Online learning via dynamic reranking for Computer Assisted Translation

Pascual Martínez-Gómez, *Germán Sanchis-Trilles*, *Francisco Casacuberta*
Instituto Tecnológico de Informática
Universidad Politécnica de Valencia
{pmartinez,gsanchis,fcn}@dsic.upv.es

Abstract. New techniques for online adaptation in computer assisted translation are explored and compared to previously existing approaches. Under the online adaptation paradigm, the translation system needs to adapt itself to real-world changing scenarios, where training and tuning may only take place once, when the system is set-up for the first time. For this purpose, post-edit information, as described by a given quality measure, is used as valuable feedback within a dynamic reranking algorithm. Two possible approaches are presented and evaluated. The first one relies on the well-known perceptron algorithm, whereas the second one is a novel approach using the Ridge regression in order to compute the optimum scaling factors within a state-of-the-art SMT system. Experimental results show that such algorithms are able to improve translation quality by learning from the errors produced by the system on a sentence-by-sentence basis.

1 Introduction

Statistical Machine Translation (SMT) systems use mathematical models to describe the translation task and to estimate the probabilities involved in the process. [1] established the SMT grounds formulating the probability of translating a source sentence $\mathbf{x}$ into a target sentence $\hat{\mathbf{y}}$, as

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\text{argmax}} \ \Pr(\mathbf{y} \mid \mathbf{x}) \tag{1}$$

In order to capture context information, *phrase-based* (PB) models [2, 3] were introduced, widely outperforming single word models [4]. PB models were employed throughout this paper. The basic idea of PB translation is to segment the source sentence $\mathbf{x}$ into *phrases* (i.e. word sequences), then to translate each source phrase $\tilde{x}_k \in \mathbf{x}$ into a target phrase $\tilde{y}_k$, and finally reorder them to compose the target sentence $\mathbf{y}$.

Recently, the direct modelling of the posterior probability $\Pr(\mathbf{y} \mid \mathbf{x})$ has been widely adopted. To this purpose, different authors [5, 6] propose the use of the so-called log-linear models, where the decision rule is given by the expression

$$\begin{aligned}\hat{\mathbf{y}} &= \underset{\mathbf{y}}{\text{argmax}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y}) \\ &= \underset{\mathbf{y}}{\text{argmax}} \, \boldsymbol{\lambda} \cdot \mathbf{h}(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{y}}{\text{argmax}} \, s(\mathbf{x}, \mathbf{y})\end{aligned} \tag{2}$$

where $h_m(\mathbf{x}, \mathbf{y})$ is a score function representing an important feature for the translation of $\mathbf{x}$ into $\mathbf{y}$, M is the number of models (or features) and λ_m are the weights of the log-linear combination. $s(\mathbf{x}, \mathbf{y})$ represents the score of a hypothesis $\mathbf{y}$ given an input sentence $\mathbf{x}$, and is not treated as a probability since the normalisation term has been omitted. Common feature functions $h_m(\mathbf{x}, \mathbf{y})$ include different translation models (TM), but also distortion models or even the target language model (LM). Typically, $\mathbf{h}(\cdot|\cdot)$ and $\boldsymbol{\lambda}$ are estimated by means of training and development sets, respectively.

Adjusting both feature functions or log-linear weights leads to one important problem in SMT: whenever the text to be translated belongs to a different domain than the training or development corpora, translation quality diminishes significantly [4]. Hence, the *adaptation* problem is very common, where the objective is to improve systems trained on out-of-domain data by using very limited amounts of in-domain data.

Typically, the weights of the log-linear combination in Equation 2 are optimised by means of Minimum Error Rate Training (MERT) [7] in two basic steps. First, N best hypotheses are extracted for each one of the sentences of a development set. Next, the optimum $\boldsymbol{\lambda}$ is computed so that the best hypotheses in the $nbest$ list, according to a reference translation and a given metric, are ranked higher within such $nbest$ list. Then, these two steps are repeated until convergence, where no further changes in $\boldsymbol{\lambda}$ are observed. However, such algorithm has an important drawback. Namely, it requires a considerable amount of time to translate the development (or adaptation) set several times, and in addition it has been shown to be quite unstable whenever the amount of adaptation data is small [8]. For these reasons, using MERT in an online environment, where adaptation data is arriving constantly, is usually not appropriate.

Adapting a system to changing tasks is specially interesting in the Computer Assisted Translation (CAT) [9] and Interactive Machine Translation (IMT) paradigms [10, 11], where the collaboration of a human translator is essential to ensure high quality results. In these scenarios, the SMT system proposes a hypothesis to a human translator, who may amend the hypothesis to obtain an acceptable target sentence in a post-edition setup. The system is expected to learn dynamically from its own errors making the best use of every correction provided by the user by adapting the system *online*, i.e. without the need of an expensive complete retraining of the model parameters.

