettolo, and Federico (2013) present an alternative approach in which a new log-linear component for the set of phrase pairs contained in the translation table is used. This component is updated so as to increase or decrease the score of specific phrase pairs depending on whether they are present or not in the user validated translations. The main difference between that work and other techniques mentioned earlier is that it does not tackle the problem of adding new phrase pairs to the translation model. Instead, it is deliberately restricted to adapt previously existing parameters. Another alternative approach is presented in Denkowski, Dyer, and Lavie (2014), where a translation grammar, a language model, and a set of log-linear weights are adapted in a post-editing task. In that case, pre-existent word alignment models estimated in batch mode are required to extend the translation grammar.

6.2 Online Learning Based on Output to Reference Alignments

Other works try to avoid retraining the phrase model by aligning the output of the decoder with the reference given by the user (Blain, Schwenk, and Senellart 2012; Simard and Foster 2013). Specifically, Blain, Schwenk, and Senellart (2012) propose obtaining word alignments between the source and reference sentences using the system output as pivot. Such alignments are obtained by combining the word alignments from source to system output and from system output to reference. Because the output and the

reference sentences are (hopefully) very similar, they can be aligned using edit-distance–based algorithms, which do not require being trained. Regarding the work of Simard and Foster (2013), it conceives the translation as a two stage process, first the source is translated by a regular SMT system, and after that, the output is translated again using an automatic post-editing module. This module is implemented as a phrase translation system trained from the system translations and their references. To extract the phrase pairs, word level alignments based on edit distance are used.

The main drawback of the approaches based on alignments between the output and the reference sentences is the strong assumption regarding the similarity of such sentences. The similarity may be low when the source sentence contains poorly represented or unseen events during the initial training process, a situation that is very common in real translation tasks. A low similarity could negatively affect the resulting word alignments because of the great simplicity of edit-distance algorithms. As a consequence, the obtained performance will be worse than that obtained by means of retraining, as is reported in Blain, Schwenk, and Senellart (2012). The authors of that paper claim that the only alternative to retraining is the application of techniques based on the so-called stepwise EM algorithm (Cappé and Moulines 2009), which is unstable when used in an online learning setting and has a lower performance. By contrast, in this paper we empirically demonstrate that the incremental version of the EM algorithm (Neal and Hinton 1998) can also be applied and does not have such disadvantages.

6.3 Quick Adaptation Based on Retraining

Mirking and Cancedda (2013) propose techniques to achieve quick model updates using batch retraining. Such techniques are based on maintaining different translation tables for out- and in-domain data. Separated in-domain models allow the user to quickly update the models via batch retraining and to give preference to the in-domain table via tuning of log-linear weights. Unfortunately, such quick model updates cannot be executed in a sentence-wise manner but in longer lapses of time (e.g., a day). Moreover, the proposed configurations have the disadvantage of becoming slower over time. Mirking and Cancedda state the great interest of replacing batch retraining by principled incremental training, but they also point out that such a technology is not still mature or available.

6.4 Pure Online Learning Techniques

Finally, there is another category of techniques that tackle the training problem following a pure online learning approach, removing the constraints on model updates imposed by the techniques mentioned previously. Levenberg, Callison-Burch, and Osborne (2010) introduced stream-based adaptation for SMT. This technique is able to incrementally learn from scratch or from previously estimated models. However, their approach only captures a restricted notion of online learning, because it is designed to process large amounts of incoming data. Indeed, their training regime uses the stepwise EM algorithm, which works by processing large blocks of training data and not in a sentence-wise manner.

On the other hand, the work presented in Ortiz-Martínez, García-Varea, and Casacuberta (2010) introduces a state-of-the-art log-linear SMT model using a set of incremental update rules for the feature functions. The resulting system is able to learn from scratch or from previously estimated models, and it is applied in an IMT scenario.

