Noname manuscript No.
(will be inserted by the editor)

Discriminative ridge regression algorithm for adaptation in Statistical Machine Translation

Mara Chinea-Rios · Germán Sanchis-Trilles · Francisco Casacuberta

Received: date / Accepted: date

Abstract We present a simple and reliable method for estimating the log-linear weights of a state-of-the-art machine translation system, which takes advantage of the method known as Discriminative Ridge Regression (DRR). Since inappropriate weight estimations lead to a wide variability of translation quality results, reaching a reliable estimate for such weights is critical for machine translation research. For this reason, a variety of methods have been proposed to reach reasonable estimates. In this paper, we present an algorithmic description and empirical results proving that DRR is able to provide comparable translation quality when compared to state-of-the-art estimation methods (i.e., MERT ([20]) and MIRA ([6])), with a reduction of computational cost. Moreover, the empirical results reported are coherent across different corpora and language pairs.

1 Introduction

Machine Translation (MT) is a specific subfield of Natural Language Processing (NLP) and studies the way in which an automatic system can automatize the translation process. Different approaches have been developed and used during the last years involving different paradigms and with different level of success. Statistical Machine Translation (SMT) is an important alternative to other MT standards and is currently state-at-the-art.

The foundation for modern SMT, the pattern recognition approach to machine translation is established in [3,13], by formulating the SMT problem as follows: given the input sentence $\mathbf{x} = x_1, \ldots, x_j, \ldots, x_J$ in certain source language, we need to find the equivalent sentence $\mathbf{y} = y_1, \ldots, y_i, \ldots, y_I$ in the target language. From all the possible translation sentences in the target language, the SMT process aims to find the sentence $\hat{\mathbf{y}}$ that maximizes the posterior probability:

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

A significant breakthrough in SMT was reached by modelling the translation process, and the posterior probability $Pr(\mathbf{y} \mid \mathbf{x})$, by means of log-linear models [21,13]: The log-linear models are defined as follow:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y}) = \underset{\mathbf{y}}{\operatorname{argmax}}\, \boldsymbol{\lambda} \cdot \mathbf{h}(\mathbf{x}, \mathbf{y}) \tag{2}$$

In this framework, $h_m(\mathbf{x}, \mathbf{y})$ is a score function that represents 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 that act as scaling factors of the score functions. The purpose of the scaling factors λ_m is to tune the discriminative power of their corresponding feature functions $h_m(\mathbf{x}, \mathbf{y})$.

In state-of-the-art phrase-based SMT, commons features are fourteen in total, including a different phrase-based translation model, a language model, a distortion model and word and phrase penalties. The translation models features describe the correspondence of words or sequences of words between languages, distortion models that account for necessary reorderings of blocks of words, and language models that account for the well-formedness of the translated sentence.

The research leading to these results were partially supported by projects CoMUN-HaT-TIN2015-70924-C2-1-R (MINECO/FEDER) and PROMETEO/2018/004.

Mara Chinea-Rios · Francisco Casacuberta
Pattern Recognition and Human Language Technology Research Center, Universitat Politècnica de València, Spain
Tel.: +34-695739050
E-mail: machirio@prhlt.upv.es, fcn@prhlt.upv.es

Germán Sanchis-Trilles
Sciling, Valencia, Spain,
E-mail: gsanchis@sciling.es

Typically, $\mathbf{h}(\mathbf{x}, \mathbf{y})$ and λ are estimated by means of training and development corpora, respectively. This leads to one important problem in SMT: whenever the training corpus belongs to a different domain than the text to be translated, the translation quality decrease significantly [4, 13].

Estimating the $\boldsymbol{\lambda}$ weights according to the importance of each single model within the specific task is often called *tuning* [19]. To exemplify this, consider for instance that the original translation model has been trained on a domain in which sentences tend to be long, such as in a European parliamentary debate. Then, if we intend to translate another domain in which sentences are rather short, such as sentences of technical manuals, we would adjust the log-linear weights $\boldsymbol{\lambda}$ conveniently to reflect this fact. So, the goal of these tuning methods is to find the best set of weights which will offer the best translation quality for a specific text using an in-domain development corpus.

In this work, we propose a method to optimize the log-linear weights in a log-linear model (Equation 2). This novel method is based on the Discriminative Ridge Regression [17] technique; to our knowledge this is the first time that, the ridge regression is used in log-linear weights optimization task. We present and analyse two different variations for estimating the log-linear weights in an off-line adaptation scenario.

The main contributions of this paper are:

- We present an optimisation algorithm for the estimation of the log-linear weights in an SMT adaptation scenario. This method is based on the Discriminative Ridge Regression technique.
- We propose an algorithmic description of DRR in both variants, sentence-by-sentence and batch.
- We evaluate empirically the DRR algorithm proposed in three different domains across to different language pairs.
- We provide a thorough comparison with state-of-the-art $\boldsymbol{\lambda}$ estimation methods, such as Minimum Error Rate Training (MERT) [20], and batch Margin Infused Relaxed Algorithm (MIRA) [6].

The rest of this paper is structured as follows. In Section 2, we perform a brief review of current approaches to log-linear weight estimation in SMT. In Section 3, we describe the algorithmic approach for applying DDR for the estimation of $\boldsymbol{\lambda}$ in a batch scenario. In Section 4, the experimental design and empirical results are detailed. Conclusions and future work are explained in Section 5.

2 Related-work

The log-linear weight adaptation problem is very common in SMT, where the goal is to improve the performance of systems trained on out-of-domain corpora by using an optimization algorithm [13]. In other words, how to find a set of weights which will offer the best translation quality after training the SMT system on a training corpus that belongs to a different domain than the text to be translated. To this end, numerous methods have been proposed.

The most popular algorithm for optimising the scaling factors λ is the one proposed by [20], commonly referred to as Minimum Error Rate Training (MERT) . The MERT algorithm implements a coordinate-wise global optimisation and consists of two basic steps. First, n-best hypotheses are extracted for each one of the sentences of a given development set. Next, the optimum λ is computed so that the best hypothesis in the n-best list, according to a reference translation and a given automatic metric, are the ones that the search algorithm would produce. These two steps are iteratively repeated until convergence, where the weights remain unchanged.

However, MERT presents two significant drawbacks: firstly, that it heavily relies on having a fair amount of data available as development set. In addition, that it only relies on the data in the development set. These two problems can produce over-fitting to the specific characteristics of the development corpus, which implies that MERT fails to provide appropriate estimates [7, 23, 24].

Various alternatives to MERT have been proposed, motivated primarily by the previous problems. For instance,[6, 9, 8] propose the use of the Margin Infused Relaxed Algorithm (MIRA) for the task of weight optimisation. More recently, [26] proposed to view the tuning problem as a set of operations over a specific semiring. Alternatively, [10, 18] proposed to view the problem as a ranking problem, where each step of the tuning procedure consists in deciding whether a given translation hypothesis should be ranked lower or higher within the set of possible hypotheses that are provided by the search procedure.

Tellingly, in the entire proceedings of ACL 2016[1], only one paper describing a statistical MT system cited the use of MIRA for tuning [16], while all others used MERT.

3 Discriminative ridge regression for SMT

In this section, the Discriminative Ridge Regression method for estimating $\boldsymbol{\lambda}$ is introduced. DRR, as proposed by [17], uses the concept of ridge regression to develop a discriminative algorithm for estimating $\boldsymbol{\lambda}$ on-line, i.e., as new adaptation samples are introduced into the system.