We analyse two online learning techniques to use such information to hopefully improve the quality of subsequent translations by adapting the scaling factors of the underlying log-linear model in a sentence-by-sentence basis.

In the next Section, existing online learning algorithms applied to SMT and CAT are briefly reviewed. In Section 3, common definitions and general terminology are established. In Section 4.1, we analyse how to apply the well-known perceptron algorithm in order to adapt the log-linear weights. Moreover, we propose in Section 4.2 a completely novel technique relying on the method of Ridge regression for learning the $\boldsymbol{\lambda}$ of Eq. 2 discriminatively. Experiments are reported in Section 5, a short study on metric correlation is done in Section 6 and conclusions can be found in the last Section.

2 Related Work

In [12], an online learning application is presented for IMT, where the models involved in the translation process are incrementally updated by means of an incremental version of the Expectation-Maximisation algorithm, allowing for the inclusion of new phrase pairs into the system. The difference between such paper and the present one is that the techniques proposed here do not depend on how the translation model has been trained, since it only relies on a dynamic reranking algorithm which is applied to a $nbest$ list, regardless of its origin. Furthermore, the present work deals with the problem of online learning as applied to the $\boldsymbol{\lambda}$ scaling factors, not to the $\mathbf{h}$ features. Hence, the work in [12] and the present one can be seen as complementary.

The perceptron algorithm was used in [13] to obtain more robust estimations of $\boldsymbol{\lambda}$, which is adapted in a batch setup, where the system only updates $\boldsymbol{\lambda}$ when it has seen a certain amount of adaptation data. In Section 4.1, a similar algorithm is used to adapt the model parameters, although in the present work the perceptron algorithm has been applied in an online manner, i.e. in an experimental setup where new bilingual sentence pairs keep arriving and the system must update its parameters after each pair.

In [14] the authors propose the use of the Passive-Aggressive framework [15] for updating the feature functions $\mathbf{h}$, combining both a memory-based MT system and a SMT system. Improvements obtained were very limited, since adapting $\mathbf{h}$ is a very sparse problem. For this reason, our intention is not to adapt the feature functions, but to adapt the log-linear weights $\boldsymbol{\lambda}$, which is shown in [8] to be a good adaptation strategy. In [8], the authors propose the use of a Bayesian learning technique in order to adapt the scaling factors based on an adaptation set. In contrast, in the present work our purpose is to perform online adaptation, i.e. to adapt the system parameters after each new sample has been provided to the system. In this paradigm, the SMT system always proposes a target sentence to the user who accepts or amends the whole sentence. If the user post-edits the hypothesis, we obtain a reference along with the hypothesis and the online-learning module is activated.

The contributions of this paper are mainly two. First, we propose a new application of the perceptron algorithm for online learning in SMT. Second, we propose a new discriminative technique for incrementally learning the scaling factors $\boldsymbol{\lambda}$, which relies on the concept of Ridge regression, and which proves to perform better than the perceptron algorithm in all analysed language pairs. Although applied here to phrase-based SMT, both strategies can be applied to rerank a $nbest$ list, which implies that they do not depend on a specific training algorithm or a particular SMT system.

3 Online learning in CAT

In general, in an online learning framework, the learning algorithm processes observations sequentially. After every input, the system makes a prediction and then receives a feedback. The information provided by this feedback can range from a simple opinion of how good the system's prediction was, to the true label of the input in completely supervised environments. The purpose of online learning algorithms is to modify its prediction mechanisms in order to improve the quality of future decisions. Specifically, in a CAT scenario, the SMT system receives a sentence in a source language and then outputs a sentence in a target language as a prediction based on its models. The user, typically a professional human translator, post-edits the system's hypothesis thus producing a reference translation $\mathbf{y}^{\tau}$. Such a reference can be used as a supervised feedback. Our intention is to learn from that interaction. Then, Eq. 2 is redefined as follows

$$\begin{aligned}\hat{\mathbf{y}} &= \underset{\mathbf{y}}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m^t h_m^t(\mathbf{x}, \mathbf{y}) \\ &= \underset{\mathbf{y}}{\operatorname{argmax}}\, \boldsymbol{\lambda}^t \cdot \mathbf{h}^t(\mathbf{x}, \mathbf{y}) \end{aligned} \tag{3}$$

where both the feature functions $\mathbf{h}^t$ and the log-linear weights $\boldsymbol{\lambda}^t$ vary according to the samples $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_{t-1}, \mathbf{y}_{t-1})$ seen before time t. We can either apply online learning techniques to adapt $\mathbf{h}^t$, or $\boldsymbol{\lambda}^t$, or both at the same time. In this paper, however, we will only attempt to adapt $\boldsymbol{\lambda}^t$, since adapting $\mathbf{h}^t$ is a very sparse problem implying the adaptation of several million parameters, which is not easily feasible when considering an on-line, sentence-by-sentence scenario.