One key element of the proposal is the use of the incremental EM algorithm. As far as we know, such work constitutes the first proposal that successfully applies online learning to SMT, solving the technical limitations encountered in previous works. However, some important aspects of the proposal were not clarified, such as its performance in a PE scenario or its effectiveness when compared with batch retraining.

Here, we extend the work by Ortiz-Martínez, García-Varea, and Casacuberta (2010) both theoretically and empirically. Regarding the theoretical aspects of online learning in SMT, we provide a much more detailed explanation of the different online update rules. Additionally, we introduce an alternative update rule using the incremental EM algorithm, which solves one important limitation of the update rule that was originally introduced in Ortiz-Martínez, García-Varea, and Casacuberta (2010). Specifically, the new update rule allows us to execute more than one training epoch over the incoming data while maintaining constant time and spatial complexity (see Sections 3 and 4.4 as well as Appendix A for more details). In addition to this, we have integrated the log-linear model proposed in this paper into the IMT technique based on stochastic error correction models described in Ortiz-Martínez (2011). This IMT technique is able to generate the IMT suffixes in real time by calculating a word graph for the given source sentence at the initial interaction of the IMT process. By contrast, the technique implemented in Ortiz-Martínez, García-Varea, and Casacuberta (2010) is much slower, since the source sentence is retranslated at each interaction. Finally, we extend the work of Bertoldi, Cettolo, and Federico (2013) on measuring the effectiveness of online learning by proposing two new automatic measures.

The empirical study on online learning in SMT has been extended here in many ways. First, the proposed online learning techniques have been applied to an additional CAT scenario (specifically, the PE scenario). Additionally, the experimentation has been extended to larger corpora. Finally, we clarify some crucial aspects of online learning that were not studied previously, such as the convergence properties of the incremental EM algorithm, the impact of the frequency of model updates in the error measures, or the differences in performance between batch and online learning.

7. Conclusions and Future Work

We have proposed online learning techniques for SMT. Such techniques allow us to incrementally extend the statistical models involved in the translation process in an efficient manner. Our proposal breaks technical limitations encountered in other works, as is explained in Section 6.

To test our techniques, we carried out experiments in two different CAT scenarios, namely, PE and IMT, since they fit nicely into the online learning paradigm. However, it is important to stress that the applicability of the techniques proposed here is not restricted to such strict online learning scenarios. Another suitable scenario is the one described by Levenberg et al. (2010), where the incoming data are processed in large blocks instead of in a sentence-wise manner.

The experiments were executed in three different corpora, namely, the XRCE corpus, which has been extensively used in the literature to report CAT results, the well-known Europarl corpus, and the EMEA corpus, which is used in this work to simulate a non-stationary translation task. As summarized in Section 5, the results demonstrate the appropriateness of the incremental EM algorithm to implement online learning, the great impact of frequent model updates in translation quality, the importance of

ordering effects in online learning tasks, and the ability of our system to learn both from scratch or from previously estimated models.

We have also defined two new measures to predict the effectiveness of online learning in SMT tasks—the restricted repetition rate (RRR) and the unseen n-gram fraction (UNF). It has been empirically demonstrated that such measures allow us to refine the predictions that can be made with the modified repetition rate (MRR) measure, which is based on the repetition rate introduced in Bertoldi, Cettolo, and Federico (2013).

Additionally, it is important to stress that the SMT system with online learning capabilities used in this paper is implemented in the freely available Thot toolkit.

In this work we have focused on the incremental estimation of the parameters of the log-linear model features instead of their weights. We consider that weight adjustment is not a crucial aspect in our proposal because it typically affects a few model parameters, whereas estimation of feature parameters (including those of phrase and language models) may involve millions of parameters. In addition to this, weight adjustment can be performed offline, since typical MERT procedures use closed development corpora of a few thousand sentences (i.e., the training data is bounded and relatively small). In spite of that, we think that removing any offline training stages from practical online SMT system implementations could be useful to simplify their usage and design. Additionally, for future work we plan to incorporate bounds to the data structures used to store the model parameters, because incoming data is in principle unbounded.

