A comparison of discriminative training criteria for continuous space translation models

Alexandre Allauzen, Quoc Khanh Do, François Yvon

A Comparison of Discrimitative Training Criteria for Continuous Space Translation Models*

Alexandre Allauzen Quoc Khanh Do
François Yvon

LIMSI, CNRS, Univ. Paris-Sud, Université Paris Saclay

first.last@limsi.fr

Abstract

This paper explores a new discriminative training procedure for continuous-space translation models (CTMs) which correlates better with translation quality than conventional training methods. The core of the method lays in the definition of a novel objective function which enables us to effectively integrate the CTM with the rest of the translation system through N-best rescoring. Using a fixed architecture, where we iteratively retrain the CTM parameters and the log-linear coefficients, we compare various ways to define and combine training criteria for each of these steps, drawing inspirations both from max-margin and learning-to-rank techniques. We experimentally show that a recently introduced loss function, which combines these two techniques, outperforms several objective functions from the literature. We also show that ensuring the consistency of the losses used to train these two sets of parameters is beneficial to the overall performance.

1 Introduction

Over the past years, research on neural networks (NN) architectures for Natural Language Processing has been rejuvenated. Boosted by early successes in language modelling for speech recognition (Schwenk, 2007; Le et al, 2011), NNs

*Preprint of a paper published as: Alexandre Allauzen, Quoc Khan Do and François Yvon (2017). "A Comparison of Discrimitative Training Criteria for Continuous Space Translation Models". *Machine Translation* (31:1).19-33 https://doi.org/10.1007/s10590-017-9195-1.

have since been successufully applied to many other tasks (Socher et al, 2013; Yang et al, 2013). In particular, these techniques have been applied to Statistical Machine Translation (SMT), first to estimate continuous-space translation models (CTMs) (Schwenk et al, 2007; Le et al, 2012; Devlin et al, 2014), more recently to implement neural end-to-end translation systems (Cho et al, 2014; Sutskever et al, 2014).

In phase-based SMT settings, CTMs are typically trained by maximizing the regularized log-likelihood on some parallel training corpora, then used as an additional feature in the conventional log-linear model (Och, 2003). Computing the log-likelihood however requires the costly normalization of scores on the output layer, and several alternative training objectives have been proposed to speed up training and inference, such as the Noise Contrastive Estimation (NCE) (Gutmann and Hyvärinen, 2010). In any case, NN training is usually performed (a) in isolation from the other components of the SMT system and (b) using a criterion that is unrelated to the actual performance of the SMT system (as measured for instance by automatic metrics such as BLEU). Therefore, the resulting NN weights may be under-optimal *wrt* their intended use.

In this paper, we study a variety of alternative training regimes aimed at addressing problems (a) and (b). Using a fixed architecture, where we iteratively retrain the NN parameters and the log-linear coefficients in a rescoring setting, we compare various ways to define and combine training criteria for each of these step, drawing inspirations both from max-margin (Watanabe et al, 2007; Chiang et al, 2008; Cherry and Foster, 2012) and learning-to-rank techniques (Hopkins and May, 2011; Simianer et al, 2012). Our experiments show that our newly introduced loss, which combines these two techniques, outperforms several widely used objective functions from the literature; ensuring the consistency of the losses used to train these two sets of parameters furthermore also significantly improves our performance. Overall, we were able to report results that surpass a conventional phrase-based system by more than 2.5 BLEU points. This work thus extends (Do et al, 2015b) by providing a thorough comparison of a much wider array of training criteria expressed here in a generic framework.

Our starting point is a non-normalized extension of the n-gram CTM (Le et al, 2012) briefly revisited in section 2. We then introduce several objective functions and the associated optimization procedures in section 3. Our proposals are evaluated in an N-best rescoring step, using the framework of n-gram-based systems, within which they integrate seamlessly[1] (see section 4). We conclude (section 6) by summing up our main findings and discussing future prospects.

[1]Note, however that they could be used with any phrase-based system.

2 n-gram-based CTMs

The n-gram-based approach in Machine Translation is a variant of the phrase-based approach (Zens et al, 2002). Introduced in (Casacuberta and Vidal, 2004), and extended in (Mariño et al, 2006; Crego and Mariño, 2006), this approach is based on a specific factorization of the joint probability of parallel sentence pairs, where the source sentence has been reordered beforehand.

2.1 n-gram-based Machine Translation

Let $(\mathbf{s}, \mathbf{t})$ denote a sentence pair made of a source $\mathbf{s}$ and target $\mathbf{t}$ sides. This sentence pair is decomposed into a sequence of L bilingual units called *tuples* defining a joint segmentation: $(\mathbf{s}, \mathbf{t}) = (\mathrm{u}_1 ... \mathrm{u}_L)$. Tuples constitute the basic translation units: like phrase pairs, they represent a matching between a source and a target chunk. The joint probability of a *synchronized* and *segmented* sentence pair can be decomposed using the n-gram assumption as follows:

$$P(\mathbf{s}, \mathbf{t}) = \prod_{i=1}^{L} P(\mathrm{u}_i | \mathrm{u}_{i-n+1}^{i-1}), \tag{1}$$

where u_{i-n+1}^{i-1} denotes the tuple sequence $\mathrm{u}_{i-n+1}, \ldots, \mathrm{u}_{i-1}$.[2] During training, the segmentation is obtained as a by-product of source reordering and ultimately derives from initial word and phrase alignments (see (Crego and Mariño, 2006) for details). During the inference step, the SMT decoder will compute and output the best derivation in a small set of pre-defined reorderings.

