Online Large-Margin Training for Statistical Machine Translation

Taro Watanabe Jun Suzuki Hajime Tsukada Hideki Isozaki
NTT Communication Science Laboratories
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237 Japan
{taro,jun,tsukada,isozaki}@cslab.kecl.ntt.co.jp

Abstract

We achieved a state of the art performance in statistical machine translation by using a large number of features with an online large-margin training algorithm. The millions of parameters were tuned only on a small development set consisting of less than 1K sentences. Experiments on Arabic-to-English translation indicated that a model trained with sparse binary features outperformed a conventional SMT system with a small number of features.

1 Introduction

The recent advances in statistical machine translation have been achieved by discriminatively training a small number of real-valued features based either on (hierarchical) phrase-based translation (Och and Ney, 2004; Koehn et al., 2003; Chiang, 2005) or syntax-based translation (Galley et al., 2006). However, it does not scale well with a large number of features of the order of millions.

Tillmann and Zhang (2006), Liang et al. (2006) and Bangalore et al. (2006) introduced sparse binary features for statistical machine translation trained on a large training corpus. In this framework, the problem of translation is regarded as a sequential labeling problem, in the same way as part-of-speech tagging, chunking or shallow parsing. However, the use of a large number of features did not provide any significant improvements over a conventional small feature set.

Bangalore et al. (2006) trained the lexical choice model by using Conditional Random Fields (CRF) realized on a WFST. Their modeling was reduced to Maximum Entropy Markov Model (MEMM) to handle a large number of features which, in turn, faced the labeling bias problem (Lafferty et al., 2001). Tillmann and Zhang (2006) trained their feature set using an online discriminative algorithm. Since the decoding is still expensive, their online training approach is approximated by enlarging a merged k-best list one-by-one with a 1-best output. Liang et al. (2006) introduced an averaged perceptron algorithm, but employed only 1-best translation. In Watanabe et al. (2006a), binary features were trained only on a small development set using a variant of voted perceptron for reranking k-best translations. Thus, the improvement is merely relative to the baseline translation system, namely whether or not there is a good translation in their k-best.

We present a method to estimate a large number of parameters — of the order of millions — using an online training algorithm. Although it was intuitively considered to be prone to overfitting, training on a small development set — less than 1K sentences — was sufficient to achieve improved performance. In this method, each training sentence is decoded and weights are updated at every iteration (Liang et al., 2006). When updating model parameters, we employ a memorization-variant of a local updating strategy (Liang et al., 2006) in which parameters are optimized toward a set of good translations found in the k-best list across iterations. The objective function is an approximated BLEU (Watanabe et al., 2006a) that scales the loss of a sentence BLEU to a document-wise loss. The parameters are trained using the

Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 764–773, Prague, June 2007.

Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006). MIRA is successfully employed in dependency parsing (McDonald et al., 2005) or the joint-labeling/chunking task (Shimizu and Haas, 2006). Experiments were carried out on an Arabic-to-English translation task, and we achieved significant improvements over conventional minimum error training with a small number of features.

This paper is organized as follows: First, Section 2 introduces the framework of statistical machine translation. As a baseline SMT system, we use the hierarchical phrase-based translation with an efficient left-to-right generation (Watanabe et al., 2006b) originally proposed by Chiang (2005). In Section 3, a set of binary sparse features are defined including numeric features for our baseline system. Section 4 introduces an online large-margin training algorithm using MIRA with our key components. The experiments are presented in Section 5 followed by discussion in Section 6.

2 Statistical Machine Translation

We use a log-linear approach (Och, 2003) in which a foreign language sentence f is translated into another language, for example English, e, by seeking a maximum solution:

$$\hat{e} = \operatorname*{argmax}_{e} \mathbf{w}^T \cdot \mathbf{h}(f, e) \tag{1}$$

where $\mathbf{h}(f, e)$ is a large-dimension feature vector. $\mathbf{w}$ is a weight vector that scales the contribution from each feature. Each feature can take any real value, such as the log of the n-gram language model to represent fluency, or a lexicon model to capture the word or phrase-wise correspondence.