The key idea is to to find a configuration of the weight vector using all the hypotheses within a given N-best list, so that good hypothesis are rewarded, and bad hypothesis are penalised, trying to narrow the correlation between the

[1] www.aclweb.org/anthology/P/P16/

score function σ, and the quality criterion used. Since DRR was proposed for an on-line computer-assisted translation scenario, it requires an N-best list of hypotheses for each one of the sentences that are evaluated by the professional translator post-editing the system's output.

In this paper, we propose two different variations of DRR for optimising the log-linear weights λ. We named the first option sentence-by-sentence DRR. In Sentence-by-sentence DRR, $\boldsymbol{\lambda}$ is obtained by adjusting the vector after observing each sentence of a development corpus. The second alternative is batch DRR. In this case, the optimisation process is performed by using a batch of development sentences.

3.1 Sentence-by-sentence DRR

In this section, we present Sentence-by-sentence DRR. Algorithm 1 shows the procedure. Here, we have a bilingual development corpus A, where S is the number of sentences in development corpus A ($S = |A|$), $s \in \{1 \dots S\}$, and I is the maximum number of epochs desired.

Data: Development corpus A
Result: $\boldsymbol{\lambda}$
Initialize: $\boldsymbol{\lambda}^0$;
forall *desired number of iterations* I **do**
  forall *number sentences in dev-corpus* $S = |A|$ **do**
    optimization: compute vector $\check{\boldsymbol{\lambda}}_{\boldsymbol{i}}^{\boldsymbol{s}}$;
    estimation: $\boldsymbol{\lambda}_i^s = (1-\alpha)\boldsymbol{\lambda}_i^{s-1} + \alpha\check{\boldsymbol{\lambda}}_{\boldsymbol{i}}^{\boldsymbol{s}}$;
  end
end
selection: output vector $\boldsymbol{\lambda}_{\boldsymbol{I}}^{\boldsymbol{S}}$

Algorithm 1: Pseudo-code for DRR estimating $\boldsymbol{\lambda}$ as described in Section 3

During the **optimization** step in Algorithm 1, we obtain the vector $\check{\boldsymbol{\lambda}}$ for each one of the development sentences $\mathbf{x}_s$. Within DRR, this optimisation is performed by computing the solution to an overdetermined system, described in detail in next section, so that changes in the scoring function σ are correlated to changes in the objective function (potentially some automatic evaluation metric like BLEU or TER).

3.1.1 Sentence-based optimisation in DRR

As exposed in the previous section, DRR obtains $\boldsymbol{\lambda}$ by computing the best vector for each one of the sentences of a development corpus. In order to compute the new log-linear weight vector $\boldsymbol{\lambda}^s$, the previously learned $\boldsymbol{\lambda}^{s-1}$ needs to be combined with an appropriate update step $\check{\boldsymbol{\lambda}}^s$. The aim is to compute an appropriate update term $\check{\boldsymbol{\lambda}}^s$ that better fits the translation search space (approximated as an n-best list) of the development sentence pair observed at s. This is often done as a linear combination [27], where:

$$\boldsymbol{\lambda}^s = (1-\alpha)\boldsymbol{\lambda}^{s-1} + \alpha\check{\boldsymbol{\lambda}}^s \tag{3}$$

for a certain learning learning rate α. To obtain $\check{\boldsymbol{\lambda}}^s$, we define an $N \times M$ matrix $H_{\mathbf{x}}$ that contains the feature functions values $\mathbf{h}$ of every hypothesis, where M is the number of features in Equation 2, and N is the size of n-best($\mathbf{x}$).

$$H_{\mathbf{x}} = [\mathbf{h}(\mathbf{x}, \mathbf{y}_1), \dots, \mathbf{h}(\mathbf{x}, \mathbf{y}_N)]',\ \forall \mathbf{y}_n\ \epsilon\ \mathrm{nbest}(\mathbf{x}_s) \tag{4}$$

Let n-best($\mathbf{x}$) be such a list computed by our models for sentence $\mathbf{x}$. Additionally, let $H^*_{\mathbf{x}}$ be a matrix such that

$$H^*_{\mathbf{x}} = [\mathbf{h}(\mathbf{x}, \mathbf{y}^*), \dots, \mathbf{h}(\mathbf{x}, \mathbf{y}^*)] \tag{5}$$

where all rows are identical and equal to the feature vector of the best hypothesis $\mathbf{y}^*$ within the n-best list. Then, $R_{\mathbf{x}}$ is defined as:

$$R_{\mathbf{x}} = H^*_{\mathbf{x}} - H_{\mathbf{x}} \tag{6}$$

The key idea is to find a vector $\check{\boldsymbol{\lambda}}$ such that differences in scores are reflected as differences in the quality of the hypotheses. That is:

$$R_{\mathbf{x}} \cdot \check{\boldsymbol{\lambda}} \propto \mathbf{l}_{\mathbf{x}} \tag{7}$$

where $\mathbf{l}_{\mathbf{x}}$ is a column vector of N rows such that:

$$\mathbf{l}_{\mathbf{x}} = [l(\mathbf{y}_1), ..., l(\mathbf{y}_n), ..., l(\mathbf{y}_N)]', \forall \mathbf{y}_n\ \in\ \mathrm{nbest}(\mathbf{x}) \tag{8}$$

The objective is to find $\check{\boldsymbol{\lambda}}^s$ such that:

$$\check{\boldsymbol{\lambda}}^s = \underset{\boldsymbol{\lambda}}{\operatorname{argmin}} |\mathbf{R}_{\mathbf{x}} \cdot \boldsymbol{\lambda} - \mathbf{l}_{\mathbf{x}}| = \underset{\boldsymbol{\lambda}}{\operatorname{argmin}} ||\mathbf{R}_{\mathbf{x}} \cdot \boldsymbol{\lambda} - \mathbf{l}_{\mathbf{x}}||^2 \tag{9}$$

where $\|\cdot\|^2$ is the Euclidean norm. Although both optimisations in Equation 9 are equivalent (i.e., the $\check{\boldsymbol{\lambda}}^s$ that minimizes the first one also minimizes the second one), the second optmisation in Equation 9 allows for a direct implementation thanks to the ridge regression. $\check{\boldsymbol{\lambda}}^s$ can be computed as the solution to the overdetermined system $R_{\mathbf{x}} \cdot \check{\boldsymbol{\lambda}}^s = \mathbf{l}_{\mathbf{x}}$, which is given by

$$\check{\boldsymbol{\lambda}}^s = (\mathbf{R}'_{\mathbf{x}} \cdot \mathbf{R}_{\mathbf{x}} + \beta\mathbf{I})^{-1} \cdot \mathbf{l}_{\mathbf{x}} \tag{10}$$

where a small β is used as a regularization term to stabilize $R'_{\mathbf{x}} \cdot R_{\mathbf{x}}$ and to ensure that it is invertible.