Finally, we think that the online learning techniques proposed here can be exported to other natural language processing applications, where the system output is supervised by the user. In addition to this, our proposed techniques can also be useful to implement active learning algorithms for their use in online settings.

Appendix A: Alternative Update Rule for HMM-Based Alignment Models

In this appendix we introduce an alternative update rule for HMM-based alignment models. This rule allows us to execute more than one training epoch over the incoming data in contrast to the rule given by Equation (24). For this purpose, at each trial we no longer keep only the last training sample but a certain number of them.

If we want to execute E training epochs over the data, one possible way to implement the update rule is to keep the last E samples at a given trial, executing one incremental EM iteration for each sample. In spite of the fact that this update rule allows us to execute E epochs over the training data, it is still very different from the way in which conventional batch training works. Specifically, batch training only starts the next training epoch after having processed the whole training corpus in the previous epoch. This requisite cannot be met in an online learning setting, since the set of training samples is unbounded. Nevertheless, it is possible to obtain a training scheme that is more similar to that of batch training. For this purpose, we can store the last R samples instead of the last E, with $R \gg E$, processing a total of E samples at each trial. To increase the lapse of time (or the number of trials) until a given sample is reprocessed, the samples are processed in an *interlaced* way, as it is depicted in Figure A.1. Specifically, the figure shows which samples are processed by means of the incremental EM algorithm at $t = 10$ when we want to execute $E = 3$ epochs, keeping the last $R = 5$ samples. Under these circumstances, samples 6, 8, and 10 would be processed.

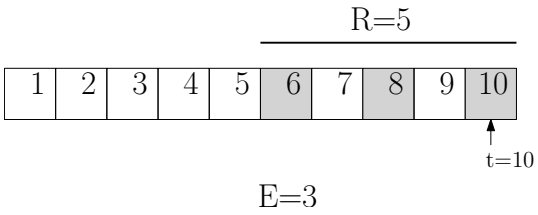


Figure A.1
Interlaced training scheme example. The squares represent the training samples that appear when the value of t is increased. Gray squares show the training samples that will be processed at trial $t = 10$ when $E = 3$ epochs are to be executed while storing the last $R = 5$ samples.

More formally, at a given trial t, if E epochs are to be executed and we choose to keep the last R samples, then the proposed update rule is as follows:

$$\tilde{s}^{(t)} = \tilde{s}^{(t-1)} - \sum_{e=1}^{E} \tilde{s}^{(t-1)}_{\lfloor t-(e-1)\cdot(R/E)\rfloor} + \sum_{e=1}^{E} \tilde{s}^{(t)}_{\lfloor t-(e-1)\cdot(R/E)\rfloor} \tag{A.1}$$

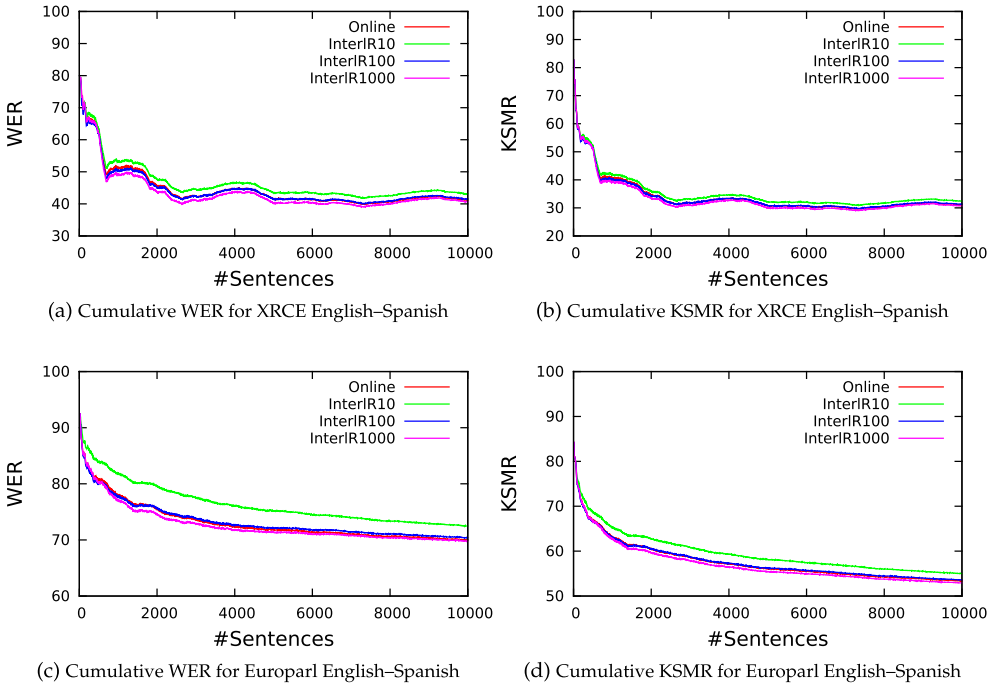
where $\lfloor\cdot\rfloor$ is the floor operator.