2.1 Hierarchical Phrase-based SMT

Chiang (2005) introduced the hierarchical phrase-based translation approach, in which non-terminals are embedded in each phrase. A translation is generated by hierarchically combining phrases using the non-terminals. Such a quasi-syntactic structure can naturally capture the reordering of phrases that is not directly modeled by a conventional phrase-based approach (Koehn et al., 2003). The non-terminal embedded phrases are learned from a bilingual corpus without a linguistically motivated syntactic structure.

Based on hierarchical phrase-based modeling, we adopted the left-to-right target generation method (Watanabe et al., 2006b). This method is able to generate translations efficiently, first, by simplifying the grammar so that the target side takes a phrase-prefixed form, namely a target normalized form. Second, a translation is generated in a left-to-right manner, similar to the phrase-based approach using Earley-style top-down parsing on the source side. Coupled with the target normalized form, n-gram language models are efficiently integrated during the search even with a higher order of n.

2.2 Target Normalized Form

In Chiang (2005), each production rule is restricted to a rank-2 or binarized form in which each rule contains at most two non-terminals. The target normalized form (Watanabe et al., 2006b) further imposes a constraint whereby the target side of the aligned right-hand side is restricted to a Greibach Normal Form like structure:

$$X \rightarrow \left\langle \gamma, \bar{b}\beta, \sim \right\rangle \tag{2}$$

where X is a non-terminal, γ is a source side string of arbitrary terminals and/or non-terminals. $\bar{b}\beta$ is a corresponding target side where $\bar{b}$ is a string of terminals, or a phrase, and β is a (possibly empty) string of non-terminals. $\sim$ defines one-to-one mapping between non-terminals in γ and β. The use of phrase $\bar{b}$ as a prefix maintains the strength of the phrase-base framework. A contiguous English side with a (possibly) discontiguous foreign language side preserves phrase-bounded local word reordering. At the same time, the target normalized framework still combines phrases hierarchically in a restricted manner.

2.3 Left-to-Right Target Generation

Decoding is performed by parsing on the source side and by combining the projected target side. We applied an Earley-style top-down parsing approach (Wu and Wong, 1998; Watanabe et al., 2006b; Zollmann and Venugopal, 2006). The basic idea is to perform top-down parsing so that the projected target side is generated in a left-to-right manner. The search is guided with a push-down automaton, which keeps track of the span of uncovered source word positions. Combined with the rest-cost estimation aggregated in a bottom-up way, our decoder efficiently searches for the most likely translation.

The use of a target normalized form further simplifies the decoding procedure. Since the rule form does not allow any holes for the target side, the integration with an n-gram language model is straightforward: the prefixed phrases are simply concatenated and intersected with n-gram.

3 Features

3.1 Baseline Features

The hierarchical phrase-based translation system employs standard numeric value features:

- n-gram language model to capture the fluency of the target side.
- Hierarchical phrase translation probabilities in both directions, $h(\gamma|\bar{b}\beta)$ and $h(\bar{b}\beta|\gamma)$, estimated by relative counts, count$(\gamma, \bar{b}\beta)$.
- Word-based lexically weighted models of $h_{lex}(\gamma|\bar{b}\beta)$ and $h_{lex}(\bar{b}\beta|\gamma)$ using lexical translation models.
- Word-based insertion/deletion penalties that penalize through the low probabilities of the lexical translation models (Bender et al., 2004).
- Word/hierarchical-phrase length penalties.
- Backtrack-based penalties inspired by the distortion penalties in phrase-based modeling (Watanabe et al., 2006b).

3.2 Sparse Features

In addition to the baseline features, a large number of binary features are integrated in our MT system. We may use any binary features, such as

$$h(f, e) = \begin{cases} 1 & \text{English word "violate" and Arabic} \\ & \text{word "tnthk" appeared in } e \text{ and } f. \\ 0 & \text{otherwise.} \end{cases}$$

The features are designed by considering the decoding efficiency and are based on the word alignment structure preserved in hierarchical phrase translation pairs (Zens and Ney, 2006). When hierarchical phrases are extracted, the word alignment is preserved. If multiple word alignments are observed with the same source and target sides, only the frequently observed word alignment is kept to reduce the grammar size.

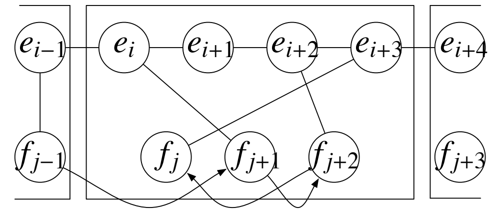

Figure 1: An example of sparse features for a phrase translation.