Let $\mathbf{y}$ be the hypothesis proposed by the system, and $\mathbf{y}^*$ the best hypothesis the system is able to produce in terms of translation quality (i.e. the most similar sentence with respect to $\mathbf{y}^\tau$). Then, our purpose is to adapt the model parameters ($\boldsymbol{\lambda}^t$ in this case) so that $\mathbf{y}^*$ is rewarded (i.e. achieves a higher score according to Eq. 2).

We define the difference in translation quality between the proposed hypothesis $\mathbf{y}$ and the best hypothesis $\mathbf{y}^*$ in terms of a given quality measure $\mu(\cdot)$:

$$l(\mathbf{y}) = |\mu(\mathbf{y}) - \mu(\mathbf{y}^*)|, \tag{4}$$

where the absolute value has been introduced in order to preserve generality, since in SMT some of the quality measures used, such as TER [16], represent an error rate (i.e. the lower the better), whereas others such as BLEU [17] measure precision (i.e. the higher the better). In addition, the difference in probability between $\mathbf{y}$ and $\mathbf{y}^*$ is proportional to $\phi(\mathbf{x})$, which is defined as

$$\phi(\mathbf{y}) = s(\mathbf{x}, \mathbf{y}^*) - s(\mathbf{x}, \mathbf{y}). \tag{5}$$

Ideally, we would like that increases or decreases in $l(\cdot)$ correspond to increases or decreases in $\phi(\cdot)$, respectively: if a candidate hypothesis $\mathbf{y}$ has a translation quality $\mu(\mathbf{y})$ which is very similar to the translation quality provided by $\mu(\mathbf{y}^*)$, we would like that such fact is reflected in the translation score s, i.e. $s(\mathbf{x}, \mathbf{y})$ is very similar to $s(\mathbf{x}, \mathbf{y}^*)$. Hence, the purpose of our online procedure should be to promote such correspondence after processing sample t.

A coarse-grained technique for tackling with the online learning problem in SMT implies adapting the log-linear weights $\boldsymbol{\lambda}$. The aim is to compute the optimum weight vector $\hat{\boldsymbol{\lambda}}_t$ for translating the sentence pair observed at time t and then update $\boldsymbol{\lambda}$ as:

$$\boldsymbol{\lambda} = \boldsymbol{\lambda}_{t-1} + \alpha \hat{\boldsymbol{\lambda}}_t \tag{6}$$

for a certain learning rate α.

The information that is usually taken into account to compute $\hat{\boldsymbol{\lambda}}_t$ is more general and imprecise than the information used when adapting feature functions, but the variation in the score of Eq. 2 can be higher since we will be modifying the scaling factors of the log-linear model. That is, when adapting the system to a different domain, we are going to adjust the importance of every single model to a new task in an online manner.

4 Online learning algorithms

4.1 Perceptron in CAT

The perceptron algorithm [18, 19] is an error driven algorithm that estimates the weights of a linear combination of features by comparing the output $\mathbf{y}$ of the system with respect to the true label $\mathbf{y}^{\tau}$ of the corresponding input $\mathbf{x}$. It iterates through the set of samples a certain number of times (epochs), or until a desired convergence is achieved.

The implementation in this work follows the proposed application of a perceptron-like algorithm in [13]. However, for comparison reasons in our CAT framework, the perceptron algorithm will not visit a sample again after being processed once.

Using feature vector $\mathbf{h}(\mathbf{x}, \mathbf{y})$ of the system's hypothesis $\mathbf{y}$ and feature vector $\mathbf{h}(\mathbf{x}, \mathbf{y}^*)$ of the best hypothesis $\mathbf{y}^*$ from the $nbest(\mathbf{x})$ list, the update term is computed as follows:

$$\boldsymbol{\lambda}_t = \boldsymbol{\lambda}_{t-1} + \epsilon \cdot \operatorname{sign}(\mathbf{h}(\mathbf{x}, \mathbf{y}^*) - \mathbf{h}(\mathbf{x}, \mathbf{y})) \tag{7}$$

where ϵ can be interpreted as the learning rate.

4.2 Discriminative regression

The problem of finding $\hat{\boldsymbol{\lambda}}_t$ such that higher values in $s(\mathbf{x}, \mathbf{y})$ correspond to improvements in the translation quality $\mu(\mathbf{y})$ as described in Section 3 can be viewed as finding $\hat{\boldsymbol{\lambda}}_t$ such that differences in scores $\phi(y)$ of two hypotheses approximate their difference in translation quality $l(y)$. So as to formalise this idea, let us first define some matrices.