The n-gram translation model manipulates *bilingual tuples*; the underlying set of events considered is thus much larger than for word-based language models, while the training data (parallel corpora) are typically order of magnitude smaller than monolingual resources. As a consequence, data sparsity issues for such models are particularly severe. Effective workarounds factorize the conditional probability of tuples (1) into terms involving smaller units: the resulting model thus splits bilingual phrases in two sequences of respectively source and target words, synchronised by the tuple segmentation. Such bilingual word-based n-gram models were initially described in (Le et al, 2012) and extended in (Devlin et al, 2014). We assume here a similar decomposition.

[2]Note that the complete model for a sentence pair involves latent variables that specify the reordering of the source sentence, as well as its segmentation into translation units. These are omitted henceforth for the sake of clarity.

2.2 Neural Architectures

The estimation of n-gram probabilities can be performed via multi-layer NN structures, as described in (Bengio et al, 2003; Schwenk, 2007) for a monolingual language model. The standard *feed-forward* structure is used to estimate the translation models sketched in the previous section. We give here a brief description, see details in (Le et al, 2012): first, each context word is projected into language dependent continuous spaces, using two projection matrices for the source and target languages. The continuous representations are then concatenated to form the representation of the context, which is input to a feed-forward NN predicting a target word.

In this architecture, the size of output vocabulary is a bottleneck when normalized distributions are needed. Various workarounds have been proposed, relying for instance on a structured output layer using word-classes (Mnih and Hinton, 2008; Le et al, 2011). A more effective alternative, which however only delivers *quasi-normalized* scores, is to train the network using the *Noise Contrastive Estimation* or NCE (Gutmann and Hyvärinen, 2010; Mnih and Teh, 2012). This technique is readily applicable for CTMs and has been adopted here. We therefore only assume that the NN outputs a positive score $\mathbf{b}_{\boldsymbol{\theta}}(w, \mathbf{c})$ for each word w given its context $\mathbf{c}$; this score is simply computed as $\mathbf{b}_{\boldsymbol{\theta}}(w, \mathbf{c}) = \exp(\mathbf{a}_{\boldsymbol{\theta}}(w, \mathbf{c}))$, where $\mathbf{a}_{\boldsymbol{\theta}}(w, \mathbf{c})$ is the activation at the output layer; $\boldsymbol{\theta}$ denotes all the network free parameters.

3 Training CTMs discriminatively

In our architecture, the primary role of CTMs is to rerank a set of base hypotheses so that the best hypotheses (w.r.t some automatic metric such as BLEU (Papineni et al, 2002)) are also the top scoring ones. Given the computational burden of evaluating continuous models, an effective use of CTMs is to rescore a list of N-best hypotheses, a scenario that we favor here; note that their integration in a first pass search is also possible (Niehues and Waibel, 2012; Vaswani et al, 2013; Devlin et al, 2014).

In reranking, the CTM score is combined with scores corresponding to other components of the system, such as the reordering model(s) or the monolingual language model(s), etc. We claim that CTM *training* should take these other scores into account. In this section, we thus develop a generic discriminative training framework where the training of the CTM is tightly integrated with the rest of the system.

3.1 A Generic Discriminative Training Framework

The decoder generates a list of N-best hypotheses for each source sentence $\mathbf{s}$. Each hypothesis $\mathbf{h}$ is composed of a target sentence $\mathbf{t}$ along with its associated derivation and is evaluated as follows:

$$G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}) = \sum_{k=1}^{K} \lambda_k f_k(\mathbf{s},\mathbf{h}) + \lambda_{K+1} f_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}), \tag{2}$$

where K conventional feature functions[3] $f_1 ... f_K$, estimated during the training phase, are scaled by coefficients $\lambda_1 ... \lambda_K$. In equation (2), the pair $(\mathbf{s},\mathbf{h})$ represents all the latent variables implied in the translation process. In an n-gram-based system, they correspond to the reordering and the segmentation into bilingual tuples (cf. § 2.1). The continuous model used in rescoring adds a supplementary feature $f_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{h})$, which accumulates NN scores over all contexts $\mathbf{c}$ and words w in the derivation:

$$f_{\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}) = \sum_{(w,\mathbf{c})\in(\mathbf{s},\mathbf{h})} \log \mathbf{b}_{\boldsymbol{\theta}}(w,\mathbf{c}).$$

$G_{\boldsymbol{\lambda},\boldsymbol{\theta}}$ thus depends both on the NN parameters $\boldsymbol{\theta}$ and on the log-linear coefficients $\boldsymbol{\lambda}$. We propose to jointly train these two sets of parameters, by alternatively updating $\boldsymbol{\theta}$ through stochastic gradient descent on the training corpus and updating $\boldsymbol{\lambda}$ using conventional tuning algorithms on the development data. This procedure, also adopted in recent studies (e.g. (He and Deng, 2012; Gao and He, 2013; Gao et al, 2014)), is sketched in algorithm 1. For practical reasons, the NN training data is divided into mini-batches, which are used to compute the sub-gradient of the appropriate training criterion (denoted by $\mathcal{L}(\boldsymbol{\theta})$, see section 3.2.1) and to update $\boldsymbol{\theta}$. After each training iteration of the CTM, the $\boldsymbol{\lambda}$s are retuned on the development set. For that purpose, several optimizers can be used such as Minimum Error Rate Training (MERT) (Och, 2003), Pairwise Ranking Optimization (PRO) (Hopkins and May, 2011), or the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003). All these optimizers are implemented in MOSES.[4]