We will refer to this new rule as the **interlaced update rule**. Note that the basic update rule given by Equation (24) is a particular case of the interlaced rule where E and R are set to 1. In order to measure the performance of interlaced training, we briefly report here the results of some experiments that we carried out for the XRCE and the Europarl corpora. Because we obtained very similar results for the different language pairs involved in the experimentation, namely, from English to Spanish, French, and German, we only report here the English to Spanish results.

One important measure to be evaluated when assessing the performance of the interlaced update rule is the convergence rate of the EM algorithm. In this case, we cannot show plots of normalized log-likelihood per training epoch as we did in Section 4.4, because interlaced updates execute the different training epochs simultaneously for a given training set. Instead, we report the final normalized log-likelihood that is obtained

Table A.1
Normalized log-likelihood of the EM algorithm for a different number of training epochs and different implementations of the update rule for the translation model, including batch, incremental, and interlaced versions. Results executed on the XRCE and Europarl English–Spanish training corpora are shown.

		Norm. Log-likelihood	
	#Epochs	**XRCE**	**Europarl**
Batch EM	5	-41.8	-90.9
Incr. EM	1	-41.8	-91.6
Incr. EM (Interlaced, R=10)	5	-40.3	-87.1
Incr. EM (Interlaced, R=100)	5	-40.0	-85.6
Incr. EM (Interlaced, R=1000)	5	-39.6	-84.5



(a) Cumulative WER for XRCE English–Spanish

(b) Cumulative KSMR for XRCE English–Spanish

(c) Cumulative WER for Europarl English–Spanish

(d) Cumulative KSMR for Europarl English–Spanish

Figure A.2
Comparison between online and interlaced update rules when translating the first 10,000 sentences of the English–Spanish language pair of the XRCE and Europarl corpora. The interlaced system was executed for different values of the R parameter ($R = 10$, 100, or 1,000). The system was initialized with empty models and default values for the weights of the log-linear model. The different plots show the evolution of the user effort in the PE and IMT scenarios measured in terms of cumulative WER and KSMR, respectively.

at the end of the estimation process. Table A.1 shows the normalized log-likelihood of the EM algorithm with batch, incremental, and interlaced (with R equal to 10, 100, and $1,000$) updates, using the English–Spanish training set of the XRCE and Europarl corpora.[8] As we can see, interlaced training outperformed batch and incremental EM algorithms for all values of the R parameter. In addition to this, increasing the value of R produced improvements in the final normalized log-likelihood.