Let $nbest(\mathbf{x})$ be the list of N best hypotheses computed by our TM for sentence $\mathbf{x}$. Then, a matrix $\mathrm{H}_{\mathbf{x}}$ of size $N \times M$, where M is the number of features in Eq. 2, containing the feature functions $\mathbf{h}$ of every hypothesis can be defined such that

$$\mathbf{s}_{\mathbf{x}} = \mathrm{H}_{\mathbf{x}} \cdot \boldsymbol{\lambda}_t \tag{8}$$

where $\mathbf{s}_{\mathbf{x}}$ is a column vector of N entries with the log-linear score combination of every hypothesis in the $nbest(\mathbf{x})$ list. Additionally, let $\mathrm{H}^*_{\mathbf{x}}$ be a matrix with N rows such that

$$\mathrm{H}^*_{\mathbf{x}} = \begin{bmatrix} \mathbf{h}(\mathbf{x}, \mathbf{y}^*) \\ \vdots \\ \mathbf{h}(\mathbf{x}, \mathbf{y}^*) \end{bmatrix}, \tag{9}$$

and $\mathrm{R}_{\mathbf{x}}$ the difference between $\mathrm{H}^*_{\mathbf{x}}$ and $\mathrm{H}_{\mathbf{x}}$:

$$\mathrm{R}_{\mathbf{x}} = \mathrm{H}^*_{\mathbf{x}} - \mathrm{H}_{\mathbf{x}} \tag{10}$$

The key idea for scaling factor adaptation is to find a vector $\hat{\boldsymbol{\lambda}}$ such that differences in scores are reflected as differences in the quality of the hypotheses. That is,

$$\mathrm{R}_{\mathbf{x}} \cdot \hat{\boldsymbol{\lambda}}_t \propto \mathbf{l}_{\mathbf{x}} \tag{11}$$

where $\mathbf{l_x}$ is a column vector of N rows such that $\mathbf{l_x} = [l(\mathbf{y}_1) \dots l(\mathbf{y}_n) \dots l(\mathbf{y}_N)]'$, $\forall \mathbf{y}_n \in nbest(\mathbf{x})$. The objective is to find $\hat{\boldsymbol{\lambda}}$ such that

$$\hat{\boldsymbol{\lambda}} = \underset{\boldsymbol{\lambda}}{\text{argmin}} \, |\mathrm{R}_\mathbf{x} \cdot \boldsymbol{\lambda} - \mathbf{l_x}| \tag{12}$$

$$= \underset{\boldsymbol{\lambda}}{\text{argmin}} \, ||\mathrm{R}_\mathbf{x} \cdot \boldsymbol{\lambda} - \mathbf{l_x}||^2 \tag{13}$$

where $|| \cdot ||^2$ is the Euclidean norm. Although Eqs. 12 and 13 are equivalent (i.e. the $\boldsymbol{\lambda}$ that minimises the first one also minimises the second one), Eq. 13 allows for a direct implementation thanks to the Ridge regression[1], such that $\hat{\boldsymbol{\lambda}}$ can be computed as the solution of the overdetermined system $\mathrm{R}_\mathbf{x} \cdot \hat{\boldsymbol{\lambda}} = \mathbf{l_x}$, given by the expression

$$\hat{\boldsymbol{\lambda}} = \left(\mathrm{R}'_\mathbf{x} \cdot \mathrm{R}_\mathbf{x} + \beta I\right)^{-1} \mathrm{R}'_\mathbf{x} \cdot \mathbf{l_x} \tag{14}$$

where a small β is used as a regularisation term to ensure $\mathrm{R}'_\mathbf{x} \cdot \mathrm{R}_\mathbf{x}$ has an inverse.

5 Experiments

5.1 Experimental setup

Given that a true CAT scenario is very expensive for experimentation purposes, since it requires a human translator to correct every hypothesis, in this paper we will be simulating such scenario by using the reference present in the test set. However, such reference will be fed one at a time, given that this would be the case in an online CAT process.

Translation quality will be assessed by means of the BLEU [17] and TER [16] scores. BLEU measures n-gram precision with a penalty for sentences that are too short, whereas TER is an error metric that computes the minimum number of edits required to modify the system hypotheses so that they match the references. Possible edits include insertion, deletion, substitution of single words and shifts of word sequences. For computing $\mathbf{y}^*$ as described in Section 3, either BLEU or TER will be used, depending on the evaluation measure reported (i.e. when reporting TER, TER will be used for computing $\mathbf{y}^*$). However, it must be noted that BLEU is not well defined at the sentence level, since it implements a geometrical average of n-grams which is zero whenever there is no common 4-gram between reference and hypothesis, even if the reference has only three words. Hence, $\mathbf{y}^*$ is not always well defined when considering BLEU. Such samples will not be considered within the online procedure. Another consideration is that BLEU and TER might not be correlated, i.e. improvements in TER do not necessarily mean improvements in BLEU. This is analysed more in detail in Section 6.

As baseline system, we trained a SMT system on the Europarl training data, in the partition established in the Workshop on SMT of the NAACL 2009[2]. Specifically, we will train our initial SMT system by using the training and development data provided that year. The Europarl corpus [20] is built from the transcription of European Parliament speeches published on the web. Statistics are provided in Table 1.

[1] Also known as Tikhonov regularisation in statistics.