Algorithm 2 shows the pseudo-code for obtaining $\check{\boldsymbol{\lambda}}^s$. In this work, we apply the original DRR approach proposed by [17] to an off-line scenario, so that the method proposed is effectively able to compete with state-of-the-art $\boldsymbol{\lambda}$ estimation approaches. In this case, DRR obtains an estimation of $\boldsymbol{\lambda}$ by previously adjusting the $\boldsymbol{\lambda}$ vector to each one of the sentences in a development corpus, i.e., the optimal $\boldsymbol{\lambda}$ is computed after performing a complete epoch on the development set.

for *each of the sentences* $\mathbf{x}_s$ *in* A **do**
  $H_{\mathbf{x}_s} \leftarrow [\mathbf{h}(\mathbf{x}_s, \mathbf{y}_{s,1}), \ldots, \mathbf{h}(\mathbf{x}_s, \mathbf{y}_{s,N})]'$;
  $H^*_{\mathbf{x}_s} \leftarrow [\mathbf{h}(\mathbf{x}_s, \mathbf{y}^*_s), \ldots, \mathbf{h}(\mathbf{x}_s, \mathbf{y}^*_s)]'$;
  $R_{\mathbf{x}_s} \leftarrow H_{\mathbf{x}^*_s} - H_{\mathbf{x}_s}$;
  $\check{\boldsymbol{\lambda}}^s \leftarrow (\mathbf{R}'_{\mathbf{x}_s} \cdot \mathbf{R}_{\mathbf{x}_s} + \beta\mathbf{I})^{-1} \cdot \mathbf{l}_{\mathbf{x}_s}$;
  $\boldsymbol{\lambda}^s \leftarrow (1-\alpha)\boldsymbol{\lambda}^{s-1} + \alpha\check{\boldsymbol{\lambda}}^s$
end

Algorithm 2: Pseudo-code for computing the vector $\boldsymbol{\lambda}^s$ as described in Section 3.1.1

3.2 Batch DRR

The second DRR alternative for an off-line scenario is batch variation. Algorithm 3 shows the difference with the previous algorithm in this case using the batch version.

Data: Development corpus A
Result: $\boldsymbol{\lambda}$
Initialize: $\boldsymbol{\lambda}^0$;
forall *desired number of iterations* I **do**
  forall *desired number of batch* $c \in C$ **do**
    optimization: compute vector $\check{\boldsymbol{\lambda}}^{\boldsymbol{c}}_{\boldsymbol{i}}$;
    estimation: $\boldsymbol{\lambda}^c_i = (1-\alpha)\boldsymbol{\lambda}^{c-1}_i + \alpha\check{\boldsymbol{\lambda}}^{\boldsymbol{c}}_{\boldsymbol{i}}$;
  end
end
selection: output vector $\boldsymbol{\lambda}^{\boldsymbol{C}}_{\boldsymbol{I}}$

Algorithm 3: Pseudo-code for DRR estimating $\boldsymbol{\lambda}$ using a set of batches C, as described in Section 3.2

In this case, we establish a set of batches C, with $A = \bigcup_{k=1}^{|C|} c_k$. The main difference between Sentence-to-sentence DRR and batch DRR is in the **optimization**. In Algorithm 3, the **optimization** step estimates the vector $\check{\boldsymbol{\lambda}}$ described in Section 2 for each one of the development sentences $\mathbf{x}_s$, but in this case this the vector is estimated using all the information in subset c_k. In next section, we present the modification of the optimization step to account for this variation.

3.2.1 Batch-based optimisation in DRR

As exposed in the previous section, DRR obtains $\boldsymbol{\lambda}$ based on obtaining the best vector for each batch c_k. This algorithm is very similar to the one presented in Section 3.1.1, but in this case, the algorithm is presented with all the information within bach c_k instead of only one sentence of the development corpus. The new log-linear vector $\boldsymbol{\lambda}^k$, the previously learned $\boldsymbol{\lambda}^{k-1}$ needs to be combined with an appropriate update step $\check{\boldsymbol{\lambda}}^k$. The $\boldsymbol{\lambda}^k$ is calculated as a linear combination:

$$\boldsymbol{\lambda}^k = (1-\alpha)\boldsymbol{\lambda}^{k-1} + \alpha\check{\boldsymbol{\lambda}}^k \tag{11}$$

To obtain $\check{\boldsymbol{\lambda}}^k$, we changed the Equation 10:

$$\check{\boldsymbol{\lambda}}^k = (\mathbf{R}'_{c_k} \cdot \mathbf{R}_{c_k} + \beta\mathbf{I})^{-1} \cdot \mathbf{l}_{c_k} \tag{12}$$

where R_{c_k} is defined as: $R_{c_k} = H^*_{c_k} - H_{c_k}$. The matrix H_{c_k} $N \cdot |c_k| \times M$ contains the feature functions $\mathbf{h}$ of every hypothesis for all the sentence $\mathbf{x} \in c_k$, where M is the number of features in Equation 2, and N is the size of n-best($\cdot$) for each sentence $\mathbf{x} \in c_k$:

$$H_{c_k} = [H_{\mathbf{x}_1}, \ldots, H_{\mathbf{x}_{|c_k|}}]' \tag{13}$$

Additionally, let H^*_M be a matrix such that

$$H^*_{c_k} = [H^*_{\mathbf{x}_1}, \ldots, H^*_{|c_k|}] \tag{14}$$

and $\mathbf{l}_{c_k}$ is a column vector of N rows such that:

$$\mathbf{l}_M = [l_{\mathbf{x}_1}, \ldots, l_{\mathbf{x}_{|c_k|}}]' \tag{15}$$

for *each of the batch* c **do**
  $H_{c_k} \leftarrow [H_{\mathbf{x}_1}, \ldots, H_{\mathbf{x}_{|c_k|}}]'$;
  $H^*_{c_k} \leftarrow [H_{\mathbf{x}_1}, \ldots, H^*_{\mathbf{x}_{|c_k|}}]'$;
  $R_{c_k} \leftarrow H^*_{c_k} - H_{M_c}$;
  $\check{\boldsymbol{\lambda}}^c \leftarrow (\mathbf{R}'_{c_k} \cdot \mathbf{R}_{c_k} + \beta\mathbf{I})^{-1} \cdot \mathbf{l}_{c_k}$;
  $\boldsymbol{\lambda}^c \leftarrow (1-\alpha)\boldsymbol{\lambda}^{c-1} + \alpha\check{\boldsymbol{\lambda}}^c$
end

Algorithm 4: Pseudo-code for computing the vector $\boldsymbol{\lambda}^c$ as described in Section 3.2.1

4 Experiments and Dicussion

In this section, we describe the experimental framework employed to assess the performance of the DRR variants described in Section 3. We will first detail the experimental setup employed, and then we will report the analysis of our method and their results. Finally, we will show a comparative of our DRR method with two state-of-the-art optimisation methods: MERT and MIRA.

4.1 Corpora

We conducted experiments on two different language pairs: English $\rightarrow$ French (EN-FR) and German $\rightarrow$ English (DE-EN). Given that the techniques descried above are suited for adaptation purposes, we researched the performance of these techniques in a cross-domain setting: we conducted experiments training the SMT system initially on an out-of-domain corpus, and then analyzing its performance on an in-domain corpus. As out-of-domain corpus we employed the European Parliament corpus (Europarl) [12]. As in-domain corpus, we experimented with three different corpora:

Table 1: Out-of-domain corpus main figures, in terms of number of sentences ($|S|$), number of words ($|W|$), vocabulary size ($|V|$) and average sentence length ($\overline{|W|}$). Two different figures are given in each column to account for the two different languages.

