Sample Selection for Large-scale MT Discriminative Training

Yuan Cao **Sanjeev Khudanpur**
Center for Language and Speech Processing
Johns Hopkins University
Baltimore, MD 21210, USA
{yuan.cao, khudanpur}@jhu.edu

Abstract

Discriminative training for MT usually involves numerous features and requires large-scale training set to reach reliable parameter estimation. Other than using the expensive human-labeled parallel corpora for training, semi-supervised methods have been proposed to generate huge amount of "hallucinated" data which relieves the data sparsity problem. However the large training set contains both good samples which are suitable for training and bad ones harmful to the training. How to select training samples from vast amount of data can greatly affect the training performance. In this paper we propose a method for selecting samples that are most suitable for discriminative training according to a criterion measuring the dataset quality. Our experimental results show that by adding samples to the training set selectively, we are able to exceed the performance of system trained with the same amount of samples selected randomly.

1 Introduction

Discriminative training methods have been successfully applied to MT tasks(Och, 2003; Shen, 2004; Liang, 2006; Li, 2008). Compared with generative training methods, discriminative training can easily incorporate task-specific knowledge represented by features, and it aims to directly minimize the specific task error instead of maximizing the likelihood. To take the advantage of discriminative training methods, people developed rich feature sets which usually include thousands or even millions of features. In order to reach reliable parameter estimations for so many features, discriminative training for MT must be scaled to large amount of training data. (Li, 2008) proposed a scheme for large-scale discriminative training by using the parallel corpus directly, however such human-labeled parallel corpora are usually very expensive to acquire, hence later on (Li, 2010; Li, 2011) proposed semi-supervised methods to "hallucinate" training data for discrminative training. Those semi-supervised methods can in principle generate almost umlimited "parallel" training data from a target-side monolingual corpus, thus become a strong component in large-scale MT discriminative training.

These huge training data sets contain both good samples which are suitable for discriminative training and noisy samples which can potentially undermine the training performance. Although large-scale training entails high-volume data, many samples are valueless for training and blindly increasing the size of the dataset does not necessarily lead to better performance. So how should we use these datasets wisely? Which samples should be prioritized in training? Solving this problem is especially meaningful when we use semi-supervised methods, which can generate almost unlimited training samples but we probably do not want to use all of them. In this paper we address this problem by selecting samples according to a criterion measuring the training data quality, instead of randomly adding samples to the training data pool.

The rest of the paper is organized as the following: Section 2 overviews work related to ours presented in this paper; Section 3 introduces semi-supervised discriminative training methods, as we experi-

mented both semi-supervised and supervised large-scale training; Section 4 describes our proposed sample selection method and Section 5 presents our experimental results.

2 Related Work

The work presented in this paper is similar to active learning (Settles, 2008; Guo, 2008; Hoi, 2006; Hoi, 2008; Monteleoni, 2007). Since human-labeled data is usually expensive to acquire, active learning aims to select unlabeled data that is most informative for manual labeling while still reaching or exceeding performance given by much more randomly selected and labeled data. Active learning has been applied to NLP problems such as information extraction (Thompson, 1999), parsing (Tang, 2002), as well as machine translation (Haffari, 2009a; Haffari, 2009b; Ananthakrishnan, 2010).

Haffari (2009a) for the first time proposed an active learning framework for SMT. The method proposed by them is to build a bilingual training corpus incrementally by labeling only those samples selected according to the most informative phrases and n-grams, confidence of translations etc. Haffari (2009b) discussed the active learning task of adding new language to an existing multilingual parallel text and improving the MT system quality. Ananthakrishnan (2010) introduced a way of selecting monolingual data for labeling by training a classifier that predicts if a sentence will be incorrectly translated.

Our work differs from the existing active learning techiniques applied to MT in that we do not aim at selecting monolingual data to reduce the human labeling effort, but given a large-scale training data(either supervised or semi-supervised), selecting samples that are most conducive to discriminative training.