[2] http://www.statmt.org/wmt10/

Table 1. Characteristics of Europarl corpus. Dev. stands for Development, OoV for "Out of Vocabulary" words, K for thousands of elements and M for millions of elements.

		Es	En	Fr	En	De	En
Training	Sentences	1.3M		1.2M		1.3M	
	Running words	27.5M	26.6M	28.2M	25.6M	24.9M	26.2M
	Vocabulary	125.8K	82.6K	101.3K	81.0K	264.9K	82.4K
Development	Sentences	2000		2000		2000	
	Running words	60.6K	58.7K	67.3K	48.7K	55.1K	58.7K
	OoV. words	164	99	99	104	348	103

Table 2. Characteristics of NC test sets. OoV stands for "Out of Vocabulary" words w.r.t. the Europarl training set. Data statistics were again collected after tokenizing and lowercasing.

		Es	En	Fr	En	De	En
Test 08	Sentences	2051		2051		2051	
	Running words	52.6K	49.9K	55.4K	49.9K	55.4K	49.9K
	OoV. words	1029	958	998	963	2016	965
Test 09	Sentences	2525		2051		2051	
	Running words	68.1K	65.6K	72.7K	65.6K	62.7K	65.6K
	OoV. words	1358	1229	1449	1247	2410	1247

The open-source MT toolkit Moses[3] [21] was used in its default setup, and the 14 weights of the log-linear combination were estimated using MERT [22] on the Europarl development set. Additionally, a 5-gram LM with interpolation and Kneser-Ney smoothing [23] was estimated using the SRILM [24] toolkit.

To test the adaptation performance of different online learning strategies, we also considered the use of two News Commentary (NC) test sets, from the 2008 and 2009 ACL shared tasks on SMT. Statistics of these test sets can be seen in Table 2.

Experiments were performed on the English–Spanish, English–German and English–French language pairs, in both directions and for NC test sets of 2008 and 2009. However, for space reasons, we only report results for the 2009 test set from the English–*foreign* pair, since this year's SMT shared task of the ACL focused on translating from English into other languages. Nevertheless, the results presented here were found to be coherent in all the experiments conducted, unless stated otherwise.

As for the different parameters adjustable in the algorithms described in Section 4, they were all set according to preliminary investigation as follows:

- Section 4.1: $\epsilon = 0.001$
- Section 4.2: $\alpha = 0.005$, $\beta = 0.01$

For Section 4.1, instead of using the true best hypothesis, the best hypothesis within a given $nbest(\mathbf{x})$ list was selected.

5.2 Experimental results

The result of applying the different online learning algorithms described in Section 4 can be seen in Fig. 1. `percep.` stands for the technique described in Section 4.1,

[3] Available from http://www.statmt.org/moses/

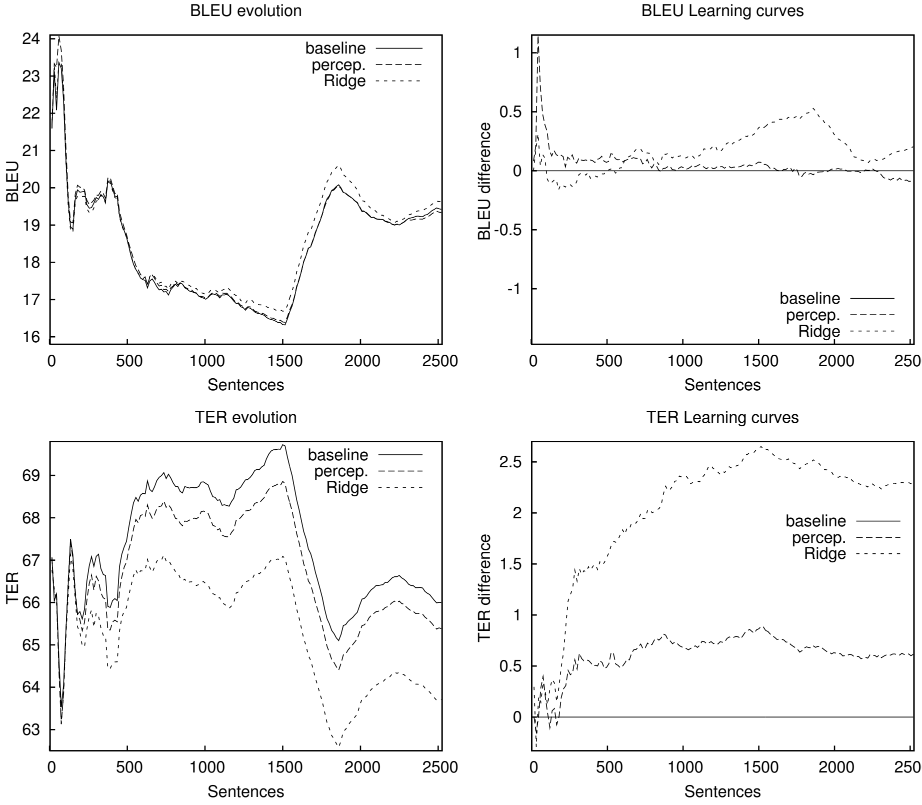

Fig. 1. BLEU/TER evolution and learning curves for English→French translation, considering all 2525 sentences within the NC 2009 test set. For clarity, only 1 every 15 points has been drawn. `percep.` stands for perceptron and `Ridge` for the technique described in Section 4.2.