	EN-FR				DE-EN			
Corpus	$\|S\|$	$\|W\|$	$\|V\|$	$\overline{\|W\|}$	$\|S\|$	$\|W\|$	$\|V\|$	$\overline{\|W\|}$
Europarl	2.0M	50.2M - 52.5M	157.7k - 215.2k	20.5 - 22.5	1.9M	44.6M - 47.8M	290.8k - 153.4k	20.5 - 21.5
Euro-Dev	2000	40.8k - 48.6k	5.1k - 6.2k	20.4 - 22.3	2000	46.4k - 49.8k	10.9k - 8.6k	23.2 - 24.9

– Medical domain [29] (henceforth referred to as EMEA). The partitions employed in this domain were established in the 2014 Workshop on Statistical Machine Translation (WMT) [2] of the Association for Computational Linguistics.
– News-comentary domain [30] (henceforth referred to as NEWS). The corpus is composed of translations of news articles. The partitions employed in this domain were established in the WMT of the Association for Computational Linguistic.
– Xerox printer manuals [1] (henceforth referred to as XRCE). This corpus consists of translations of Xerox printer manuals.

Statistics of the in-domain corpora are provided in Table 2.

4.2 Experimental setup

Experiments were performed by means of the open-source MT toolkit Moses [14], where the included features are five translation models, seven re-ordering models, the word-penalty and a word-based language model,i.e., $|\boldsymbol{\lambda}| = 14$. The language model used was a 5-gram with modified Kneser-Ney smoothing [11], built with the SRILM toolkit [28]. The translation quality would ideally be measured by humans. However, this is a very expensive resource, not commonly available in research tasks. Hence, the SMT research community developed some automatic metrics to measure translation quality. In this work we evaluated the translation systems with three different automatic metrics: BLEU, TER and METEOR.

– BLEU (BiLingual Evaluation Understudy) [22] is a precision metric (i.e., the higher the better) that measures n-ngram precision of the system hypothesis with respect to the reference, with a penalty for sentences that are too short.
– TER (TranslationError Rate) [25] is an error metric (i.e., the lower the better) that computes the minimum number of edits required to modify the system hypotheses so that they match the reference. Possible edits include insertion, deletion and substitution of single words, as well as shifts of word sequences.
– METEOR [15] is a precision metric (i.e., the higher the better). This metric is based on word alignments, which can be either exact, stem, synonym, or paraphrase matches, both between words and phrases. Segment and system level metric scores are calculated based on the alignments between hypothesis-reference pairs.

As described in Section 3, the score function σ is correlated with some quality criterion, which is measured by some automatic metric like BLEU or TER. We will analyze the behaviour of DRR both using BLEU and TER. We favour the use of BLEU because of its wider acceptance in the SMT community. However, the original implementation of BLEU is not always well defined at the sentence level, given that it implements a geometric average which is zero whenever there is no common 4-gram between the hypothesis and the reference, e.g. 3-word sentence. For this reason, we used smoothed BLEU, as defined by [5]. In case of the TER metric, the original work by [17] applied on-line DRR to optimise TER scores, which is why we decided analyze it in this new scenario. In the case of METEOR, we did not conduct experiments with this metric because MERT and MIRA are not suited for optimizing METEOR, and no appropriate baseline would be available for comparison.

For each corpus, we trained baseline systems for comparison. This baseline was obtained by tuning the SMT system using an out-of-domain development corpus: the Euro-DEV (Details in Table 1). We named this system $bsln$. In addition, the points in the plots presented in this paper display the average of 10 repetitions of each experiment. The scale of the y-axis will be linear whenever the plot displays translation quality, and logarithmic in the case of the confidence interval sizes. These confidence intervals present the 95% confidence level and were computed as 2σ, with σ in this case being the empirical standard deviation observed in the 10 repetitions. Note that the full confidence interval would be 4σ, i.e., $\pm 2\sigma$. Confidence intervals are displayed in different plots, instead of using error bars, because otherwise, the translation quality plots would present vertical lines across the complete plot, rendering it unreadable.

Table 2: In-domain corpora main figures.

		EN-FR				DE-EN			
Domain	Corpus	$\|S\|$	$\|W\|$	$\|V\|$	$\overline{\|W\|}$	$\|S\|$	$\|W\|$	$\|V\|$	$\overline{\|W\|}$
EMEA	DEV	484	9.8k - 11.6k	0.9k - 1.0k	20.3 - 24.1	500	9.8k - 10.2k	0.8k - 0.9k	19.7 - 20.6
	TEST	1000	21.4k - 26.9k	4.1k - 4.4k	21.4 - 26.9	1000	20.6k - 21.3k	5.5k - 4.1k	20.6 - 21.3
NEWS	DEV	1500	24.1k - 24.9k	4.0k - 4.8k	15.7 - 16.8	1777	34.4k - 36.5k	3.4k - 3.7k	19.3 - 20.6
	TEST	1395	21.9k - 24.1k	4.4k - 4.9k	15.7 - 17.2	2767	51.5k - 53.6k	11.0k - 8.0k	18.6 - 19.3
XRCE	DEV	976	11.9k - 13.3k	1.2k - 1.4k	12.2 - 13.6	931	10.6k - 10.9k	1.4k - 1.1k	11.4 - 11.5
	TEST	936	11.4k - 12.3k	1.1k - 1.2k	12.2 - 13.2	980	12.6k - 12.8k	1.8k - 1.4k	12.9 - 13.1

4.3 DRR experiments

In this section, we present a study of our DRR variations, where the following issues were researched:

1. Varying learning rate and N-best size considered.
2. Difference between sentence-by-sentence DRR and batch DRR.

In this study, we used only the development corpus of each domain since the purpose is to analyze the effect of adapting $\boldsymbol{\lambda}$. Accordingly, the results displayed here are using the corresponding corpus for each domain (EMEA-DEV, NC-DEV and XRCE-DEV, details in Section 4.1). In addition, the translation quality was measured with the BLEU metric, in all the experiments in this section.

4.3.1 Varying learning rate and increasing the N-best size

As a first step, we analyzed the effect of varying the learning rate α described in Equation 3, together with different N-best sizes. Also, we used the sentence-by-sentence DRR variation for these experiments.

Results are shown in Figure 1 for English-French, and for each domain considered. We analyzed a broad range of learning rates α, but we show here the most significant α values for clarity purposes. In addition, the translation quality obtained with the baseline system is also displayed. Several conclusion can be drawn:

- The results show that high values of α lead to a significant degradation in translation quality. The reason for this can be explained by looking at Equation 3. High values produce bigger changes of the $\boldsymbol{\lambda}^s$ respect to $\boldsymbol{\lambda}^{s-1}$, and consequently an important change in the search space. On the other hand, smaller values of λ can obtain better translation quality.
- The effect of increasing the size of N-best considered was also analysed. As it can be seen, the size of N-best and α are strongly related parameters, and high α values need more N-best for obtaining better results. But when the algorithm has the adequate learning rate, the N-best size does not have a large influence on the outcome in terms of translation quality.

4.3.2 Varying batch size

In this section, we compare the batch DRR (Section 3.2) and sentence-by-sentence DRR (Section 3.1). For this comparison, we conducted experiments similar to those in previous Section 4.3.1 (varying α and N-best size), but we included the use of batches. The best results for each domain are presented in Table 3. The results are shown considering BLEU as evaluation metric, and for varying number of batches $|C|$. The following table illustrates other important parameters to consider. These are: $|c_k|$, which represents the number of sentences in each batch c_k and α, the learning rate used. Then, the first line in the table represents the best result obtained with sentence-by-sentence DRR for each domain. As for the effect on the batch DRR variation, as compared to sentence-by-sentence DRR, we can observe similar results in terms of BLEU. Hence, we can conclude that both alternatives converge to a similar search space. The most remarkable difference is the learning rate value used to obtain the best results. In the case of sentence-by-sentence DRR, the best results are obtained with a very small value of α. In other hand, batch DRR obtained the same results with higher values of α. We think this happens because batch DRR includes more information in each update, and hence the update steps are more stable.