3 Semi-supervised Discriminative Training

Discriminative training aims to directly resolve the ambiguities between confusable outputs in applications like ASR and MT(Roark, 2004; Roark, 2007; Shen, 2004; Li, 2008). Given a set of training samples$(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1 \dots n$ where $\mathcal{X}$ is the input set and $\mathcal{Y}$ is the output set(for example in MT, $\mathcal{X}$ is a set of source language sentences and $\mathcal{Y}$ a set of translations in the target language), then under the log-linear model framework, the output of the model is

$$\hat{y}(x) = \underset{y \in \mathbf{GEN}(x)}{\arg\max} \frac{\exp\{\mathbf{\Phi}(x,y)\mathbf{\Theta}\}}{Z(x, \mathbf{\Theta})} \qquad (1)$$

in which $\mathbf{GEN}(x)$ is a function enumerating the candidates for an input x(in other words the confusion set of the true output y. In MT, this can be the n-best translation list for a given source sentence), $\mathbf{\Phi}(x, y)$ is the feature vector associated with the input-output pair (x, y), and $Z(x, \mathbf{\Theta})$ is a normalization term. The goal of discriminative training is to find a parameter vector $\mathbf{\Theta}$ that minimizes the number of times $\hat{y}(x_i) \neq y_i$, or at least assigning more probability mass to the correct outputs.

In practice, as mentioned in Section 1, it is usually expensive to acquire large number of labeled training samples $(x_i, y_i), i = 1 \dots n$. This has led to the recent development of semi-supervised training methods that hallucinate "labeled" training set by using samples y_i from the target side only. We briefly discribe these methods as the following.

The first method, proposed in Li (2011), is a "round-trip"-based method. Given a monolingual corpus on the target language side, plus an existing reverse translation system (from the target to source language) trained from a parallel dataset with limited size, we translate the monolingual corpus into the source language, and treat the automatic translations as approximations to the missing inputs to the bilingual corpus. We can then use these hallucinated input-output pairs for discriminative training.

The second method, proposed in (Bannard, 2005; Madnani, 2007; Li, 2010), is a "confusion-grammar"-based method. The idea is to derive a synchronous monolingual grammar rule set from a synchronous bilingual grammar rule set. For example, if we have two synchronous bilingual grammar rules

$$X \rightarrow < f, e_1 > \qquad X \rightarrow < f, e_2 >$$

where X is the non-terminal, f and e_1, e_2 are the source and target side strings respectively, then we can derive the following target side monolingual grammar rules by using f as a pivot:

$$X \rightarrow < e_1, e_1 > \qquad X \rightarrow < e_1, e_2 >$$
$$X \rightarrow < e_2, e_1 > \qquad X \rightarrow < e_2, e_2 >$$

Once we derive the set of monolingual grammar rules, a target-to-target translation system can be built to translate the monolingual corpus thus generating the confusion sets of the true outputs. Of course, to derive the monolingual rules, we need an existing parallel corpus to extract the set of bilingual rules.

It can be seen that both semi-supervised training methods require only the target side samples $y \in \mathcal{Y}$, and a set of confusions $\mathbf{CON}(y)$ can be generated to mimic the true confusions $\mathbf{GEN}(x)$. We then make use of the $\mathbf{CON}(y)$ together with the true outputs y for discriminative raining.

In our work presented in this paper, we use the n-best lists translated from the hallucinated inputs by the baseline system as the confusion set $\mathbf{CON}(y)$, and we use the perceptron algorithm (Collins, 2002) for n-best list re-ranking, similar to the framework proposed in (Li, 2008).

4 Selective Sampling for Training

As mentioned in section 1, facing vast amounts of data we want to prioritize good samples and discard those which are harmful to discriminative training. In this section, we propose a criterion for selecting samples which makes the training more effective.

Discriminative training essentially aims to reward features indicating good translations and penalize those indicating bad ones. A good training set therefore should meet the following conditions:

1. High corpus-level oracle score: the best candidates (with respect to a metric) should be good enough, otherwise the feature weights would be updated only towards inferior translations far from the golden standards (references), making the weights powerless in predicting real fluent translations.

2. Large contrast between oracle and non-oracle candidate scores: if there is only a narrow gap between oracle and non-oracle translation quality, then it would be hard to identify features indicating real good translations and those indicating bad ones, and the discriminating ability of weights trained in this environment would be weakened.

3. Good and bad candidates should be easily separated by the learning machine: each candidate is represented mathematically by its feature vector. If we treat good and bad candidates as two classes, it is desirable that the two classes in the feature space be as distant as possible, so that the data is more separable and feature weights in a log-linear model (representing a hyperplane in the feature space) can be found more easily by linear learning algorithms which is most widely used in discriminative training.