and `Ridge` for the one described in Section 4.2. In the plots shown in this figure, the translation pair was English→French, the test set was the NC 2009 test set, and the size of the considered $nbest$ list was 1000. The two plots on the left display the BLEU and TER scores averaged up to the considered t-th sentence. The reason for plotting the average BLEU/TER is that plotting individual sentence BLEU and TER scores would result in a very chaotic, unreadable plot given that differences in translation quality between two single sentences may be very big; in fact, such chaotic behaviour can still be seen in the first 100 sentences. The two plots on the right display the difference in translation quality between the two online learning techniques and the baseline.

The analysed online learning procedures perform better in terms of TER than in terms of BLEU (Fig. 1). Again, since BLEU is not well defined at the sentence level, learning methods that depend on BLEU being computed at the sentence level may be severely penalised. Although it appears that the learning curves peak at about 1500 sentences, this finding is not coherent throughout all experiments carried out, since such peak ranges from 300 to 2000 in other cases. This means that the particular shape of the learning curves depends strongly on the chosen test set, and that the information that can be extracted is only whether or not the implemented algorithms provide improvements.

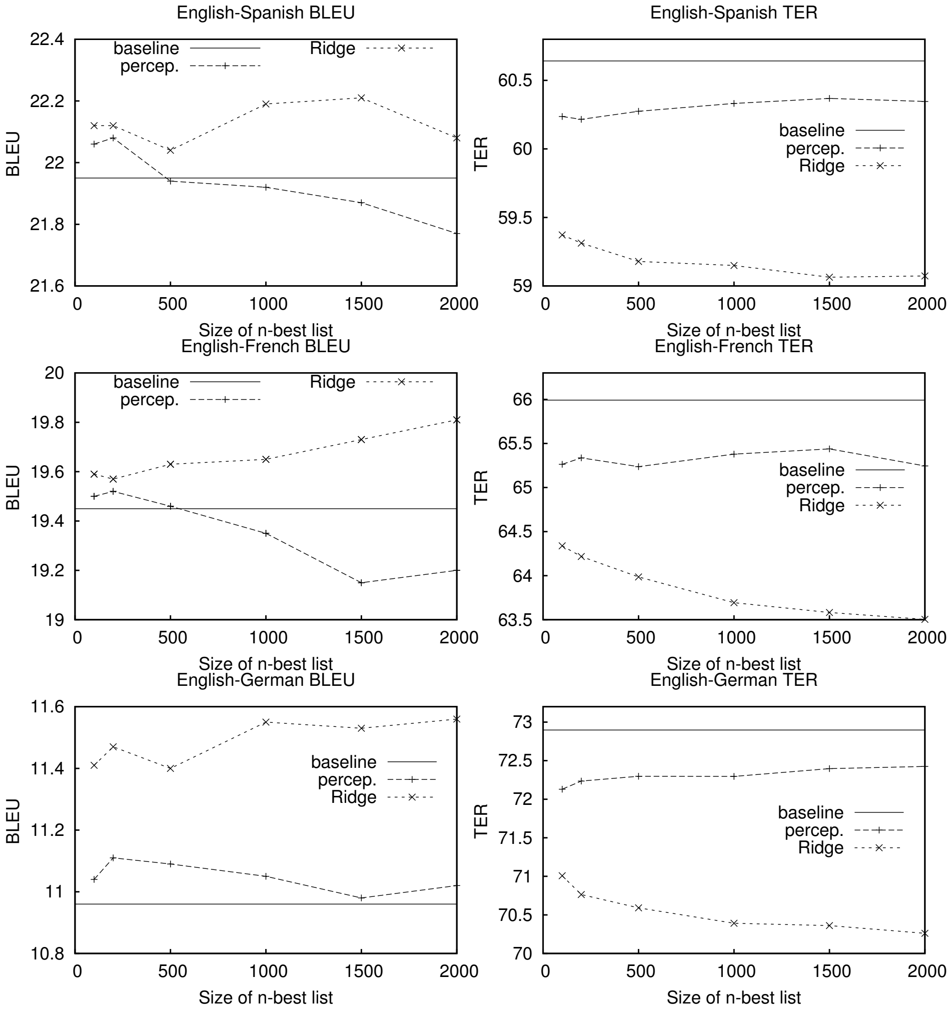

Fig. 2. Final BLEU and TER scores for the NC 2009 test set, for all language pairs considered. `percep.` stands for perceptron and `Ridge` for the technique described in Section 4.2.