3.2.1 Word Pair Features

Word pair features reflect the word correspondence in a hierarchical phrase. Figure 1 illustrates an example of sparse features for a phrase translation pair $f_j, ..., f_{j+2}$ and $e_i, ..., e_{i+3}$ [1]. From the word alignment encoded in this phrase, we can extract word pair features of (e_i, f_{j+1}), (e_{i+2}, f_{j+2}) and (e_{i+3}, f_j).

The bigrams of word pairs are also used to capture the contextual dependency. We assume that the word pairs follow the target side ordering. For instance, we define $((e_{i-1}, f_{j-1}), (e_i, f_{j+1}))$, $((e_i, f_{j+1}), (e_{i+2}, f_{j+2}))$ and $((e_{i+2}, f_{j+2}), (e_{i+3}, f_j))$ indicated by the arrows in Figure 1.

Extracting bigram word pair features following the target side ordering implies that the corresponding source side is reordered according to the target side. The reordering of hierarchical phrases is represented by using contextually dependent word pairs across their boundaries, as with the feature $((e_{i-1}, f_{j-1}), (e_i, f_{j+1}))$ in Figure 1.

3.2.2 Insertion Features

The above features are insufficient to capture the translation because spurious words are sometimes inserted in the target side. Therefore, insertion features are integrated in which no word alignment is associated in the target. The inserted words are associated with all the words in the source sentence, such as $(e_{i+1}, f_1), ..., (e_{i+1}, f_J)$ for the non-aligned word e_{i+1} with the source sentence f_1^J in Figure 1. In the

[1] For simplicity, we show an example of phrase translation pairs, but it is trivial to define the features over hierarchical phrases.

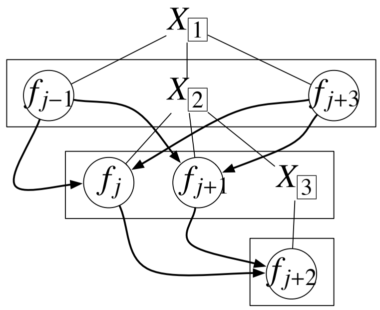

Figure 2: Example hierarchical features.

same way, we will be able to include deletion features where a non-aligned source word is associated with the target sentence. However, this would lead to complex decoding in which all the translated words are memorized for each hypothesis, and thus not integrated in our feature set.

3.2.3 Target Bigram Features

Target side bigram features are also included to directly capture the fluency as in the n-gram language model (Roark et al., 2004). For instance, bigram features of $(e_{i-1}, e_i), (e_i, e_{i+1}), (e_{i+1}, e_{i+2})$... are observed in Figure 1.

3.2.4 Hierarchical Features

In addition to the phrase motivated features, we included features inspired by the hierarchical structure. Figure 2 shows an example of hierarchical phrases in the source side, consisting of $X_{\boxed{1}} \rightarrow \left\langle f_{j-1} X_{\boxed{2}} f_{j+3} \right\rangle$, $X_{\boxed{2}} \rightarrow \left\langle f_j f_{j+1} X_{\boxed{3}} \right\rangle$ and $X_{\boxed{3}} \rightarrow \left\langle f_{j+2} \right\rangle$. Hierarchical features capture the dependency of the source words in a parent phrase to the source words in child phrases, such as (f_{j-1}, f_j), (f_{j-1}, f_{j+1}), (f_{j+3}, f_j), (f_{j+3}, f_{j+1}), (f_j, f_{j+2}) and (f_{j+1}, f_{j+2}) as indicated by the arrows in Figure 2. The hierarchical features are extracted only for those source words that are aligned with the target side to limit the feature size.

3.3 Normalization

In order to achieve the generalization capability, the following normalized tokens are introduced for each surface form:

- Word class or POS.
- 4-letter prefix and suffix. For instance, the word "violate" is normalized to "viol+" and "+late" by taking the prefix and suffix, respectively.
- Digits replaced by a sequence of "@". For example, the word "2007/6/27" is represented as "@@@@/@/@@".

We consider all possible combination of those token types. For example, the word pair feature (violate, tnthk) is normalized and expanded to (viol+, tnthk), (viol+, tnth+), (violate, tnth+), etc. using the 4-letter prefix token type.