Figure 1 recaps the training process. Two training corpora[5] are required: the first one (*out-of-domain*) is used to train (see left part of Figure 1) a baseline translation system, the second one (*in-domain*) (on the right part) to optimize the NN parameters $\boldsymbol{\theta}$. Our approach departs from conventional training frameworks

[3]The features used in our experiments are standard phrase-based features, see e.g. (Crego et al, 2011).

[4]`http://www.statmt.org/moses/`. For MIRA, we use the *KB MIRA* implementation (Cherry and Foster, 2012).

[5]Note that these corpora need not necessarily be distinct and can also partly overlap. For the sake of this presentation, we refer to these corpora respectively as the *out-of-domain* and the *in-domain* data. This also corresponds to our experimental setting

Algorithm 1 Joint optimization of $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$

Init. of $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$
for each iteration **do**
  for P mini-batch **do** ▷ $\boldsymbol{\lambda}$ is fixed
    Compute the sub-gradient of $\mathcal{L}(\boldsymbol{\theta})$ for each sentence $\mathbf{s}$ in the mini-batch
    Update $\boldsymbol{\theta}$
  end for
  Update $\boldsymbol{\lambda}$ on development set ▷ $\boldsymbol{\theta}$ is fixed
end for

in the interaction between the NN training and the tuning of the other feature weights (visualized by red connections in the right bottom of figure 1): for a given set of $\boldsymbol{\lambda}$s, the N-best-lists generated for NN training is first rescored by the neural model allowing us to update $\boldsymbol{\theta}$; a new pass of tuning then reestimates the $\boldsymbol{\lambda}$s. Note that implementing this architecture requires to translate the *in-domain* corpus with the baseline system, so as to generate the N-lists that are needed to train the NN parameters (see below). Unlike the baseline system tuning step, we only perform this decoding once.

3.2 Discriminative Loss Functions

In this section, we describe various loss functions that can be used to discriminatively train CTMs. Starting from max-margin and pairwise ranking approaches, we define a loss function which borrows ideas from both techniques. We also recall the definition of the expected-BLEU criterion,[6] initially introduced in (Zens et al, 2007) and used since in many studies, notably in (Gao et al, 2014).

3.2.1 A max-margin approach

As explained above, each hypothesis $\mathbf{h}_i$ produced by the decoder is scored according to (2). Its quality can also be evaluated by the sentence-level approximation of the BLEU score $SBLEU(\mathbf{h}_i)$. Let $\mathbf{h}^*$ denote the hypothesis having the highest sentence BLEU score. A max-margin loss function (Freund and Schapire, 1999; McDonald et al, 2005; Watanabe et al, 2007) for estimating $\boldsymbol{\theta}$ can then be

[6] Two variants of expected-BLEU exist in the literature: one (that we use here) takes the expectation of BLEU score over an approximation of the search space; the other, used for instance in (Rosti et al, 2010), computes BLEU with expected n-gram statistics.

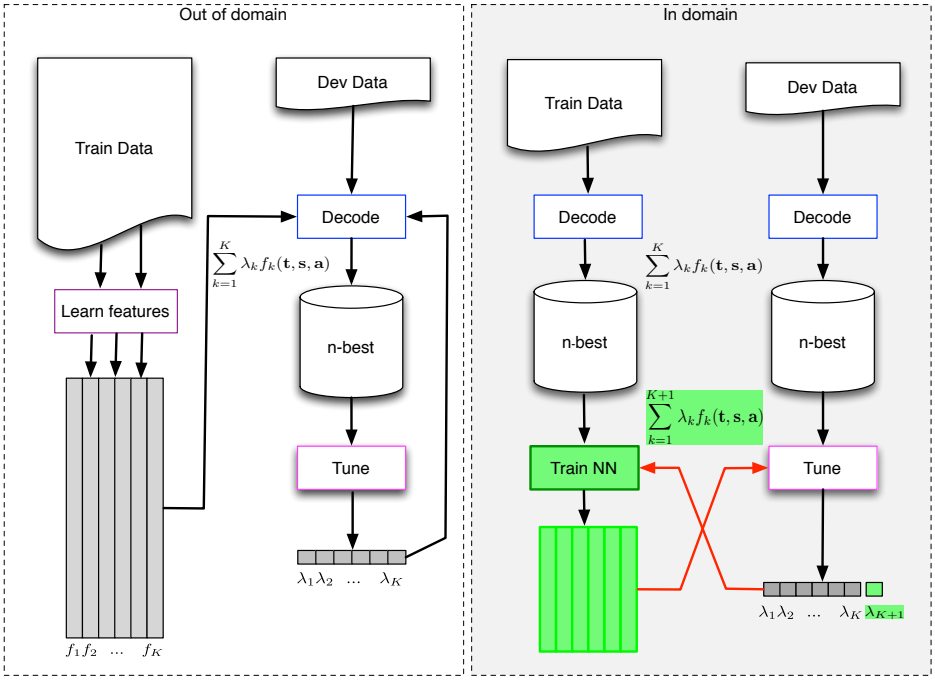

Figure 1: The whole training process uses two corpora: the first one to train a baseline system, while the second one to perform the joint discriminative training of $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$. Each source sentence in the "in-domain" corpus needs to be processed by the baseline system to generate a list of N-best hypotheses.