4.4 Comparison between DRR, MERT and MIRA

Once the effect of the different parameters of DRR was analysed, we pursue to compare our method with standard methods such as MERT and MIRA. This comparative study was conducted taking into account the following issues:

1. Varying the size of the development corpus (Section 4.4.1).
2. Increasing the number of n-best used within eache method (Section 4.4.2).

4.4.1 Development Size

As a first step in this comparative, we study the effect of increasing the number of development samples made available to the system. Figure 2 and Figure 3 show the effect of

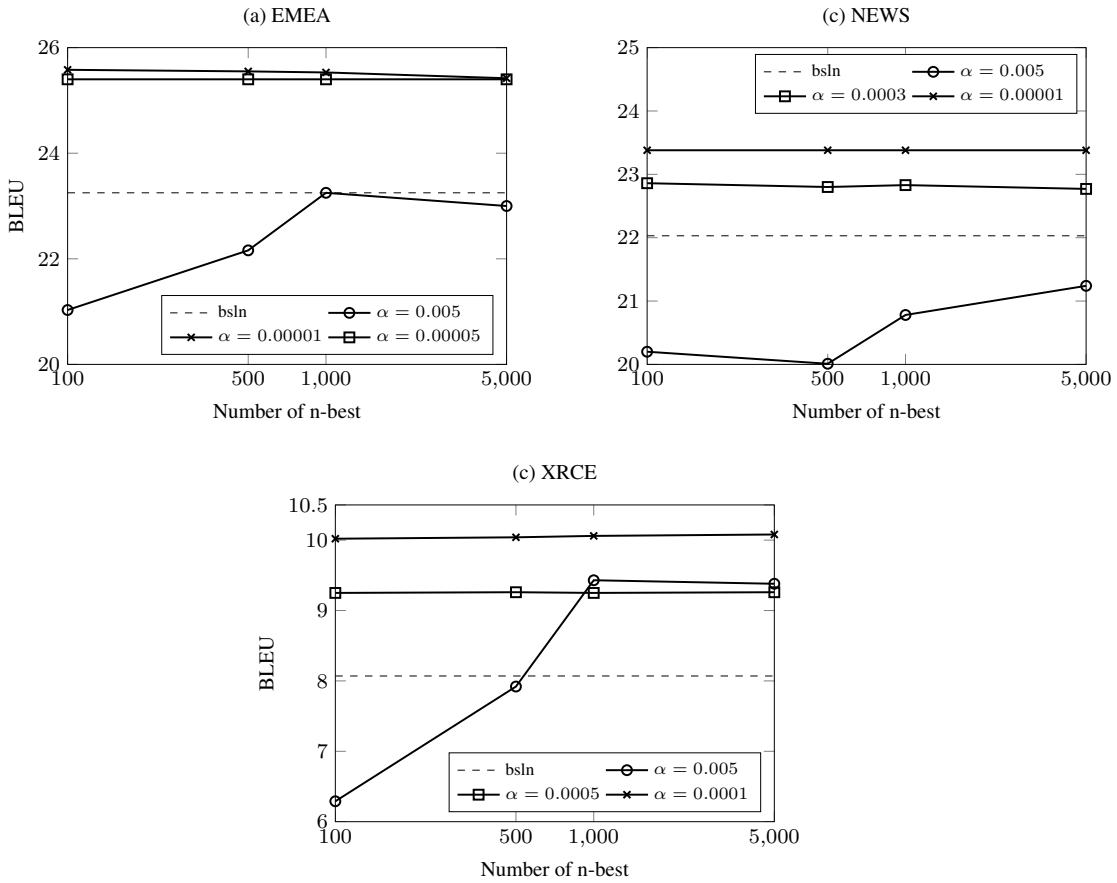

Fig. 1: Translation quality comparison for different α values and number of n-best.

Table 3: Best results varying the batch size for each domain, evaluation made with BLEU.

	EN-FR				DE-EN			
Domain	C	$\|M\|$	α	BLEU	C	$\|M\|$	α	BLEU
EMEA	-	-	0.00001	25.6	-	-	0.0005	19.6
	3	240	0.01	25.5	3	240	0.01	19.4
	4	160	0.005	25.5	4	160	0.01	19.4
	7	80	0.005	25.5	7	80	0.001	19.4
NEWS	-	-	0.00001	23.3	-	-	0,00005	17.6
	4	400	0.005	23.4	4	400	0.01	17.5
	7	200	0.001	23.5	7	200	0.001	17.6
	28	50	0.001	23.5	28	50	0.0001	17.4
XRCE	-	-	0.0001	10.1	-	-	0.00005	10.3
	4	300	0.005	10.1	4	300	0.01	10.3
	7	150	0.0005	10.1	7	150	0.001	10.3
	20	50	0.0005	10.1	20	50	0.001	10.4

adding sentences to development corpus and the confidence intervals derived.

These results show translation quality in terms of BLEU and TER (Figure 2 for BLEU, Figure 3 for TER), for each domain and development corpus considered (details in Section 4.1). For clarity, we only show results for the best meta-parameter configuration of DRR for each domain, obtained in previous Section 4.3, and the standard parameter configuration of MERT and MIRA present in Moses toolkit.

Results of such comparison can be seen in Figure 2, in temrs of BLEU. There are several things that should be noted:

– Results obtained for all methods are better than the baseline system (bsln) at the beginning, as could be expected.