In addition, it can be seen that the best performing method, both in terms of TER and in terms of BLEU, is the one described in Section 4.2. However, in order to assess these differences, further experiments were conducted. Furthermore, and in order to evidence the final improvement in translation quality that can be obtained after seeing a complete test set, the final translation quality obtained with varying sizes of $nbest(\mathbf{x})$ was measured. The results of such experiments can be seen in Fig. 2. Although the differences are sometimes scarce, they were found to be coherent in all the considered cases, i.e. for all language pairs, all translation directions, and all test sets.

Although the final BLEU and TER scores are reported for the whole considered test set, all of the experiments described here were performed following an online CAT approach: each reference sentence was used for adapting the system parameters after such sentence has been translated and its translation quality has been assessed. For this

source	in the first round , half of the amount is planned to be spent .	
reference	au premier tour , la moitié de cette somme va être dépensée .	
baseline	dans la première phase , la moitié de la somme prévue pour être dépensé .	8
ridge	au premier tour , la moitié de la somme prévue pour être dépensé .	4
perceptron	dans un premier temps , la moitié de la somme prévue pour être dépensé .	7
source	it enables the probes to save a lot of fuel .	
reference	ainsi , les sondes peuvent économiser beaucoup de carburant .	
baseline	il permet à la probes de sauver une quantité importante de carburant .	10
ridge	il permet aux probes à économiser beaucoup de carburant .	5
perceptron	il permet à la probes à économiser beaucoup de carburant .	6

Fig. 3. Example of translations found in the corpus. The third column corresponds to the number of necessary editions to convert the string into the reference.

reason, the final reported translation score corresponds to the average over the complete test set, even though the system was still not adapted at all for the first samples.

In Fig. 2 it can be observed how `Ridge` seems to provide better translation quality when the size of $nbest(\mathbf{x})$ increases, which is a desirable behaviour.

Fig. 3 shows specific examples of the performance of the presented methods. For the first sentence, the baseline produces a phrase that, although being correct, does not match the reference; in this case, the discriminative Ridge regression finds the correct phrase in one of the sentences of the $nbest$ list. In the second example, discriminative regression and perceptron are able to find more accurate translations than the baseline.

One last consideration involves computation time. When adapting $\boldsymbol{\lambda}$, the procedures implemented take about 100 seconds to rerank the complete test set. We consider this fact important, since in a CAT scenario the user is waiting actively for the system to produce a hypothesis.

6 Metric correlation

From the experiments, it was observed that online learning strategies that optimise a certain quality measure do not necessarily optimise other measures.

To analyse such statement, 100.000 weight vectors $\boldsymbol{\lambda}$ of the log-linear model were randomly generated and a static rerank of a fixed $nbest(\mathbf{x})$ list of hypotheses was performed for every sentence in a test set. For every weight vector configuration, BLEU (B) and TER (T) were evaluated for the test set of NC 2008 Spanish → English.

The correlation as defined by the covariance (cov) divided by the product of standard deviations (σ)

$$\rho_{B,T} = \frac{\mathrm{cov}(B,T)}{\sigma_B \sigma_T} \tag{15}$$

returned a value $\rho_{B,T} = -0.23798$. This result suggests that even if such correlation is not specially strong, one can expect to optimise TER (as an error metric) only to certain extent when optimising BLEU (as a precision metric), and vice-versa. A plot of the translation quality yielded by the random weight configurations is presented in Fig.4. It can be observed that it is relatively frequent to obtain low BLEU scores after optimising TER (high density area in the bottom left part of the graph). On the other hand, if BLEU is optimised, TER scores are reasonably good (right side of the plot).

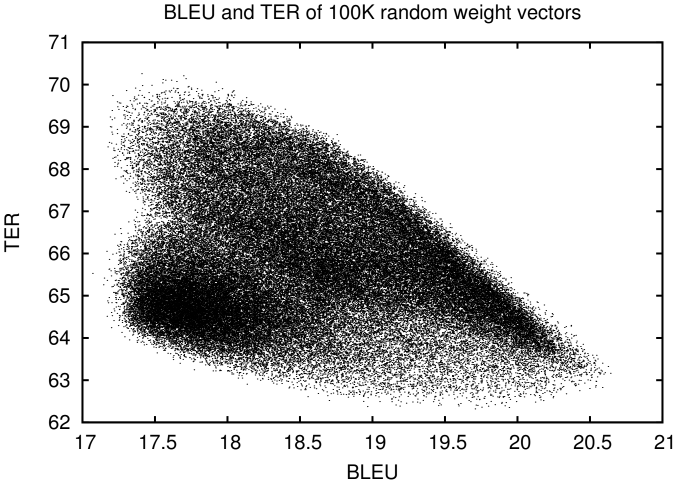

Fig. 4. Correlation between BLEU and TER of 100.000 configurations of $\boldsymbol{\lambda}$. A slightly negative correlation can be appreciated, although not strong.