Algorithm 1 Online Training Algorithm

Training data: $\mathcal{T} = \{(f^t, \mathbf{e}^t)\}_{t=1}^{T}$
m-best oracles: $\mathcal{O} = \{\}_{t=1}^{T}$
$i = 0$
1: **for** $n = 1, ..., N$ **do**
2: **for** $t = 1, ..., T$ **do**
3: $C^t \leftarrow \text{best}_k(f^t; \mathbf{w}^i)$
4: $\mathcal{O}^t \leftarrow \text{oracle}_m(\mathcal{O}^t \cup C^t; \mathbf{e}^t)$
5: $\mathbf{w}^{i+1}$ = update $\mathbf{w}^i$ using C^t w.r.t. $\mathcal{O}^t$
6: $i = i + 1$
7: **end for**
8: **end for**
9: **return** $\frac{\sum_{i=1}^{NT} \mathbf{w}^i}{NT}$

4 Online Large-Margin Training

Algorithm 1 is our generic online training algorithm. The algorithm is slightly different from other online training algorithms (Tillmann and Zhang, 2006; Liang et al., 2006) in that we keep and update oracle translations, which is a set of good translations reachable by a decoder according to a metric, i.e. BLEU (Papineni et al., 2002). In line 3, a k-best list is generated by $\text{best}_k(\cdot)$ using the current weight vector $\mathbf{w}^i$ for the training instance of $(f^t, \mathbf{e}^t)$. Each training instance has multiple (or, possibly one) reference translations $\mathbf{e}^t$ for the source sentence f^t. Using the k-best list, m-best oracle translations $\mathcal{O}^t$ is updated by $\text{oracle}_m(\cdot)$ for every iteration (line 4). Usually, a decoder cannot generate translations that exactly match the reference translations due to its beam search pruning and OOV. Thus, we cannot always assign scores for each reference translation. Therefore, possible oracle translations are maintained according to an objective function, i.e. BLEU. Tillmann and Zhang (2006) avoided the problem by precomputing the oracle translations in advance. Liang et al. (2006) presented a similar updating strategy in which parameters were updated toward an oracle translation found in C^t, but ignored potentially better translations discovered in the past iterations.

New $\mathbf{w}^{i+1}$ is computed using the k-best list C^t with respect to the oracle translations O^t (line 5). After N iterations, the algorithm returns an averaged weight vector to avoid overfitting (line 9). The key to this online training algorithm is the selection of the updating scheme in line 5.

4.1 Margin Infused Relaxed Algorithm

The Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006) is an online version of the large-margin training algorithm for structured classification (Taskar et al., 2004) that has been successfully used for dependency parsing (McDonald et al., 2005) and joint-labeling/chunking (Shimizu and Haas, 2006). The basic idea is to keep the norm of the updates to the weight vector as small as possible, considering a margin at least as large as the loss of the incorrect classification.

Line 5 of the weight vector update procedure in Algorithm 1 is replaced by the solution of:

$$\begin{aligned}
&\hat{\mathbf{w}}^{i+1} = \operatorname*{argmin}_{\mathbf{w}^{i+1}} \|\mathbf{w}^{i+1} - \mathbf{w}^i\| + C \sum_{\hat{e},e'} \xi(\hat{e}, e') \\
&\text{subject to} \\
&s^{i+1}(f^t, \hat{e}) - s^{i+1}(f^t, e') + \xi(\hat{e}, e') \geq L(\hat{e}, e'; \mathbf{e}^t) \\
&\xi(\hat{e}, e') \geq 0 \\
&\forall \hat{e} \in O^t, \forall e' \in C^t
\end{aligned} \tag{3}$$

where $s^i(f^t, e) = \left\{\mathbf{w}^i\right\}^T \cdot \mathbf{h}(f^t, e)$. $\xi(\cdot)$ is a nonnegative slack variable and $C \geq 0$ is a constant to control the influence to the objective function. A larger C implies larger updates to the weight vector. $L(\cdot)$ is a loss function, for instance difference of BLEU, that measures the difference between $\hat{e}$ and e' according to the reference translations $\mathbf{e}^t$. In this update, a margin is created for each correct and incorrect translation at least as large as the loss of the incorrect translation. A larger error means a larger distance between the scores of the correct and incorrect translations. Following McDonald et al. (2005), only k-best translations are used to form the margins in order to reduce the number of constraints in Eq. 3. In the translation task, multiple translations are acceptable. Thus, margins for m-oracle translation are created, which amount to $m \times k$ large-margin constraints. In this online training, only active features constrained by Eq. 3 are kept and updated, unlike offline training in which all possible features have to be extracted and selected in advance.