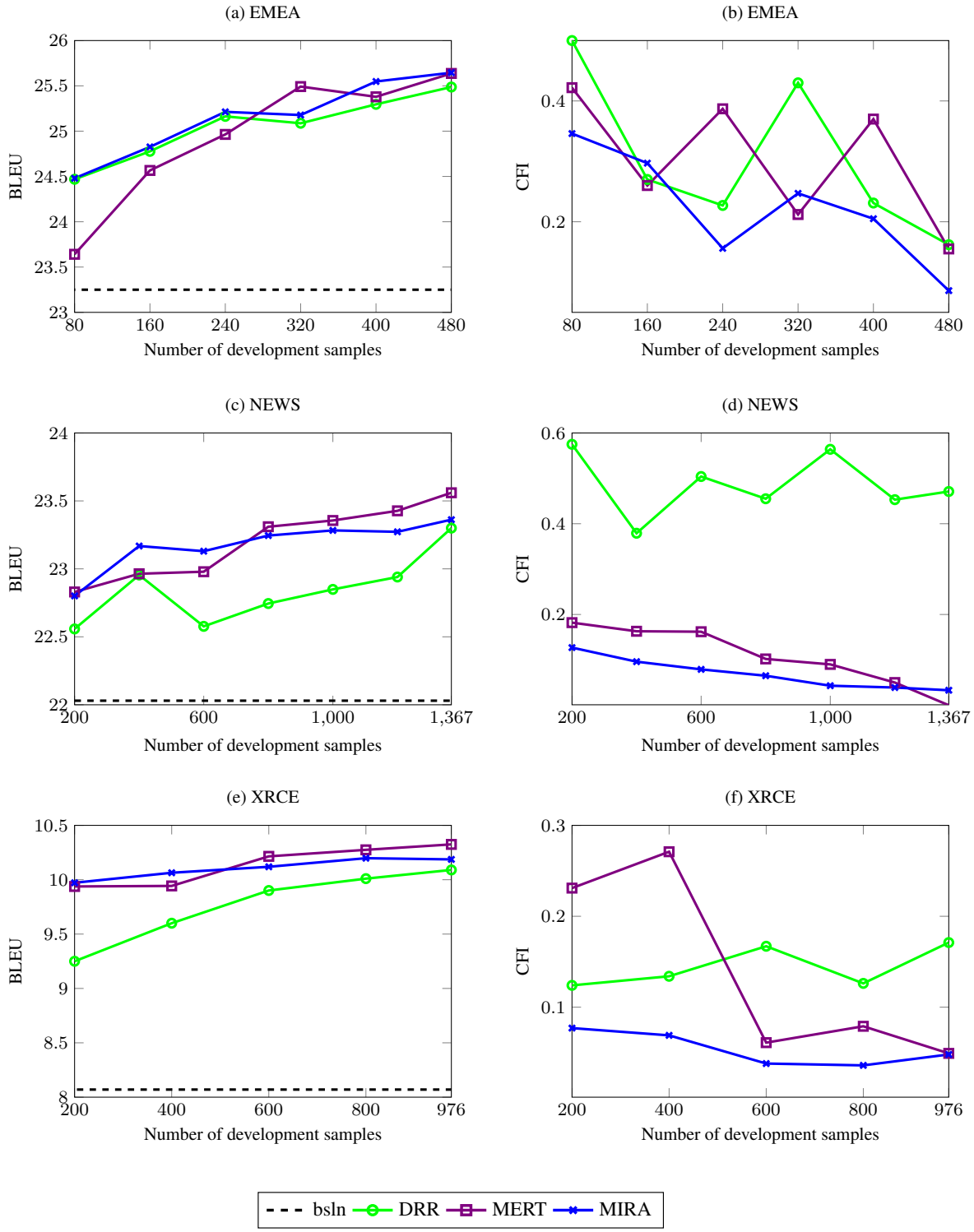

Fig. 2: Performance comparison across the different corpora analysed and with the different λ estimation methods. The three plots on the left display BLEU, while the three plots on the right display the size of the confidence intervals (CFI).

This could be demonstrating the effectiveness of these methods in an SMT adaptation task.

– All the methods have the same behaviour when the development size increases, leading to improvements in

translation quality as measured by BLEU, (around $1-2$ points better).
- Results obtained with our DRR method are similar than the ones obtained with MIRA and MERT.
- As expected, smaller amounts of development corpus lead to larger confidence intervals.
- The NEWS corpus appears to be a specially difficult corpus for the DRR method. Also, confidence intervals are especially high when compared to the other methods.
- Lastly, when looking at the translation quality of the XRCE corpus, it stands out that curves behave especially poorly in terms of BLUE (around 10 points). The development set of the XRCE corpus seems to be quite different from the training data (Europarl training), which implies that the system is not able to obtain a good N-best list.

Since TER is another evaluation metric commonly used in the SMT community and was the metric used initially for on-line DRR, we also analyzed the behaviour when using TER. The overall analysis is very similar concerning the one involving BLEU (λ estimation method behaviour when increasing the development size). Figure 3 shows the main results obtained for the three domains and languages pair (EN-FR).

- Using TER for estimating λ leads to similar results as compared to using BLEU.
- Increasing the number of adaptation samples leads to better results for all the methods considered, without being statistically significant in terms of BLEU.
- Similarly to previous experiments, DRR obtains weak reuslts in the NEWS corpus. Nevertheless, the difference is not significant.

4.4.2 Variating N-best size

As explained before, all methods described leverage an N-best list for optimizing the $\boldsymbol{\lambda}$ of the log-lineal model. In case of MERT, the N-best list is used to obtain the best hypotheses. In the case of DRR, the N-best is of greater importance to the algorithm, since it uses all the information contained in the list. For this reason, in this section we analyze the difference of N-best sizes. We studied the following N-best sizes; $|N-best| = \{100, 200, 500, 1000, 1500, 5000\}$, although not reported here in order to avoid including too many plots in this paper. For this reason, we only report the best result obtained regarding the N-best size.

Table 4 and Table 5 show the best results in terms of BLEU, TER and METEOR. There are several things that should be noted:

- In terms of the metric used in Table 4 and Table 5 (BLEU or TER respectively) we can see that all the methods obtained very similar results without statistical significance. These results are consistent across different domains and language pairs.
- The DRR method needs more N-best to obtain better results, as compared to the other two methods. This should be expected, since MERT and MIRA conduct several translation steps, whereas DRR only conducts one single step. Hence, for a fixed n-best size, MERT and MIRA have access to more information than DRR, since they re-compute the search space after each translation step. For this reason, DRR was actually designed to have access to a large N-best list.
- In addition, we can see different results when using BLEU or TER, leading to differences in the range of $1-3$ BLEU points and $2-3$ TER points.

4.4.3 Results analysis for test corpus and computational time

Given that tuning is critical for adapting a SMT system to a specific domain or corpus, in this section we presented the results obtained for each method when translating different domains (EMEA-TEST, NEWS-TEST and XRCE-TEST). These corpora were only used for optimizing the log-linear weights, so the only information that the system has at this moment was obtained during the tuning process using the development corpus, which belongs to the same domain as the test corpus.

The vector weights $\boldsymbol{\lambda}$ used in each method were obtained using the best configuration analysed in the previous sections. In addition, we saw in the previous sections (Section 4.4.1 and Section 4.4.2) that the methods analyzed have very similar behaviour in terms of BLEU or TER. For this reason, the result using TER was removed from the final comparison in order to avoid clogging the paper with too many similar tables.

In Table 6, we show the mail results obtained for each method (MERT, MIRA and DRR) in terms of the three metrics studied (BLEU, TER and METEOR). As shown, our method is able to yield slightly better results in most domains (EMEA DE-EN, XRCE EN-FR and DE-EN), although differences are not statistical significant. In the other cases, our DRR method leads to competitive results with respect to the state of the art. We understand that is important, since it proves the competitiveness of our proposal in this task, with respect to other techniques which have been largely studied by the SMT community and are considered state of the art. Regarding computational time, Table 7 reports the time consumed by each one of the approaches reported in Table 6. Computational time was measured in single-threaded runs of the algorithms presented. As shown, the DRR method is faster than the other two methods, while still obtaining better or similar results.