We also compared the performance in terms of WER and KSMR of the interlaced update rule with that of batch and online rules. Figure A.2 shows plots with the evolution of WER and KSMR for online and interlaced (with R equal to 10, 100, and $1,000$) update rules when translating the first $10,000$ sentences of the English–Spanish language pair of the XRCE and Europarl training corpora. As we can see, the value of the R parameter had a strong influence in the system performance. Specifically, the interlaced system with $R = 10$ clearly underperformed the results obtained by the online system; increasing the value to $R = 100$ obtained almost identical results; and $R = 1{,}000$ produced slightly better results. We think that this phenomenon is linked to the propensity of HMM-based alignment models for overfitting (see Och and Ney 2003 for more details). Under this point of view, the worse results for $R = 10$

8 To speed up the experiments, we took the first 100,000 sentences of the Europarl training corpus.

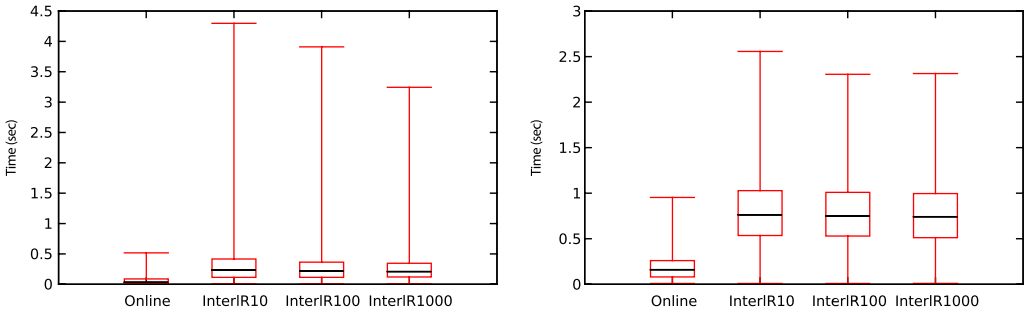


Figure A.3
Boxplots of the learning time per sentence required to train the first 10,000 training samples of the XRCE and Europarl corpora using online and interlaced systems (for R equal to 10, 100, and 1,000). Times are measured in seconds.

would be due to some sort of *local* overfitting, which is alleviated for greater values of R.

Finally, we also studied the impact of the interlaced training scheme in the learning time per sentence. Figure A.3 shows boxplots for the learning time per sentence that was required to train the first 10,000 training samples of the XRCE and Europarl English to Spanish corpora using online and interlaced systems for different values of the R parameter. The online system executed only one training epoch, whereas the interlaced systems executed five. All the times are reported in seconds. As we can see, interlaced updates increased the training time with respect to that of basic online updates (this was the expected outcome since five samples were processed at each trial instead of one). However, the time costs of interlaced updates were still affordable for both corpora (worst case times of a few seconds, median times less than 1 second for the different values of R).

Acknowledgments

The author wishes to thank Francisco Casacuberta and Ismael Garía Varea for their insightful comments on the article, always greatly appreciated. Work supported by the European Union 7th Framework Programme (FP7/2007-2013) under the CasMaCat project (grant agreement nº 287576) and by the Generalitat Valenciana under grant ALMAMATER (PROMETEOII/2014/030).

References

Alabau, V., C. Buck, M. Carl, F. Casacuberta, M. García-Martínez, U. Germann, J. González-Rubio, R. Hill, P. Koehn, L. A. Leiva, B. Mesa-Lao, D. Ortiz-Martínez, H. Saint-Amand, G. Sanchis-Trilles, and C. Tsoukala. 2014. Casmacat: A computer-assisted translation workbench. In *14th Annual Meeting of the European Association for Computational Linguistics: System Demonstrations*, pages 25–28, Gothenburg.

Anthony, M. and N. Biggs. 1992. *Computational Learning Theory: An Introduction*. Cambridge University Press, New York.

Barrachina, S., O. Bender, F. Casacuberta, J. Civera, E. Cubel, S. Khadivi, A. L. Lagarda, H. Ney, J. Tomás, E. Vidal, and J. M. Vilar. 2009. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.

Bertoldi, N., M. Cettolo, and M. Federico. 2013. Cache-based online adaptation for machine translation enhanced computer assisted translation. In *Proceedings of the XIV Machine Translation Summit*, pages 35–42, Nice.