The Lagrange dual form of Eq. 3 is:

$$\begin{aligned}
\max_{\alpha(\cdot)\geq 0} \quad & -\frac{1}{2} \| \sum_{\hat{e},e'} \alpha(\hat{e}, e') \left(\mathbf{h}(f^t, \hat{e}) - \mathbf{h}(f^t, e')\right) \|^2 \\
& + \sum_{\hat{e},e'} \alpha(\hat{e}, e') L(\hat{e}, e'; \mathbf{e}^t) \\
& - \sum_{\hat{e},e'} \alpha(\hat{e}, e') \left(s^i(f^t, \hat{e}) - s^i(f^t, e')\right) \\
\text{subject to} \quad & \sum_{\hat{e},e'} \alpha(\hat{e}, e') \leq C
\end{aligned} \tag{4}$$

with the weight vector update:

$$\mathbf{w}^{i+1} = \mathbf{w}^i + \sum_{\hat{e},e'} \alpha(\hat{e}, e') \left(\mathbf{h}(f^t, \hat{e}) - \mathbf{h}(f^t, e')\right) \tag{5}$$

Equation 4 is solved using a QP-solver, such as a coordinate ascent algorithm, by heuristically selecting $(\hat{e}, e')$ and by updating $\alpha(\cdot)$ iteratively:

$$\begin{aligned}
\alpha(\hat{e}, e') &= \max\left(0, \alpha(\hat{e}, e') + \delta(\hat{e}, e')\right) \qquad (6) \\
\delta(\hat{e}, e') &= \frac{L(\hat{e}, e'; \mathbf{e}^t) - \left(s^i(f^t, \hat{e}) - s^i(f^t, e')\right)}{\|\mathbf{h}(f^t, \hat{e}) - \mathbf{h}(f^t, e')\|^2}
\end{aligned}$$

C is used to clip the amount of updates.

A single oracle with 1-best translation is analytically solved without a QP-solver and is represented as the following perceptron-like update (Shimizu and Haas, 2006):

$$\alpha = \max\left(0, \min\left(C, \frac{L(\hat{e}, e'; \mathbf{e}^t) - \left(s^i(f^t, \hat{e}) - s^i(f^t, e')\right)}{\|\mathbf{h}(f^t, \hat{e}) - \mathbf{h}(f^t, e')\|^2}\right)\right)$$

Intuitively, the update amount is controlled by the margin and the loss between the correct and incorrect translations and by the closeness of two translations in terms of feature vectors. Indeed, Liang et al. (2006) employed an averaged perceptron algorithm in which α value was always set to one. Tillmann and Zhang (2006) used a different update style based on a convex loss function:

$$\alpha = \eta L(\hat{e}, e'; \mathbf{e}^t) \cdot \max\left(0, 1 - \left(s^i(f^t, \hat{e}) - s^i(f^t, e')\right)\right)$$

Table 1: Experimental results obtained by varying normalized tokens used with surface form.

	# features	2003 (dev)		2004		2005	
		NIST	BLEU [%]	NIST	BLEU [%]	NIST	BLEU [%]
surface form	492K	11.32	54.11	10.57	49.01	10.77	48.05
w/ prefix/suffix	4,204K	12.38	63.87	10.42	48.74	10.58	47.18
w/ word class	2,689K	10.87	49.59	10.63	49.55	10.89	48.79
w/ digits	576K	11.01	50.72	10.66	49.67	10.84	48.39
all token types	13,759K	11.24	52.85	10.66	49.81	10.85	48.41

where $\eta > 0$ is a learning rate for controlling the convergence.