In other words, condition 1 and 2 indicate that the training results should be enough meaningful, condition 3 indicates how easy it is for the learning machine to reach the goal. Considering the three conditions given above, we propose a quality measurement of a training set $\mathcal{D}$ as the following:

$$Q(\mathcal{D}) = \frac{(N-1)M(\mathcal{R}_1)^2}{\sum_{i\neq 1} M(\mathcal{R}_i)} J(\mathcal{D}) \tag{2}$$

Here N is the size of the n-best list, $\mathcal{R}_i$ is the set of all candidates with rank i (according to the task metric, e.g. BLEU) in the n-best lists, so that $\mathcal{R}_1$ is the set of oracle-best hypotheses, $M(\mathcal{R}_i)$ gives the corpus-level score[1] of $\mathcal{R}_i$.

$J(\mathcal{D})$ is a class separability measure defined as [2].

$$J(\mathcal{D}) = \frac{tr\{S_b\}}{tr\{S_w\}} \tag{3}$$

in which S_b and S_w are the between-class and within-class scatter matrices (Duda, 2000) corresponding to the training set $\mathcal{D}$. For all the n-best lists in $\mathcal{D}$, we treat the top $p\%$-ranking (according to the metric score) candidates as class 1, and the rest as class 2. S_b and S_w are defined as

$$\begin{aligned} S_b &= \sum_{i=1}^{2} (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T \\ S_w &= \sum_{i=1}^{2} \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x}\in\mathcal{C}_i} (\mathbf{x} - \mathbf{m_i})(\mathbf{x} - \mathbf{m_i})^T \end{aligned} \tag{4}$$

[1] The criterion given by (2) assumes to use a metric associating higher score with better translation quality (like BLEU). For metrics like TER (Snover, 2006) which assign lower score to good translations simply replace the factor $\frac{(N-1)M(\mathcal{R}_1)^2}{\sum_{i\neq 1} M(\mathcal{R}_i)}$ with $\frac{\sum_{i\neq 1} M(\mathcal{R}_i)}{(N-1)M(\mathcal{R}_1)^2}$.

[2] There are other forms of definition such as $J(\mathcal{D}) = tr\{S_w^{-1} S_b\}$ and $J(\mathcal{D}) = |S_w^{-1} S_b|$. However these forms entails computing the inverse of matrix which is computationally expensive and numerically unstable when the number of features is large and features are sparse, therefore we use the form given by (3).

in which $\mathbf{m_i}$ is the mean feature vector of class i, $\mathbf{m}$ is the mean feature vector of all candidates, $\mathcal{C}_i$ is the set of candidates belonging to the i^{th} class.

We see in $Q(\mathcal{D})$ that factors $M(\mathcal{R}_1)$, $\frac{\sum_{i\neq 1} M(\mathcal{R}_i)}{N-1}$ and $J(\mathcal{D})$ correspond respectively to condition 1, 2, 3 mentioned above. Now we would like to select subsets from the large hallucinated training set so that $Q(\mathcal{D})$ is as large as possible.

If we hallucinate a raw training set $\mathcal{G}$ (namely the large monolingual corpus from which we select good samples to build up the final training set) with M sentences(that is $|\mathcal{G}| = M$ and $\mathcal{G}$ has 2^M subsets), of course we can select a subset $\mathcal{D} \in 2^{\mathcal{G}}$ having the highest $Q(\mathcal{D})$ score in brute-force manner by computing the score for each subset of $\mathcal{G}$. But this is obviously not manageable when M is large, therefore we hope to compute $Q(\mathcal{D})$ on-the-fly as a new training sample comes and decide if we want to add this sample to the final training set or not.

Suppose we use a metric like BLEU or TER which can be computed by accumulating the sufficient statistics(Omar, 2009)(for example, for BLEU we only need to accumulate the n-gram matches, the source and reference sentence length), then $M(R_i), i \in 1...N$ can be easily updated as a new sample comes and the factor $\frac{(N-1)M(\mathcal{R}_1)^2}{\sum_{i\neq 1} M(\mathcal{R}_i)}$ in $Q(\mathcal{D})$ can also be computed.