Blain, F., H. Schwenk, and J. Senellart. 2012. Incremental adaptation using translation information and post-editing analysis. In *International Workshop on Spoken Language Translation*, pages 234–241, Hong-Kong.

Bojar, O., C. Buck, C. Callison-Burch, C. Federmann, B. Haddow, P. Koehn, C. Monz, M. Post, R. Soricut, and L. Specia. 2013. Findings of the 2013 Workshop on

Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia.

Brown, P. F., S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Cappé, O. and E. Moulines. 2009. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society Series. B* 71(1):593–613.

Cesa-Bianchi, N., G. Reverberi, and S. Szedmak. 2008. Online learning algorithms for computer-assisted translation. Deliverable D4.2, SMART: Stat. Multilingual Analysis for Retrieval and Translation. University of Southhampton.

Chen, S. F. and J. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 310–318, San Francisco.

Chiang, D., Y. Marton, and P. Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Stroudsburg, PA.

Church, K. W. and W. A. Gale. 1995. Poisson mixtures. *Natural Language Engineering*, 1:163–190.

Denkowski, M., C. Dyer, and A. Lavie. 2014. Learning from post-editing: Online model adaptation for statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 395–404, Gothenburg.

Federico, M., N. Bertoldi, M. Cettolo, M. Negri, M. Turchi, M. Trombetti, A. Cattelan, A. Farina, D. Lupinetti, A. Martines, A. Massidda, H. Schwenk, L. Barrault, F. Blain, P. Koehn, C. Buck, and U. Germann. 2014. The Matecat tool. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference System Demonstrations*, pages 129–132, Dublin.

Foster, G., P. Isabelle, and P. Plamondon. 1997. Target-text mediated interactive machine translation. *Machine Translation*, 12(1):175–194.

Giraud-Carrier, C. 2000. A note on the utility of incremental learning. *AI Communications*, 13(4):215–223.

González-Rubio, J., D. Ortiz-Martínez, and F. Casacuberta. 2012. Active learning for interactive machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 245–254, Avignon.

Green, S., J. Heer, and C. D. Manning. 2013. The efficacy of human post-editing for language translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 439–448, Paris.

Hardt, D. and J. Elming. 2010. Incremental re-training for post-editing SMT. In *Proceedings of the 9th Annual Conference of the Association for Machine Translation in the Americas*, Denver, CO. Available at `amta2010.amtaweb.org/`.

Irvine, A., J. Morgan, M. Carpuat, H. D. III, and D. S. Munteanu. 2013. Measuring machine translation errors in new domains. *Transactions of the Association for Computational Linguistics*, 1:429–440.

Knuth, D. E. 1981. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, MA, 2nd edition.

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

Koehn, P., 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. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague.

Koehn, P., F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, pages 48–54, Edmonton.

Koehn, P. 2009. A process study of computer-aided translation. *Machine Translation*, 23(4):241–263.

Lagarda, A. L., D. Ortiz-Martínez, V. Alabau, and F. Casacuberta. 2015. Translating without in-domain corpus: Machine translation post-editing with online learning techniques. *Computer Speech & Language*, 32(1):109–134.

Levenberg, A., C. Callison-Burch, and M. Osborne. 2010. Stream-based translation models for statistical machine translation. In *Proceedings of the North*

American Chapter of the Association for Computational Linguistics - Human Language Technologies, pages 394–402, Los Angeles, CA.

Liang, P., A. Bouchard-Côté, D. Klein, and B. Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Morristown, NJ.

Liang, P. and D. Klein. 2009. Online EM for unsupervised models. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 611–619, Morristown, NJ.

Macklovitch, E. 2006. Transtype2: The last word. In *Proceedings of LREC 2006*, Genoa, pages 167–172.

Martínez-Gómez, P., G. Sanchis-Trilles, and F. Casacuberta. 2012. Online adaptation strategies for statistical machine translation in post-editing scenarios. *Pattern Recognition*, 45(9):3193–3203.