4.2 Approximated BLEU

We used the BLEU score (Papineni et al., 2002) as the loss function computed by:

$$\mathrm{BLEU}(E; \mathbf{E}) = \exp\left(\frac{1}{N}\sum_{n=1}^{N}\log p_n(E, \mathbf{E})\right)\cdot \mathrm{BP}(E, \mathbf{E}) \tag{7}$$

where $p_n(\cdot)$ is the n-gram precision of hypothesized translations $E = \{e^t\}_{t=1}^{T}$ given reference translations $\mathbf{E} = \{\mathbf{e}^t\}_{t=1}^{T}$ and $\mathrm{BP}(\cdot) \leq 1$ is a brevity penalty. BLEU is computed for a set of sentences, not for a single sentence. Our algorithm requires frequent updates on the weight vector, which implies higher cost in computing the document-wise BLEU. Tillmann and Zhang (2006) and Liang et al. (2006) solved the problem by introducing a sentence-wise BLEU. However, the use of the sentence-wise scoring does not translate directly into the document-wise score because of the n-gram precision statistics and the brevity penalty statistics aggregated for a sentence set. Thus, we use an approximated BLEU score that basically computes BLEU for a sentence set, but accumulates the difference for a particular sentence (Watanabe et al., 2006a).

The approximated BLEU is computed as follows: Given oracle translations O for $\mathcal{T}$, we maintain the best oracle translations $O_1^T = \left\{\hat{e}^1, ..., \hat{e}^T\right\}$. The approximated BLEU for a hypothesized translation e' for the training instance $(f^t, \mathbf{e}^t)$ is computed over O_1^T except for $\hat{e}^t$, which is replaced by e':

$$\mathrm{BLEU}(\{\hat{e}^1, ..., \hat{e}^{t-1}, e', \hat{e}^{t+1}, ..., \hat{e}^T\}; \mathbf{E})$$

The loss computed by the approximated BLEU measures the document-wise loss of substituting the correct translation $\hat{e}^t$ into an incorrect translation e'. The score can be regarded as a normalization which scales a sentence-wise score into a document-wise score.

5 Experiments

We employed our online large-margin training procedure for an Arabic-to-English translation task. The training data were extracted from the Arabic/English news/UN bilingual corpora supplied by LDC. The data amount to nearly 3.8M sentences. The Arabic part of the bilingual data is tokenized by isolating Arabic scripts and punctuation marks. The development set comes from the MT2003 Arabic-English NIST evaluation test set consisting of 663 sentences in the news domain with four reference translations. The performance is evaluated by the news domain MT2004/MT2005 test set consisting of 707 and 1,056 sentences, respectively.

The hierarchical phrase translation pairs are extracted in a standard way (Chiang, 2005): First, the bilingual data are word alignment annotated by running GIZA++ (Och and Ney, 2003) in two directions. Second, the word alignment is refined by a grow-diag-final heuristic (Koehn et al., 2003). Third, phrase translation pairs are extracted together with hierarchical phrases by considering holes. In the last step, the hierarchical phrases are constrained so that they follow the target normalized form constraint. A 5-gram language model is trained on the English side of the bilingual data combined with the English Gigaword from LDC.

First, the use of normalized token types in Section 3.3 is evaluated in Table 1. In this setting, all the structural features in Section 3.2 are used, but differentiated by the normalized tokens combined with surface forms. Our online large-margin training algorithm performed 50 iterations constrained

Table 2: Experimental results obtained by incrementally adding structural features.

	# features	2003 (dev)		2004		2005	
		NIST	BLEU [%]	NIST	BLEU [%]	NIST	BLEU [%]
word pairs	11,042K	11.05	51.63	10.43	48.69	10.73	47.72
+ target bigram	11,230K	11.19	53.49	10.40	48.60	10.66	47.47
+ insertion	13,489K	11.21	52.20	10.77	50.33	10.93	48.08
+ hierarchical	13,759K	11.24	52.85	10.66	49.81	10.85	48.41

Table 3: Experimental results for varying k-best and m-oracle translations.