The factor $J(\mathcal{D})$, on the other hand, can be updated online as follows. First of all, $tr\{S_b\} = \sum_{i=1}^{2} \langle \mathbf{m}_i - \mathbf{m}, \mathbf{m}_i - \mathbf{m} \rangle$. When a new training sample comes, denote $\mathcal{C}_i^{new}, i = 1, 2$ to be the set of candidates belonging to the i^{th} class in the n-best list, and the updated mean feature vector of class i can be written as

$$\begin{aligned}
\mathbf{m}_i' &= \frac{1}{|\mathcal{C}_i| + |\mathcal{C}_i^{new}|} \sum_{\mathbf{x} \in \mathcal{C}_i \cup \mathcal{C}_i^{new}} \mathbf{x} \\
&= \frac{\sum_{\mathbf{x}\in\mathcal{C}_i} \mathbf{x} + \sum_{\mathbf{x}\in\mathcal{C}_i^{new}} \mathbf{x}}{|\mathcal{C}_i|} \frac{|\mathcal{C}_i|}{|\mathcal{C}_i| + |\mathcal{C}_i^{new}|} \\
&= \left(\mathbf{m}_i + \frac{\sum_{\mathbf{x}\in\mathcal{C}_i^{new}} \mathbf{x}}{|\mathcal{C}_i|} \right) \frac{|\mathcal{C}_i|}{|\mathcal{C}_i| + |\mathcal{C}_i^{new}|}
\end{aligned} \tag{5}$$

Similarly,

$$\mathbf{m}' = \left(\mathbf{m} + \frac{\sum_{\mathbf{x}\in\mathcal{D}^{new}} \mathbf{x}}{|\mathcal{D}|} \right) \frac{|\mathcal{D}|}{|\mathcal{D}| + |\mathcal{D}^{new}|} \tag{6}$$

in which $\mathcal{D}^{new} = \bigcup_{i=1,2} C_i^{new}$, and we have

$$tr\{S_b'\} = \sum_{i=1}^{2} \langle \mathbf{m}_i' - \mathbf{m}', \mathbf{m}_i' - \mathbf{m}' \rangle. \tag{7}$$

Therefore we only need to save $\mathbf{m}$ and $\mathbf{m}_i$ after processing each new sample and $tr\{S_b'\}$ can be updated on-the-fly.

For $tr\{S_w\}$:

$$\begin{aligned}
S_w &= \sum_{i=1}^{2} \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x}\in\mathcal{C}_i} (\mathbf{x} - \mathbf{m_i})(\mathbf{x} - \mathbf{m_i})^T \\
tr\{S_w\} &= \sum_{i=1}^{2} \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x}\in\mathcal{C}_i} \langle \mathbf{x} - \mathbf{m_i}, \mathbf{x} - \mathbf{m_i} \rangle
\end{aligned} \tag{8}$$

and when a new training sample arrives,

$$\begin{aligned}
&tr\{S_w'\} \\
&= \sum_{i=1}^{2} \frac{1}{|\mathcal{C}_i| + |\mathcal{C}_i^{new}|} \sum_{\mathbf{x}\in\mathcal{C}_i\cup\mathcal{C}_i^{new}} \langle \mathbf{x} - \mathbf{m_i'}, \mathbf{x} - \mathbf{m_i'} \rangle \\
&= \sum_{i=1}^{2} \frac{1}{|\mathcal{C}_i'|} \left(\sum_{\mathbf{x}\in\mathcal{C}_i} \langle \mathbf{x} - \mathbf{m_i'}, \mathbf{x} - \mathbf{m_i'} \rangle \right. \\
&\qquad \left. + \sum_{\mathbf{x}\in\mathcal{C}_i^{new}} \langle \mathbf{x} - \mathbf{m_i'}, \mathbf{x} - \mathbf{m_i'} \rangle \right)
\end{aligned} \tag{9}$$

Since $\mathbf{m_i'}$ has been computed as shown above, the term $\sum_{\mathbf{x}\in\mathcal{C}_i^{new}} \langle \mathbf{x} - \mathbf{m_i'}, \mathbf{x} - \mathbf{m_i'} \rangle$ can be easily computed, and we denote $A_i = \sum_{\mathbf{x}\in\mathcal{C}_i^{new}} \langle \mathbf{x} - \mathbf{m_i'}, \mathbf{x} - \mathbf{m_i'} \rangle$. On the other hand, the term $\sum_{\mathbf{x}\in\mathcal{C}_i} \langle \mathbf{x} - \mathbf{m_i'}, \mathbf{x} - \mathbf{m_i'} \rangle$ can be computed incrementally as the following. Let

$$\mathbf{\Delta}_i = \mathbf{m_i} - \mathbf{m_i'} = \left(1 - \frac{|\mathcal{C}_i|}{|\mathcal{C}_i'|} \right) \mathbf{m_i} - \frac{\sum_{\mathbf{x}\in\mathcal{C}_i^{new}} \mathbf{x}}{|\mathcal{C}_i'|} \tag{10}$$