Mathur, P., M. Cettolo, and M. Federico. 2013. Online learning approaches in computer assisted translation. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, pages 301–308, Sofia.

Mirkin, S. and N. Cancedda. 2013. Assessing quick update methods of statistical translation models. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 264–271, Heidelberg.

Neal, R. M. and G. E. Hinton. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Proceedings of the NATO-ASI on Learning in graphical models*, pages 355–368, Norwell, MA.

Nepveu, L., G. Lapalme, P. Langlais, and G. Foster. 2004. Adaptive language and translation models for interactive machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 190–197, Barcelona.

Och, F. J. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41th Annual Conference of the Association for Computational Linguistics*, pages 160–167, Sapporo.

Och, F. J. and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, PA.

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

Ortiz, D., I. García-Varea, and F. Casacuberta. 2005. Thot: A toolkit to train phrase-based statistical translation models. In *Proceedings of the X Machine Translation Summit*, pages 141–148, Phuket.

Ortiz-Martínez, D. 2011. *Advances in Fully-Automatic and Interactive Phrase-Based Statistical Machine Translation*. Ph.D. thesis, Universitat Politècnica de València.

Ortiz-Martínez, D. and F. Casacuberta. 2014. The new Thot toolkit for fully automatic and interactive statistical machine translation. In *14th Annual Meeting of the European Association for Computational Linguistics: System Demonstrations*, pages 45–48, Gothenburg.

Ortiz-Martínez, D., I. García-Varea, and F. Casacuberta. 2010. Online learning for interactive statistical machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT)*, pages 546–554, Los Angeles, CA.

Ortiz-Martínez, D., L. A. Leiva, V. Alabau, I. García-Varea, and F. Casacuberta. 2011. An interactive machine translation system with online learning. In *Proceedings of the Association for Computational Linguistics Conference (System Demonstrations)*, pages 68–73, Portland, OR.

Ortiz-Martínez, D., J. González-Rubio, V. Alabau, G. Sanchis-Trilles, and F. Casacuberta. 2015. Integrating online and active learning in a computer-assisted translation workbench. In M. Carl, S. Bangalore, and M. Schaeffer, editors, *New Directions in Empirical Translation Process Research*, Springer, pages 57–76.

Papineni, K. A., S. Roukos, T. Ward, and W. Zhu. 2001. BLEU: A method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY.

SchlumbergerSema S.A., Instituto Tecnológico de Informática, Rheinisch Westfälische Technische Hochschule Aachen Lehrstul für Informatik VI, Recherche Appliquée en Linguistique Informatique Laboratory University of Montreal, Celer Soluciones,

Société Gamma, and Xerox Research Centre Europe. 2001. TT2. TransType2 - computer assisted translation. Project technical annex. Information Society Technologies (IST) Programme, IST-2001-32091.

Simard, M. and G. Foster. 2013. Pepr: Post-edit propagation using phrase-based statistical machine translation. In *Proceedings of the XIV Machine Translation Summit*, pages 191–198, Nice.

TAUS-Project. 2010. Postediting in practice. A TAUS Report. Technical report, TAUS – Enabling better translation.

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

Toutanova, K., H. T. Ilhan, and C. Manning. 2002. Extensions to HMM-based statistical word alignment models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 87–94, Philadelphia, PA.

Vogel, S., H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 836–841, Copenhagen.

Wäeschle, K., P. Simianer, N. Bertoldi, S. Riezler, and M. Federico. 2013. Generative and discriminative methods for online adaptation in SMT. In *Proceedings of the XIV Machine Translation Summit*, pages 11–18, Nice.

Watanabe, T., J. Suzuki, H. Tsukada, and H. Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the EMNLP-CONLL joint conference*, pages 764–733, Prague.

Zens, R., F. J. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In *Advances in Artificial Intelligence. 25. Annual German Conference on AI*, volume 2479 of *LNCS*. Springer Verlag, September, pages 18–32.