		# features	2003 (dev)		2004		2005	
			NIST	BLEU [%]	NIST	BLEU [%]	NIST	BLEU [%]
baseline			10.64	46.47	10.83	49.33	10.90	47.03
1-oracle	1-best	8,735K	11.25	52.63	10.82	50.77	10.93	48.11
1-oracle	10-best	10,480K	11.24	53.45	10.55	49.10	10.82	48.49
10-oracle	1-best	8,416K	10.70	47.63	10.83	48.88	10.76	46.00
10-oracle	10-best	13,759K	11.24	52.85	10.66	49.81	10.85	48.41
sentence-BLEU		14,587K	11.10	51.17	10.82	49.97	10.86	47.04

by 10-oracle and 10-best list. When decoding, a 1000-best list is generated to achieve better oracle translations. The training took nearly 1 day using 8 cores of Opteron. The translation quality is evaluated by case-sensitive NIST (Doddington, 2002) and BLEU (Papineni et al., 2002)[2]. The table also shows the number of active features in which non-zero values were assigned as weights. The addition of prefix/suffix tokens greatly increased the number of active features. The setting severely overfit to the development data, and therefore resulted in worse results in open tests. The word class[3] with surface form avoided the overfitting problem. The digit sequence normalization provides a similar generalization capability despite of the moderate increase in the active feature size. By including all token types, we achieved better NIST/BLEU scores for the 2004 and 2005 test sets. This set of experiments indicates that a token normalization is useful especially trained on a small data.

Second, we used all the normalized token types, but incrementally added structural features in Table 2. Target bigram features account for only the fluency of the target side without considering the source/target correspondence. Therefore, the inclusion of target bigram features clearly overfit to the development data. The problem is resolved by adding insertion features which can take into account an agreement with the source side that is not directly captured by word pair features. Hierarchical features are somewhat effective in the 2005 test set by considering the dependency structure of the source side.

Finally, we compared our online training algorithm with sparse features with a baseline system in Table 3. The baseline hierarchical phrase-based system is trained using standard max-BLEU training (MERT) without sparse features (Och, 2003). Table 3 shows the results obtained by varying the m-oracle and k-best size ($k, m = 1, 10$) using all structural features and all token types. We also experimented sentence-wise BLEU as an objective function constrained by 10-oracle and 10-best list. Even the 1-oracle 1-best configuration achieved significant improvements over the baseline system. The use of a larger k-best list further optimizes to the development set, but at the cost of degraded translation quality in the 2004 test set. The larger m-oracle size seems to be harmful if coupled with the 1-best list. As indicated by the reduced active feature size, 1-best translation seems to be updated toward worse translations in 10-oracles that are "close" in terms of features. We achieved significant improvements

[2]We used the tool available at `http://www.nist.gov/speech/tests/mt/`

[3]We induced 50 classes each for English and Arabic.

Table 4: Two-fold cross validation experiments.

	closed test		open test	
	NIST	BLEU [%]	NIST	BLEU [%]
baseline	10.71	44.79	10.68	44.44
online	11.58	53.42	10.90	47.64

when the k-best list size was also increased. The use of sentence-wise BLEU as an objective provides almost no improvement in the 2005 test set, but is comparable for the 2004 test set.

As observed in three experiments, the 2004/2005 test sets behaved differently, probably because of the domain mismatch. Thus, we conducted a two-fold cross validation using the 2003/2004/2005 test sets to observe the effect of optimization as shown in Table 4[4]. The MERT baseline system performed similarly both in closed and open tests. Our online large-margin training with 10-oracle and 10-best constraints and the approximated BLEU loss function significantly outperformed the baseline system in the open test. The development data is almost doubled in this setting. The MERT approach seems to be confused with the slightly larger data and with the mixed domains from different epochs.

6 Discussion

In this work, the translation model consisting of millions of features are successfully integrated. In order to avoid poor overfitting, features are limited to word-based features, but are designed to reflect the structures inside hierarchical phrases. One of the benefit of MIRA is its flexibility. We may include as many constraints as possible, like m-oracle constraints in our experiments. Although we described experiments on the hierarchical phrase-based translation, the online training algorithm is applicable to any translation systems, such as phrase-based translations and syntax-based translations.

Online discriminative training has already been studied by Tillmann and Zhang (2006) and Liang et al. (2006). In their approach, training was performed on a large corpus using the sparse features of phrase translation pairs, target n-grams and/or bag-of-word pairs inside phrases. In Tillmann and Zhang (2006), k-best list generation is approximated by a step-by-step one-best merging method that separates the decoding and training steps. The weight vector update scheme is very similar to MIRA but based on a convex loss function. Our method directly employs the k-best list generated by the fast decoding method (Watanabe et al., 2006b) at every iteration. One of the benefits is that we avoid the rather expensive cost of merging the k-best list especially when handling millions of features.