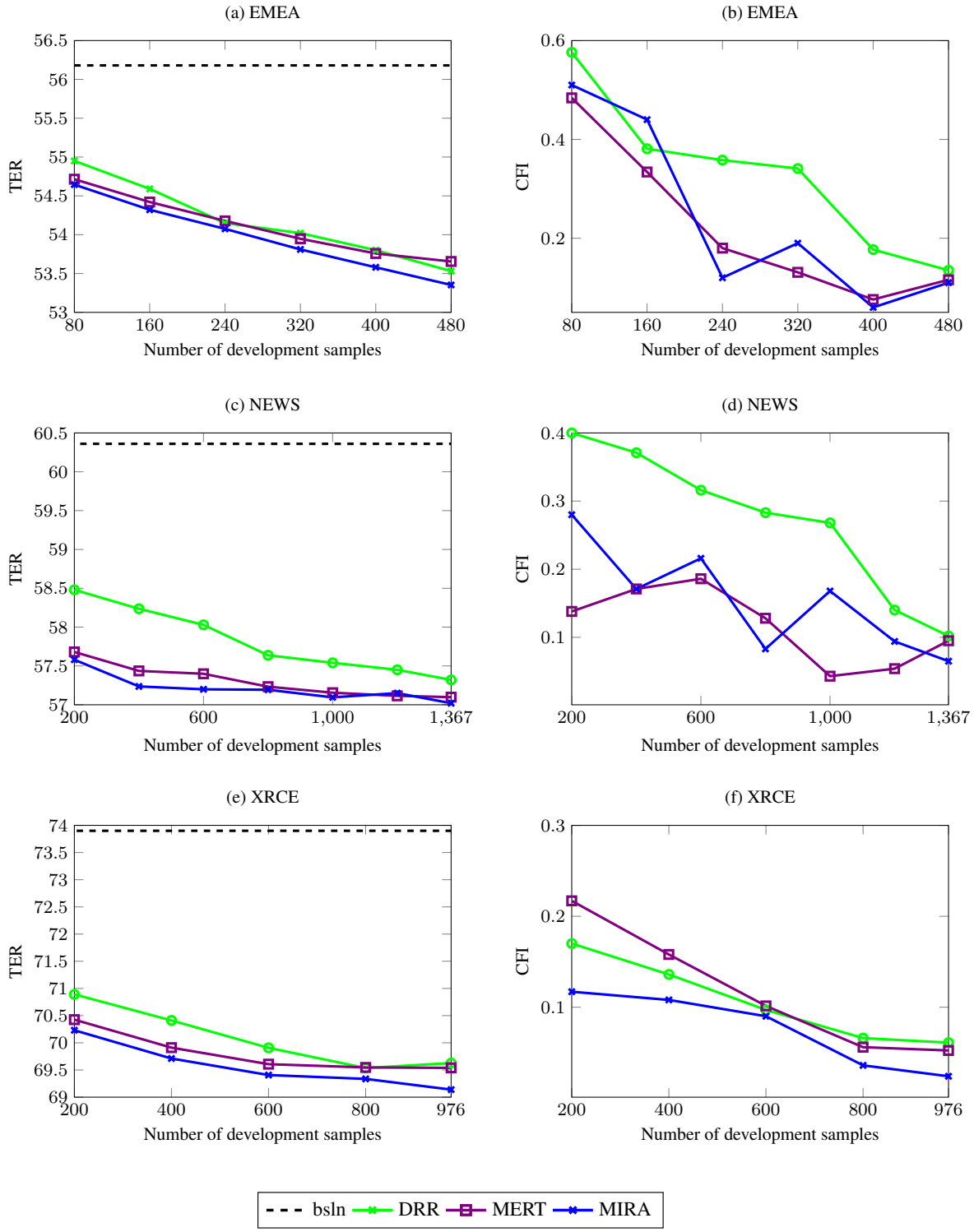

Fig. 3: Performance comparison across the different corpora analysed and with the different λ estimation methods. The three plots on the left display TER, while the three plots on the right display the size of the confidence intervals (CFI).

5 Conclusions

In this paper, the DRR method has been thoroughly analysed for its application to log-linear vector weight adaptation in SMT. Experimental results analysing the effectiveness of such adaptation method have been reported. Results show that DRR is able to provide consistent improvements

Table 4: Effect to change the n-best size for each method, evaluation made with BLEU.

		EN-FR				DE-EN			
Domain	Method	$\|N-best\|$	BLEU	TER	METEOR	$\|N-best\|$	BLEU	TER	METEOR
EMEA	MERT	200	25.7 ± 0.1	56.1 ± 0.5	48.6 ± 0.2	500	19.8 ± 0.1	60.6 ± 0.4	26.6 ± 0.1
	MIRA	200	25.6 ± 0.1	55.8 ± 0.2	48.6 ± 0.1	100	19.7 ± 0.1	60.6 ± 0.2	26.6 ± 0.3
	DRR	100	25.5 ± 0.2	55.8 ± 0.3	48.5 ± 0.1	200	19.4 ± 0.3	60.0 ± 0.4	26.6 ± 0.1
NEWS	MERT	100	23.5 ± 0.1	59.3 ± 0.002	47.5 ± 0.1	200	18.1 ± 0.04	65.8 ± 0.1	27.4 ± 0.03
	MIRA	200	23.4 ± 0.03	59.2 ± 0.8	47.5 ± 0.04	200	17.9 ± 0.02	65.3 ± 0.1	27.3 ± 0.01
	DRR	500	23.2 ± 0.5	58.5 ± 0.7	46.1 ± 0.7	500	17.5 ± 0.2	66.7 ± 1.1	27.3 ± 0.1
XRCE	MERT	200	10.3 ± 0.04	74.2 ± 0.6	34.2 ± 0.3	500	11.3 ± 0.04	71.9 ± 1.4	17.6 ± 0.2
	MIRA	200	10.2 ± 0.04	74.0 ± 0.2	34.0 ± 0.1	200	10.8 ± 0.2	72.1 ± 1.3	17.5 ± 0.02
	DRR	500	10.1 ± 0.1	73.7 ± 0.2	33.7 ± 0.1	500	10.4 ± 0.2	69.6 ± 0.3	17.4 ± 0.1

Table 5: Effect to change the n-best size for each method, evaluation made with TER.

		EN-FR				DE-EN			
Domain	Method	$\|N-best\|$	TER	BLEU	METEOR	$\|N-best\|$	TER	BLEU	METEOR
EMEA	MERT	200	53.7 ± 0.1	24.4 ± 0.2	47.2 ± 0.3	200	57.5 ± 0.1	17.4 ± 0.3	25.4 ± 0.1
	MIRA	200	53.4 ± 0.1	24.7 ± 0.2	47.2 ± 0.1	200	57.3 ± 0.1	17.6 ± 0.2	25.5 ± 0.2
	DRR	500	53.6 ± 0.2	24.7 ± 0.3	47.1 ± 0.1	500	57.6 ± 0.3	17.9 ± 0.4	25.4 ± 0.1
NEWS	MERT	200	57.1 ± 0.1	21.5 ± 0.2	44.8 ± 0.2	100	62.3 ± 0.01	16.0 ± 0.2	26.2 ± 0.1
	MIRA	200	57.1 ± 0.04	21.3 ± 0.4	44.8 ± 0.1	200	62.6 ± 0.02	16.3 ± 0.1	26.1 ± 0.1
	DRR	1000	57.5 ± 0.4	20.7 ± 0.7	44.0 ± 0.6	500	63.0 ± 0.2	15.8 ± 0.9	26.1 ± 0.3
XRCE	MERT	1000	69.4 ± 0.1	8.7 ± 0.1	30.8 ± 0.2	200	66.8 ± 0.1	9.1 ± 0.4	16.4 ± 0.1
	MIRA	500	69.1 ± 0.1	8.9 ± 0.1	31.0 ± 0.2	200	67.2 ± 0.1	8.8 ± 0.4	16.3 ± 0.1
	DRR	500	69.5 ± 0.2	8.8 ± 0.1	30.7 ± 0.2	200	67.1 ± 0.1	9.0 ± 0.4	16.2 ± 0.1

Table 6: Translation results for the three domains using different optimization algorithm using BLEU metric.