formulated as follows:

$$\mathcal{L}_{mm}(\boldsymbol{\theta}) = -G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s}, \mathbf{h}^*) + \max_{1 \leq j \leq N} \left(G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s}, \mathbf{h}_j) + \text{cost}_\alpha(\mathbf{h}_j) \right), \tag{3}$$

where $\text{cost}_\alpha(\mathbf{h}_j) = \alpha\big(SBLEU(\mathbf{h}^*) - SBLEU(\mathbf{h}_j)\big)$. The parameter α mitigates the contribution of the cost function to the objective function; when $\alpha = 0$, this criterion is equivalent to the structured perceptron loss (Collins, 2002). This objective function aims to discriminatively learn to give the highest model score to the hypothesis $\mathbf{h}^*$ having the best sentence level BLEU. Moreover, the margin term enforces the score difference between $\mathbf{h}^*$ and the rest of the N-best list to be greater than a margin.

However, it is often the case that the N-best list contains several good translations, that differ only slightly from the best hypothesis. The max-margin objective function defined above nevertheless considers all hypotheses, except the best one, to be wrong. The ranking-based approach defined below tries to correct this weakness.

3.2.2 Pairwise ranking

Inspired by (Hopkins and May, 2011), we define another objective function that learns the ranking of a set of hypotheses with respect to their BLEU scores. Assuming that r_i denotes the rank of the hypothesis $\mathbf{h}_i$ when the N-best list is reordered according to the sentence-level BLEU, this objective is defined as:

$$\mathcal{L}_{pro}(\boldsymbol{\theta}) = \sum_{1 \leq i,k \leq N} \mathbb{I}_{\{r_i + \delta \leq r_k, G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_i) < G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_k)\}} \left(-G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_i) + G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_k)\right), \quad (4)$$

where $\mathbb{I}_x$ denotes an indicator function which returns 1 when the condition x is true and 0 otherwise. This loss function only involves a subset of the $N(N-1)/2$ pairs of hypotheses, since two hypotheses are included in the sum only if they are sufficiently apart in terms of their ranks: formally, the absolute difference of ranks should be greater than a predefined threshold δ. Like in PRO (Hopkins and May, 2011), the ranking problem is thus reduced to a binary classification task taking candidate translation pairs as inputs. A major difference with PRO, though, is the fact that we use this loss function to train the CTM's parameters $\boldsymbol{\theta}$, rather than the feature weights $\boldsymbol{\lambda}$.

3.2.3 Combining max-margin and pairwise ranking

The pairwise ranking criterion can be generalized with the notion of margin: for a pair of hypotheses $(\mathbf{h}_i, \mathbf{h}_k)$ such as $r_i + \delta < r_k$, the scoring difference $G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_i) - G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_k)$ should exceed a positive margin. Therefore, a pair of hypotheses is deemed critical when this constraint is violated and the set of all critical pairs of hypotheses is defined as:

$$\begin{aligned} \mathcal{C}_\delta^\alpha = &\{(i,k) : 1 \leq i,k \leq N, r_i + \delta \leq r_k, \\ &G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_i) - G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_k) < \text{cost}_\alpha(\mathbf{h}_k) - \text{cost}_\alpha(\mathbf{h}_i)\}. \end{aligned} \quad (5)$$

As above, the margin takes into account the sentence-level BLEU scores via the use of the cost function cost_α. Taking both the pairwise ranking and the max-margin criterion into account, we obtain the following objective function:

$$\mathcal{L}_{pro-mm}(\boldsymbol{\theta}) = \sum_{(i,k) \in \mathcal{C}_\delta^\alpha} -G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_i) + G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_k). \quad (6)$$

When $\alpha = 0$, this function is equivalent to the pairwise ranking criterion (4).

3.2.4 Expected-BLEU

Another way to introduce the notion of translation quality consists in approximating the expectation of the BLEU score using N-best lists. For a given source sentence, this loss function is defined as:

$$\mathcal{L}_{x\text{BLEU}}(\boldsymbol{\theta}) = -\sum_{1\leq i\leq N} SBLEU(\mathbf{h}_i) P_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{h}_i|\mathbf{s}). \tag{7}$$

The term $P_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{h}_i|\mathbf{s})$ represents the probability of an hypothesis given the source sentence and can be computed as follows:

$$P_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{h}_i|\mathbf{s}) = \frac{\exp(\gamma G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_i))}{\sum_{1\leq j\leq N} \exp(\gamma G_{\boldsymbol{\lambda},\boldsymbol{\theta}}(\mathbf{s},\mathbf{h}_j))}, \tag{8}$$

where $\gamma \in [0,+\infty)$ is a scaling factor that flattens the distribution for $\gamma < 1$ and sharpens it for $\gamma > 1$. Following (Auli and Gao, 2014a), this hyper-parameter is set to 1. In comparison to the other losses, this loss function takes into account all the hypotheses in the N-best list, weighting the contribution of each candidate translation by a measure of its quality.