Liang et al. (2006) employed an averaged perceptron algorithm. They decoded each training instance and performed a perceptron update to the weight vector. An incorrect translation was updated toward an oracle translation found in a k-best list, but discarded potentially better translations in the past iterations.

An experiment has been undertaken using a small development set together with sparse features for the reranking of a k-best translation (Watanabe et al., 2006a). They relied on a variant of a voted perceptron, and achieved significant improvements. However, their work was limited to reranking, thus the improvement was relative to the performance of the baseline system, whether or not there was a good translation in a list. In our work, the sparse features are directly integrated into the DP-based search.

The design of the sparse features was inspired by Zens and Ney (2006). They exploited the word alignment structure inside the phrase translation pairs for discriminatively training a reordering model in their phrase-based translation. The reordering model simply classifies whether to perform monotone decoding or not. The trained model is treated as a single feature function integrated in Eq. 1. Our approach differs in that each sparse feature is individually integrated in Eq. 1.

7 Conclusion

We exploited a large number of binary features for statistical machine translation. The model was trained on a small development set. The optimization was carried out by MIRA, which is an online version of the large-margin training algorithm. Millions of sparse features are intuitively considered prone to overfitting, especially when trained on a small development set. However, our algorithm with

[4]We split data by document, not by sentence.

millions of features achieved very significant improvements over a conventional method with a small number of features. This result indicates that we can easily experiment many alternative features even with a small data set, but we believe that our approach can scale well to a larger data set for further improved performance. Future work involves scaling up to larger data and more features.

Acknowledgements

We would like to thank reviewers and our colleagues for useful comment and discussion.

References

Srinivas Bangalore, Patrick Haffner, and Stephan Kanthak. 2006. Sequence classification for machine translation. In *Proc. of Interspeech 2006*, pages 1157–1160, Pittsburgh.

Oliver Bender, Richard Zens, Evgeny Matusov, and Hermann Ney. 2004. Alignment templates: the RWTH SMT system". In *Proc. of IWSLT 2004*, pages 79–84, Kyoto, Japan.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL 2005*, pages 263–270, Ann Arbor, Michigan, June.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, March.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *In Proc. ARPA Workshop on Human Language Technology*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. of COLING/ACL 2006*, pages 961–968, Sydney, Australia, July.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of NAACL 2003*, pages 48–54, Edmonton, Canada.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proc. of COLING/ACL 2006*, pages 761–768, Sydney, Australia, July.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proc. of ACL 2005*, pages 91–98, Ann Arbor, Michigan, June.

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

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL 2003*, pages 160–167, Sapporo, Japan, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL 2002*, pages 311–318, Philadelphia, Pennsylvania.

Brian Roark, Murat Saraclar, Michael Collins, and Mark Johnson. 2004. Discriminative language modeling with conditional random fields and the perceptron algorithm. In *Proc. of ACL 2004*, pages 47–54, Barcelona, Spain, July.

Nobuyuki Shimizu and Andrew Haas. 2006. Exact decoding for jointly labeling and chunking sequences. In *Proc. of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 763–770, Sydney, Australia, July.

Ben Taskar, Dan Klein, Mike Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *Proc. of EMNLP 2004*, pages 1–8, Barcelona, Spain, July.

Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical MT. In *Proc. of COLING/ACL 2006*, pages 721–728, Sydney, Australia, July.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2006a. NTT Statistical Machine Translation for IWSLT 2006. In *Proc. of IWSLT 2006*, pages 95–102, Kyoto, Japan.

Taro Watanabe, Hajime Tsukada, and Hideki Isozaki. 2006b. Left-to-right target generation for hierarchical phrase-based translation. In *Proc. of COLING/ACL 2006*, pages 777–784, Sydney, Australia, July.

Dekai Wu and Hongsing Wong. 1998. Machine translation with a stochastic grammatical channel. In *Proc. of COLING 98*, pages 1408–1415, Montreal, Quebec, Canada.

Richard Zens and Hermann Ney. 2006. Discriminative reordering models for statistical machine translation. In *Proc. of WSMT 2006*, pages 55–63, New York City, June.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proc. of WSMT 2006*, pages 138–141, New York City, June.