7 Conclusions and future work

In this paper, two different online learning algorithms have been applied to SMT. The first one is a well-known algorithm, namely the perceptron algorithm, whereas the second one is completely novel and relies on the concept of discriminative regression. Both of these strategies have been applied to adapt the log-linear weights of a state-of-the-art SMT system, providing interesting improvements.

From the experiments conducted, it emerges that discriminative regression, as implemented for SMT in this paper, provides a larger gain than the perceptron algorithm, and is able to provide improvements from the very beginning and in a consistent manner, in all language pairs analysed.

Although BLEU is probably the most popular quality measure used in MT, it has been shown that its use within online, sentence-by-sentence learning strategies may not be very adequate. In order to cope with the discrepancies between optimising BLEU and TER, we plan to analyse the effect of combining both measures, and also consider other measures such as NIST which are also well defined at the sentence level.

We plan to analyse the application of these learning algorithms to feature functions to study how the behaviour of such techniques evolves in much sparser problems.

Acknowledgements

This paper is based upon work supported by the EC (FEDER/FSE) and the Spanish MICINN under projects MIPRCV “Consolider Ingenio 2010” (CSD2007-00018) and iTrans2 (TIN2009-14511). Also supported by the Spanish MITyC under the erudito.com (TSI-020110-2009-439) project, by the Generalitat Valenciana under grant Prometeo/2009/014 and scholarship GV/2010/067 and by the UPV under grant 20091027.

References

1. Brown, P., Pietra, S.D., Pietra, V.D., Mercer, R.: The mathematics of machine translation. In: Computational Linguistics. Volume 19. (1993) 263–311
2. Zens, R., Och, F., Ney, H.: Phrase-based statistical machine translation. In: Proc. of KI'02. (2002) 18–32
3. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: Proc. HLT/NAACL'03. (2003) 48–54
4. Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., Schroeder, J.: (meta-) evaluation of machine translation. In: Proc. of the Workshop on SMT, ACL (2007) 136–158
5. Papineni, K., Roukos, S., Ward, T.: Maximum likelihood and discriminative training of direct translation models. In: Proc. of ICASSP'98. (1998) 189–192
6. Och, F., Ney, H.: Discriminative training and maximum entropy models for statistical machine translation. In: Proc. of the ACL'02. (2002) 295–302
7. Och, F., Zens, R., Ney, H.: Efficient search for interactive statistical machine translation. In: Proc. of EACL'03. (2003) 387–393
8. Sanchis-Trilles, G., Casacuberta, F.: Log-linear weight optimisation via bayesian adaptation in statistical machine translation. In: Proceedings of COLING2010, Beijing, China (2010)
9. Callison-Burch, C., Bannard, C., Schroeder, J.: Improving statistical translation through editing. In: Proc. of 9th EAMT workshop "Broadening horizons of machine translation and its applications", Malta (2004)
10. Barrachina, S., et. al.: Statistical approaches to computer-assisted translation. Computational Linguistics **35** (2009) 3–28
11. Casacuberta, F., et. al.: Human interaction for high quality machine translation. Communications of the ACM **52** (2009) 135–138
12. Ortiz-Martínez, D., García-Varea, I., Casacuberta, F.: Online learning for interactive statistical machine translation. In: Proceedings of NAACL HLT, Los Angeles (2010)
13. España-Bonet, C., Màrquez, L.: Robust estimation of feature weights in statistical machine translation. In: 14th Annual Conference of the EAMT. (2010)
14. Reverberi, G., Szedmak, S., Cesa-Bianchi, N., et al.: Deliverable of package 4: Online learning algorithms for computer-assisted translation (2008)
15. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. Journal of Machine Learning Research **7** (2006) 551–585
16. Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J.: A study of translation edit rate with targeted human annotation. In: Proc. of AMTA, Cambridge, MA, USA (2006)
17. Papineni, K., Roukos, S., Ward, T., Zhu, W.: Bleu: A method for automatic evaluation of machine translation. In: Proc. of ACL'02. (2002)
18. Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review **65** (1958) 386–408
19. Collins, M.: Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In: EMNLP 2002, Philadelphia, PA, USA (2002) 1–8
20. Koehn, P.: Europarl: A parallel corpus for statistical machine translation. In: Proc. of the MT Summit X. (2005) 79–86
21. Koehn et al., P.: Moses: Open source toolkit for statistical machine translation. In: Proc. of the ACL Demo and Poster Sessions, Prague, Czech Republic (2007) 177–180
22. Och, F.: Minimum error rate training for statistical machine translation. In: Proc. of ACL'03. (2003) 160–167
23. Kneser, R., Ney, H.: Improved backing-off for m-gram language modeling. IEEE Int. Conf. on Acoustics, Speech and Signal Processing **II** (1995) 181–184
24. Stolcke, A.: SRILM – an extensible language modeling toolkit. In: Proc. of ICSLP'02. (2002) 901–904