3.3 Initialization issues

Initialization is an important issue when optimizing neural networks, since the stochastic gradient descent algorithm only converges to a local optimum. Moreover, our training procedure heavily depends on the log-linear coefficients $\boldsymbol{\lambda}$. These coefficients reflect the relevance of the associated feature functions f_k for ranking hypotheses. However, at the beginning of the discriminative training procedure, the CTM is close to its random initialization. The related feature function (f_{K+1}) is therefore not informative and the optimization algorithm will set its coefficient (λ_{K+1}) near 0. In such configuration, discriminative training is ineffective since the error signal used to update of the CTM is also close to 0.

As a workaround, experiments (Do et al, 2014, 2015a) show that it is more efficient to start with a NCE pre-trained NN, while the discriminative loss is used in a fine-tuning phase. Given the pre-trained CTM's scores, we initialize $\boldsymbol{\lambda}$ by optimizing it on the development set. As the CTM has been pre-trained, this step always delivers a positive value for $\boldsymbol{\lambda}_{K+1}$, which will not mislead the discriminative training. Moreover, this strategy forces the training of $\boldsymbol{\theta}$ to focus on errors made by the system as a whole.

4 Experiments

4.1 Tasks and Corpora

The discriminative optimization framework is evaluated in an adaptation scenario, where large *out-of-domain* corpora are used to train the baseline SMT system, while the CTM is trained on a much smaller, *in-domain* corpus and only serves for rescoring. To assess the impact of this discriminative framework, the experimental set-up is based on the TED Talks task.[7] The parallel *in-domain* data contains 180K sentence pairs; the *out-of-domain* data is much larger and contains all corpora allowed in the translation shared task of WMT'14 (English-French), amounting to 12M parallel sentences. In this setup, training the CTM on the *in-domain* data as the effect of adapting a large scale *out-of-domain* system. The retuning phase for the complete system also uses an in-domain development set.

The baseline translation system is n-code, an open source implementation[8] of the bilingual n-gram approach to MT. A full description of this system is given in (Allauzen et al, 2013). For the NN architecture, each vocabulary word is projected into a 500-dimension space followed by two hidden layers of respectively 1000 and 500 units. Each hidden layer has a sigmoid activation function. For discriminative training, the baseline SMT system is used to generate 300 best hypotheses for each sentence of the in-domain corpus. The threshold δ is set to 250. All our MT experiments use BLEU (Papineni et al, 2002) as the automatic evaluation metric; all reported results are averages over 4 optimization runs (the last update of $\boldsymbol{\lambda}$). Additional experiments on hyper-parameters setting are reported in (Do et al, 2014; Do, 2016).

4.2 Experimental results

Table 1 compares the results obtained under this configuration using the various loss functions described in section 3.2. The upper part reports baseline scores on the development and test sets, as well as the improvements obtained by integrating a CTM. This model is trained using NCE and its addition outperforms the baseline by 1 BLEU point. This score will serve as the comparison point to evaluate discriminative loss functions. The lower part of table 1 reports the BLEU scores obtained with a discriminatively trained CTM. The best results is obtained with the loss function that combines the max-margin and pairwise ranking. Out of the four losses, $\mathcal{L}_{pro-mm}$ yields the largest improvement over the NCE baseline, increasing the BLEU score by more than 1 point. In our setting, $\mathcal{L}_{pro}$ is the second

[7] `http://workshop2014.iwslt.org/`

[8] `http://ncode.limsi.fr/`

	dev	test
n-code Baseline system on WMT	28.5	32.0
n-code Baseline + CTM NCE	29.2	33.0
n-code Baseline + discriminatively trained CTM		
$\mathcal{L}_{mm}$ (Max-margin), $\alpha = 100$	29.6	33.1
$\mathcal{L}_{pro}$ (Pairwise ranking)	29.6	33.4
$\mathcal{L}_{pro-mm}$, $\alpha = 75$	**29.8**	**34.1**
$\mathcal{L}_{x\text{BLEU}}$ (expected-BLEU)	29.2	33.0

Table 1: A comparison of discriminative loss functions

	dev	test
Random init.	28.8	32.7
Monolingual init.	29.6	33.6
Bilingual init.	**29.8**	**34.1**

Table 2: Comparison of initialization schemes, where the CTM is initialized randomly, or with two monolingual language models, or simply pre-trained with NCE criterion (Bilingual init.).

best choice. However, results on the development set suggest that $\mathcal{L}_{pro}$ tends to overfit, while this effect can be reduced with the margin term of $\mathcal{L}_{pro-mm}$.