so that

$$\begin{aligned}
&\sum_{\mathbf{x}\in\mathcal{C}_i}\langle \mathbf{x}-\mathbf{m}'_{\mathbf{i}}, \mathbf{x}-\mathbf{m}'_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{x}\in\mathcal{C}_i}\langle \mathbf{x}-\mathbf{m}_{\mathbf{i}}+\boldsymbol{\Delta}_i, \mathbf{x}-\mathbf{m}_{\mathbf{i}}+\boldsymbol{\Delta}_i\rangle \\
&= \sum_{\mathbf{x}\in\mathcal{C}_i}(\langle \mathbf{x}-\mathbf{m}_{\mathbf{i}}, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle + \langle \boldsymbol{\Delta}_i, \boldsymbol{\Delta}_i\rangle \\
&\quad +2\langle \boldsymbol{\Delta}_i, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle) \qquad (11)
\end{aligned}$$

Denote $B_i = \langle \boldsymbol{\Delta}_i, \boldsymbol{\Delta}_i\rangle$, which is easy to compute, and let $S_i = \sum_{\mathbf{x}\in\mathcal{C}_i}\langle \mathbf{x}-\mathbf{m}_{\mathbf{i}}, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle$, then

$$\begin{aligned}
tr\{S'_w\} &= \sum_{i=1}^{2}\frac{1}{|\mathcal{C}'_i|}S'_i \\
&= \sum_{i=1}^{2}\frac{1}{|\mathcal{C}'_i|}\left(\sum_{\mathbf{x}\in\mathcal{C}_i}\langle \mathbf{x}-\mathbf{m}_{\mathbf{i}}, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle + \right. \\
&\quad \left. 2\sum_{\mathbf{x}\in\mathcal{C}_i}\langle \boldsymbol{\Delta}_i, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle + A_i + |\mathcal{C}_i|B_i\right) \\
&= \sum_{i=1}^{2}\frac{1}{|\mathcal{C}'_i|}\left(\sum_{\mathbf{x}\in\mathcal{C}_i}\langle \mathbf{x}-\mathbf{m}_{\mathbf{i}}, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle + \right. \\
&\quad \left. 2\langle \boldsymbol{\Delta}_i, \sum_{\mathbf{x}\in\mathcal{C}_i}(\mathbf{x}-\mathbf{m}_{\mathbf{i}})\rangle + A_i + |\mathcal{C}_i|B_i\right) \\
&= \sum_{i=1}^{2}\frac{1}{|\mathcal{C}'_i|}\left(\sum_{\mathbf{x}\in\mathcal{C}_i}\langle \mathbf{x}-\mathbf{m}_{\mathbf{i}}, \mathbf{x}-\mathbf{m}_{\mathbf{i}}\rangle + \right. \\
&\quad \left. 2\langle \boldsymbol{\Delta}_i, \sum_{\mathbf{x}\in\mathcal{C}_i}\mathbf{x}-|\mathcal{C}_i|\mathbf{m}_{\mathbf{i}})\rangle + A_i + |\mathcal{C}_i|B_i\right) \\
&= \sum_{i=1}^{2}\frac{1}{|\mathcal{C}'_i|}\left(S_i + 2\langle \boldsymbol{\Delta}_i, \sum_{\mathbf{x}\in\mathcal{C}_i}\mathbf{x}-|\mathcal{C}_i|\mathbf{m}_{\mathbf{i}})\rangle + \right. \\
&\quad \left. A_i + |\mathcal{C}_i|B_i\right) \\
&= \sum_{i=1}^{2}\frac{1}{|\mathcal{C}'_i|}\left(S_i + A_i + |\mathcal{C}_i|B_i\right) \qquad (12)
\end{aligned}$$

It can be seen that the term S_i is computed incrementally as $S'_i = S_i + A_i + |\mathcal{C}_i|B_i$, hence $tr\{S'_w\}$ can be easily computed. Together with $tr\{S'_b\}$, we get $J(\mathcal{D}')$.

Now that $Q(\mathcal{D})$ can be updated online, it is natural to consider adding samples greedily to the training set by selecting the sample which gives the highest $Q(\mathcal{D}')$ score from the remaining sample pool in $\mathcal{G}$. However this is computationally intractable as we need to go through all the remaining samples everytime after updating $\mathcal{D}$, which requires up to $M!$ operations. To make the process more practical, we first extract K samples uniformly from the remaining sample pool, compute $Q(D')$ for each of the samples and pick up the one that gives the highest score and return rest to the remaining sample pool. In other words, we limit our search space to K samples.