		EN-FR			DE-EN		
Domain	Method	BLEU	TER	METEOR	BLEU	TER	METEOR
EMEA	MERT	22.3 ± 0.2	58.8 ± 0.1	45.0 ± 0.2	19.1 ± 0.2	61.7 ± 0.5	27.1 ± 0.1
	MIRA	$\mathbf{22.4 \pm 0.1}$	$\mathbf{58.5 \pm 0.2}$	$\mathbf{45.1 \pm 0.1}$	19.1 ± 0.1	61.4 ± 0.2	27.2 ± 0.02
	DRR	22.3 ± 0.2	58.5 ± 0.2	45.0 ± 0.2	$\mathbf{19.2 \pm 0.2}$	$\mathbf{60.4 \pm 0.5}$	$\mathbf{27.2 \pm 0.1}$
NEWS	MERT	27.0 ± 0.01	53.6 ± 0.1	50.8 ± 0.2	21.6 ± 0.1	58.5 ± 0.1	30.4 ± 0.04
	MIRA	$\mathbf{27.4 \pm 0.1}$	$\mathbf{53.0 \pm 0.04}$	$\mathbf{51.1 \pm 0.04}$	$\mathbf{21.7 \pm 0.02}$	$\mathbf{57.9 \pm 0.03}$	$\mathbf{30.5 \pm 0.02}$
	DRR	26.8 ± 0.3	54.3 ± 0.9	51.0 ± 0.2	21.1 ± 0.3	58.6 ± 1.0	30.2 ± 0.1
XRCE	MERT	9.9 ± 0.3	73.6 ± 0.8	36.8 ± 0.3	9.4 ± 0.3	73.9 ± 0.9	18.1 ± 0.2
	MIRA	10.2 ± 0.1	73.3 ± 0.2	36.1 ± 0.1	9.8 ± 0.1	72.5 ± 1.0	18.1 ± 0.03
	DRR	$\mathbf{10.2 \pm 0.1}$	$\mathbf{73.0 \pm 0.2}$	$\mathbf{36.7 \pm 0.1}$	$\mathbf{9.6 \pm 0.1}$	$\mathbf{70.1 \pm 0.4}$	$\mathbf{18.0 \pm 0.1}$

in translation quality over the baseline systems, as measured by BLEU or TER. In addition, we have demonstrated, via empirical experiments, that our DRR method obtains comparable in some cases better result than MERT and MIRA, with a reduction of computational time, across different domain and language pairs. We consider this important, since it means that DRR is able to lead to competitive results, while using less computational resources.

As future work, we plan to extend the current DRR implementation so that it is able to deal with more feature-rich SMT models and we will carry out new experiments with large amounts of corpus and language diversity.

References

1. Barrachina, S., Bender, O., Casacuberta, F., Civera, J., Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Tomás, J., Vidal, E., et al.: Statistical approaches to computer-assisted translation. Computational Linguistics **35**(1), 3–28 (2009)
2. Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., Specia, L. (eds.): Proceedings of the Ninth Workshop on Statistical Machine Translation. Association for Computational Linguistics (2014)

Table 7: Time consumed by the different approaches compared. Results are given in minutes.

		EN-FR	DE-EN
Domain	Method	Computational Time	Computational Time
EMEA	MERT	252	240
	MIRA	300	350
	DRR	**190**	**200**
NC	MERT	480	900
	MIRA	660	840
	DRR	**432**	**720**
XRCE	MERT	400	390
	MIRA	420	390
	DRR	**360**	**350**

3. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational linguistics **19**, 263–311 (1993)
4. Callison-Burch, C., Koehn, P., Monz, C., Peterson, K., Przybocki, M., Zaidan, O.F.: Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 17–53 (2010)
5. Chen, B., Cherry, C.: A systematic comparison of smoothing techniques for sentence-level bleu. In: Proceedings of the Workshop on Statistical Machine Translation, pp. 362–367 (2014)
6. Cherry, C., Foster, G.: Batch tuning strategies for statistical machine translation. In: Proceedings of the North American Chapter of the Association for Computational Linguistics, pp. 427–436 (2012)
7. Clark, J.H., Dyer, C., Lavie, A., Smith, N.A.: Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 176–181 (2011)
8. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. The Journal of Machine Learning Research **7**, 551–585 (2006)
9. Hasler, E., Haddow, B., Koehn, P.: Margin infused relaxed algorithm for moses. The Prague Bulletin of Mathematical Linguistics **96**, 69–78 (2011)
10. Hopkins, M., May, J.: Tuning as ranking. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1352–1362 (2011)
11. Kneser, R., Ney, H.: Improved backing-off for m-gram language modeling. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 181–184 (1995)
12. Koehn, P.: Europarl: A parallel corpus for statistical machine translation. In: Proceedings of the Machine Translation Summit, pp. 79–86 (2005)
13. Koehn, P.: Statistical machine translation. Cambridge University Press (2010)
14. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., Herbst, E.: Moses: open source toolkit for statistical machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 177–180 (2007)
15. Lavie, M.D.A.: Meteor universal: Language specific translation evaluation for any target language. In: Proccedings of the Annual Meeting of the Association for Computational Linguistics, pp. 376–387 (2014)
16. Marie, B., Max, A.: Multi-pass decoding with complex feature guidance for statistical machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 554–559 (2015)
17. Martínez-Gómez, P., Sanchis-Trilles, G., Casacuberta, F.: Online adaptation strategies for statistical machine translation in post-editing scenarios. Pattern Recognition **45**(9), 3193–3203 (2012)
18. Nakov, P., Vogel, S.: Robust tuning datasets for statistical machine translation (2017). *arXiv:1710.00346*
19. Neubig, G., Watanabe, T.: Optimization for statistical machine translation: A survey. Computational Linguistics **42**(1), 1–54 (2016)
20. Och, F.J.: Minimum error rate training in statistical machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 160–167 (2003)
21. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational linguistics **29**, 19–51 (2003)
22. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 311–318 (2002)
23. Sanchis-Trilles, G., Casacuberta, F.: Log-linear weight optimisation via bayesian adaptation in statistical machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics, pp. 1077–1085 (2010)
24. Sanchis-Trilles, G., Casacuberta, F.: Improving translation quality stability using bayesian predictive adaptation. Computer Speech & Language **34**(1), 1–17 (2015)
25. Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J.: A study of translation edit rate with targeted human annotation. In: Proccedings of the Annual Meeting of the Association for Machine Translation in the Americas, pp. 223–231 (2006)
26. Sokolov, A., Yvon, F.: Minimum error rate training semiring. In: Proceedings of the Annual Conference of the European Association for Machine Translation, pp. 241–248 (2011)
27. Stauffer, C., Grimson, W.E.L.: Learning patterns of activity using real-time tracking. Pattern Analysis and Machine Intelligence **22**(8), 747–757 (2000)
28. Stolcke, A.: Srilm-an extensible language modeling toolkit. In: Proccedings of the International Conference on Spoken Language Processing, pp. 901–904 (2002)
29. Tiedemann, J.: News from opus - a collection of multilingual parallel corpora with tools and interfaces. In: Proceedings of the Recent Advances in Natural Language Processing, pp. 237–248 (2009)
30. Tiedemann, J.: Parallel data, tools and interfaces in opus. In: Proceedings of the Language Resources and Evaluation Conference, pp. 2214–2218 (2012)