Tables 2 and 3 provides control experiments using the best configuration observed in table 1. Table 2 shows the benefits of choosing an appropriate initial value for the NN parameter, with a variance of almost 1.5 BLEU between the best and the worst initialization schemes. Table 3 contrasts several ways to choose the training data: in the first setting, the baseline system is entirely in-domain, and the NN is trained with the same data as the baseline; in the second, the baseline system is out-of-domain, and NN training can be understood as a mere adaptation using in-domain data. As our result show, the former approach is much worst. Since the SMT system used to generete the N-best lists is trained on the same data, it produced unreasonably good n-best lists for the NN learning procedure, while this is not the case during the tuning and testing steps.[9]

The training procedure used so far has consistently relied on KB-MIRA to optimize the log-linear coefficient, while the NN parameters have been trained with other losses. In our last experiments, we evaluate the impact of this decision and contrast KB-MIRA with PRO (Hopkins and May, 2011) for tuning the $\boldsymbol{\lambda}$s. As it turns out (Table 4), using a ranking loss both for tuning and training the NN has

[9]This is reflected in the **train** column, where we observe an important difference in BLEU score between the both scenarios.

	dev	test	train
"Training"			
n-code Baseline on TED	28.1	32.3	**65.6**
n-code Baseline + CTM NCE	28.9	33.1	64.1
n-code Baseline + CTM discriminative	**29.0**	**33.5**	64.9
"Adaptation"			
n-code Baseline on WMT	28.5	32.0	33.3
n-code Baseline + CTM NCE	29.2	33.0	34.9
n-code Baseline + CTM discriminative	**29.8**	**34.1**	**35.8**

Table 3: BLEU scores for various data selection scenario.

a beneficial effect and we managed to obtain our best results with combinations of PRO and $\mathcal{L}_{pro}$ (+1.3 BLEU) and of PRO and $\mathcal{L}_{pro-mm}$ (+1.6 BLEU). Note that in this case both losses are consistent. The results obtained using consistent max-margin criteria are comparatively much worse: this suggests that our implementation of $\mathcal{L}_{mm}$ might be suboptimal, and could be improved by smoothing the BLEU-based criteria, as is done in KB-MIRA.

	Loss for $\boldsymbol{\lambda}$			
	MIRA		PRO	
Loss for $\boldsymbol{\theta}$	**dev**	**test**	**dev**	**test**
n-code Baseline	28.5	32.0	27.8	31.7
n-code Baseline + CTM NCE	29.2	33.0	28.8	33.7
n-code Baseline + discriminatively trained CTM				
$\mathcal{L}_{mm}$ (Max-margin), $\alpha = 100$	29.6	32.9	28.7	33.0
$\mathcal{L}_{pro}$ (Pairwise ranking)	29.6	33.4	29.2	34.3
$\mathcal{L}_{pro-mm}$, $\alpha = 75$	29.8	**34.1**	29.4	**34.6**

Table 4: Comparison between the loss functions used to optimize $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$

5 Related work

Conventional MT systems, be they phrase-based, n-gram based, syntax-based or hierarchical, are typically trained in two steps: the first step (*training*) estimates individual features functions; the second one (*tuning*) learns to combine these features so as to optimize translation quality, for instance using Minimum Error Rate Training (MERT) (Och, 2003). The limitations of MERT, notably its inability to train feature sets containing more than a dozen of features, have long been

reported, and more effective discriminative training procedures have been sought (see (Neubig and Watanabe, 2016) for a recent review).

Early proposals have investigated the use of global optimization frameworks to train a complete translation model (Liang et al, 2006; Blunsom et al, 2008; Blunsom and Osborne, 2008; Dyer and Resnik, 2010; Lavergne et al, 2011, 2013). In this framework, all the parameters are learnt discriminatively in a unified manner, by optimizing a well-understood objective function, such as the log-likelihood, over the entire training set. This methodology dispenses with the need to build separate models and to tune their interpolation weights. No matter how appealing this approach might sound, these approaches do not scale up to large systems, and face fundamental design problems, such as the choice of appropriate references (or pseudo-references); moreover, it is not immediately obvious how they could integrate continous space models.

Regarding the loss function, perceptron-based learning has first been introduced in (Shen and Joshi, 2005; Liang et al, 2006). However, margin-based algorithms such as MIRA (Watanabe et al, 2007; Chiang et al, 2008; Cherry and Foster, 2012) are nowadays considered as more efficient to train feature-rich translation systems. This property is particularly relevant in our setting, as we learn large sets of parameters ($\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$). Another recent popular trend has been to adapt the learning to rank framework to tune SMT systems (Shen et al, 2004; Shen and Joshi, 2005; Hopkins and May, 2011; Simianer et al, 2012). The ranking task corresponds to the integration of CTM based on N-best list rescoring. Our objective functions borrow from these two lines of research to both train the CTM ($\boldsymbol{\theta}$) and to tune its contribution ($\boldsymbol{\lambda}$). This procedure can thus be considered as an instance of *discriminative integrated training*.

The architecture described in section 3 has previously been used to jointly train parameters of sparse ($\boldsymbol{\theta}$) and dense ($\boldsymbol{\lambda}$) features: in (He and Deng, 2012; Gao and He, 2013; Gao et al, 2014) the sparse features are phrase pairs, in (Auli and Gao, 2014a) $\boldsymbol{\theta}$ parameterizes a recurrent NNLM. Note that all these works optimize expected-BLEU, which is another way to take multiple hypotheses (and not just the best one) into account when training the system. In these studies, the $\boldsymbol{\theta}$s are trained only once, whereas we see benefits in performing multiple iterations of the general procedure sketched in Algorithm 1. Also note that although N-best rescoring is used in this work to facilitate discriminative training, the integration of CTM's into SMT could be performed differently, eg. with lattice reranking (Auli et al, 2013) or direct decoding with CTM (Niehues and Waibel, 2012; Devlin et al, 2014; Auli and Gao, 2014a).

To the best of our knowledge, the most similar work on discriminative training or adaptation of neural network models is (Gao et al, 2014). The authors propose to estimate a neural network-based phrase translation model using expected-BLEU, while tuning $\boldsymbol{\lambda}$s with standard tools, a strategy we also adopt here. We

however consider alternative loss functions and also preserve the sequential structure of joint model, where (Gao et al, 2014) uses a separate bag-of-word representation of source and target phrases. Expected-BLEU training has also been applied to recurrent NNLM (Auli and Gao, 2014b). For ranking language models, (Collobert and Weston, 2008; Collobert et al, 2011) also introduce a ranking-type objective function, but which aims only to estimate word embeddings in a multi-task learning framework. Furthermore, (Socher et al, 2013) investigates the use of a max-margin criterion to train a recursive neural network for syntactic parsing. Interestingly, their model is also used to rerank N-best derivations generated by a conventional probabilistic context-free grammar. However, as showed in our experiments, the max-margin criterion alone is less adapted to SMT as it lacks of a truly reliable and unambiguous metrics for evaluating translation quality.

Regarding the CTM's structure, our model is based on the feed-forward CTM described in (Le et al, 2012) and extended in (Devlin et al, 2014). This structure, though simple, has been shown to achieve consistent improvement in performance over a wide array of tasks. Moreover, efficient computational tricks are available for this architecture and greatly speed up training and inference. While the models in (Le et al, 2012) employ a structured output layer to reduce the cost of *softmax* operations, we have chosen here to use a *self-normalized* NCE training, which is also very efficient. Another form of self-normalization is presented in (Devlin et al, 2014), but is computationnally less efficient.

This review would not be complete without mentioning Neural Machine Translation (NMT) systems (Cho et al, 2014; Bahdanau et al, 2014; Sutskever et al, 2014). These recent architectures implement an arguably more direct model of translation, which is entirely computed with recurrent NNs; training however usually optimizes the log-likelihood, where we successfully attempt to optimize a translation quality metric. Such discrimininative criteria could certainly also be used for training NMT, as was already done for expected-BLEU in (Shen et al, 2015).

6 Conclusions

In this paper, we have proposed a new discriminative training procedure for continuous-space translation models, which correlates better with translation quality than conventional training methods. This procedure has been validated using an n-gram-based CTM, but the general idea could be applied to other continuous models which compute a score for each translation hypothesis. The core of the method lays in the definition of a new objective function inspired both from max-margin and Pairwise Ranking approach in MT, which enables us to effectively integrate the CTM into the SMT system through N-best rescoring. A major difference with

most past efforts along these lines is the joint training of the CTM and the log-linear parameters. In all our experiments, discriminative training, when applied on a CTM initially trained with NCE, yields substantial performance gains.

References

Allauzen A, Pécheux N, Do QK, Dinarelli M, Lavergne T, Max A, Le H, Yvon F (2013) LIMSI @ WMT13. In: Proceedings of the Workshop on Statistical Machine Translation, Sofia, Bulgaria, pp 62–69

Auli M, Gao J (2014a) Decoder integration and expected bleu training for recurrent neural network language models. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), pp 136–142

Auli M, Gao J (2014b) Decoder integration and expected BLEU training for recurrent neural network language models. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14), pp 136–142

Auli M, Galley M, Quirk C, Zweig G (2013) Joint language and translation modeling with recurrent neural networks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp 1044–1054

Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:14090473

Bengio Y, Ducharme R, Vincent P, Janvin C (2003) A neural probabilistic language model. Journal of Machine Learning Research 3:1137–1155

Blunsom P, Osborne M (2008) Probabilistic inference for machine translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp 215–223

Blunsom P, Cohn T, Osborne M (2008) A discriminative latent variable model for statistical machine translation. In: ACL, pp 200–208

Casacuberta F, Vidal E (2004) Machine translation with inferred stochastic finite-state transducers. Computational Linguistics 30(3):205–225

Cherry C, Foster G (2012) Batch tuning strategies for statistical machine translation. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp 427–436

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

Cho K, van Merrienboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, pp 1724–1734

Collins M (2002) Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp 1–8

Collobert R, Weston J (2008) A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning, ACM, pp 160–167

Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. Journal of Machine Learning Research 12:2493–2537

Crammer K, Singer Y (2003) Ultraconservative online algorithms for multiclass problems. Journal of Machine Learning Research 3:951–991

Crego JM, Mariño JB (2006) Improving statistical MT by coupling reordering and decoding. Machine Translation 20(3):199–215

Crego JM, Yvon F, Mariño JB (2011) N-code: an open-source bilingual N-gram SMT toolkit. Prague Bulletin of Mathematical Linguistics 96:49–58

Devlin J, Zbib R, Huang Z, Lamar T, Schwartz R, Makhoul J (2014) Fast and robust neural network joint models for statistical machine translation. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, pp 1370–1380

Do QK (2016) Discriminative training of continuous space translation models. PhD thesis, Université Paris-Sud and Université Paris-Saclay

Do QK, Allauzen A, Yvon F (2014) Discriminative adaptation of continuous space translation models. In: International Workshop on Spoken Language Translation (IWSLT 2014), Lake Tahoe, USA

Do QK, Allauzen A, Yvon F (2015a) Apprentissage discriminant des modèles continus de traduction. In: Actes de la 22e conférence sur le Traitement Automatique des Langues Naturelles, Association pour le Traitement Automatique des Langues, Caen, France, pp 267–278

Do QK, Allauzen A, Yvon F (2015b) A discriminative training procedure for continuous translation models. In: Conference on Empirical Methods in Natural Language Processing (EMNLP 2015), Lisboa, Portugal, p 7

Dyer C, Resnik P (2010) Context-free reordering, finite-state translation. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp 858–866

Freund Y, Schapire RE (1999) Large margin classification using the perceptron algorithm. Machine learning 37(3):277–296

Gao J, He X (2013) Training MRF-Based Phrase Translation Models using Gradient Ascent. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Atlanta, pp 450–459

Gao J, He X, Yih Wt, Deng L (2014) Learning continuous phrase representations for translation modeling. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD

Gutmann M, Hyvärinen A (2010) Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Teh YW, Titterington M (eds) Proceedings of th International Conference on Artificial Intelligence and Statistics (AISTATS), vol 9, pp 297–304

He X, Deng L (2012) Maximum expected bleu training of phrase and lexicon translation models. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1, pp 292–301

Hopkins M, May J (2011) Tuning as ranking. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Edinburgh, Scotland, UK., pp 1352–1362

Lavergne T, Crego JM, Allauzen A, Yvon F (2011) From n-gram-based to CRF-based translation models. In: Proceedings of the Sixth Workshop on Statistical Machine Translation, pp 542–553

Lavergne T, Allauzen A, Yvon F (2013) Un cadre d'apprentissage intégralement discriminant pour la traduction statistique. TALN-RÉCITAL 2013 p 450

Le HS, Oparin I, Allauzen A, Gauvain JL, Yvon F (2011) Structured output layer neural network language model. In: Proceedings of the International Conference on Audio, Speech and Signal Processing, pp 5524–5527

Le HS, Allauzen A, Yvon F (2012) Continuous space translation models with neural networks. In: Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Montréal, Canada, pp 39–48

Liang P, Bouchard-Côté A, Klein D, Taskar B (2006) An end-to-end discriminative approach to machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), pp 761–768

Mariño JB, Banchs RE, Crego JM, de Gispert A, Lambert P, Fonollosa JA, Costa-Jussà MR (2006) N-gram-based machine translation. Computational Linguistics 32(4):527–549

McDonald R, Crammer K, Pereira F (2005) Online large-margin training of dependency parsers. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), pp 91–98

Mnih A, Hinton GE (2008) A scalable hierarchical distributed language model. In: Koller D, Schuurmans D, Bengio Y, Bottou L (eds) Advances in Neural Information Processing Systems 21, vol 21, pp 1081–1088

Mnih A, Teh YW (2012) A fast and simple algorithm for training neural probabilistic language models. In: Proceedings of the International Conference of Machine Learning (ICML)

Neubig G, Watanabe T (2016) Optimization for statistical machine translation: A survey. Computational Linguistics 42(1):1–54

Niehues J, Waibel A (2012) Continuous space language models using restricted Boltzmann machines. In: Proceedings of International Workshop on Spoken Language Translation (IWSLT), Hong-Kong, China, pp 164–170

Och FJ (2003) Minimum error rate training in statistical machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Association for Computational Linguistics, pp 160–167

Papineni K, Roukos S, Ward T, Zhu WJ (2002) Bleu: a method for automatic evaluation of machine translation. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), pp 311–318

Rosti AV, Zhang B, Matsoukas S, Schwartz R (2010) BBN System Description for WMT10 System Combination Task. In: Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR, Association for Computational Linguistics, Uppsala, Sweden, pp 321–326

Schwenk H (2007) Continuous space language models. Computer Speech and Language 21(3):492–518

Schwenk H, R Costa-jussa M, R Fonollosa JA (2007) Smooth bilingual n-gram translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Prague, Czech Republic, pp 430–438

Shen L, Joshi AK (2005) Ranking and reranking with perceptron. Machine Learning 60(1-3):73–96

Shen L, Sarkar A, Och FJ (2004) Discriminative reranking for machine translation. In: HLT-NAACL, pp 177–184

Shen S, Cheng Y, He Z, He W, Wu H, Sun M, Liu Y (2015) Minimum risk training for neural machine translation. CoRR abs/1512.02433

Simianer P, Riezler S, Dyer C (2012) Joint feature selection in distributed stochastic learning for large-scale discriminative training in SMT. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), pp 11–21

Socher R, Bauer J, Manning CD, Andrew Y N (2013) Parsing with compositional vector grammars. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Sofia, Bulgaria, pp 455–465

Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, Montréal, Canada, NIPS*27, pp 3104–3112

Vaswani A, Zhao Y, Fossum V, Chiang D (2013) Decoding with large-scale neural language models improves translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Seattle, Washington, USA, pp 1387–1392

Watanabe T, Suzuki J, Tsukada H, Isozaki H (2007) Online large-margin training for statistical machine translation. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, pp 764–773

Yang N, Liu S, Li M, Zhou M, Yu N (2013) Word alignment modeling with context dependent deep neural networks. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Sofia, Bulgaria, pp 166–175

Zens R, Och FJ, Ney H (2002) Phrase-based statistical machine translation. In: KI '02: Proceedings of the 25th Annual German Conference on AI, Springer-Verlag, London, UK, pp 18–32

Zens R, Hasan S, Ney H (2007) A systematic comparison of training criteria for statistical machine translation. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, pp 524–532