5 Experimental Results

In this section we report our experimental results on the NIST Urdu-English and Chinese-English MT tasks. We used the open source translation software Joshua (Li, 2009) which implemented the Hiero translation model (Chiang, 2007), and used BLEU as the evaluation metric. We experimented on both the human-labeled parallel corpus(supervised data) and the "hallucinated" parallel corpus(semi-supervised data) created by the "round-trip" method introduced in section 3.

5.1 Datasets

For Urdu-English translation task, the bilingual parallel corpus we used was distributed as part of the 2008 NIST Open Machine Translation Evaluation Workshop, and the corpus contains 88.1K sentences. For test and baseline system tuning set, we split the NIST MT-08 test set into two portions. The test set contains 883 sentences, and the tuning set contains 981 sentences. A 5-gram LM is trained from the English side of the parallel text only with Kneser-Ney smoothing.

For Chinese-English translation task, the bilingual parallel corpus contains about 1M sentences sub-sampled from the corpora distributed by LDC for the NIST MT evaluation using the sampling module in Joshua. A 5-gram LM is trained on the English side of the bilingual text and the English Gigaword corpus (LDC2007T07) with Kneser-Ney smoothing.. We used MT03 set (919 sentences) for baseline system tuning, MT04 (1788 sentences) and

MT05 (1082 sentences) for testing.

5.2 Feature selection

We used two types of features as proposed by (Li, 2011):

1. Regular Hiero features: including a 5-gram LM feature, three baseline translation model features and a word penalty feature. The baseline system makes use of the regular features only.

2. Target-rule bigram discriminative features: for each bilingual grammar rule, extract the bigram POS over the target-side symbols. For example, if the target-side rule is "X1 on X2" where X1 and X2 are non-terminals, then the feature extracted is "X1 PREP", "PREP X2", since PREP is the dominant POS of the non-terminal "on". We extracted about 800 and 1000 most frequent discriminative features for Urdu-English and Chinese-English tasks respectively.

5.3 Training procedure

We followed the training procedure outlined below:

1. Build the forward and reverse baseline systems with the translation model trained from the parallel corpus, and the regular features weights are tuned on the tuning set. Our baseline test set scores are given by the forward baseline system built in this step, and the reverse system is used to hallucinate the missing inputs for semi-supervised discriminative training, according to the "round-trip" method. For supervised method, the reverse system is not used.

   Instead of using MERT (Och, 2003) to tune the baseline system feature weights, we made use of "pairwise ranking optimization" (PRO) proposed by Hopkins (2011) for parameter tuning, as this method is shown to reach comparable performance as MERT but with much less variance and more reliable.

2. For semi-supervised method, randomly choose M sentences from only the English side of the parallel corpus as the monolingual data[3]. Use the "round-trip" method to generate the n-best lists as the confusion set of the English sentences. For supervised method, select M sentences from the source side and generate the n-best lists directly. Now we have the raw training set $\mathcal{G}$[4]. In our experiments we set $M = 10000$.

3. Using method introduced in section 4 to select L samples from $\mathcal{G}$ as the actual discriminative training set. In our experiments we tried $L = 1000, 2000, \ldots, 10000$, and we set $p = 15$, $K = 200$.

4. Train the discriminative feature weights using the average perceptron algorithm.

5. Similar to the methodology used in (Li, 2010), fix the discriminative feature weights and treat the discriminative feature score as one additional feature. Together with the five baseline features, tune the six feature weights using PRO. This gives us the final MT system.

5.4 Results

Our main results are presented in Figure 1. We compare the performance on the test sets given by the baseline system without discriminative training, system with discriminative training by random and selective sampling. The semi-supervised training results are shown in the left column of Figure 1, and supervised training results are shown in the right column. It can be observed that:

1. With the same amount of training data, selective sampling outperforms random sampling in most cases. Given the same raw training set containing 10K samples, the best scores selective sampling can reach are all higher than random sampling. In the Urdu-English case, with random sampling the discriminative training was not very efficient and actually some results were even worse than the baseline system. However selective sampling improved the system performance and raised most scores above baseline.

[3] Of course we could have used the entire English side as the monolingual data, but with limited computing resource we sampled a subset of it just for experimental purpose.

[4] Notice that the translation model is also trained from the parallel corpus. To avoid overfitting on the training data we used the complementary set of $\mathcal{G}$ to build the translation model for the forward translation system to decode the hallucinated inputs (so it is different from the baseline forward system in step 1), as was done by Li (2008).

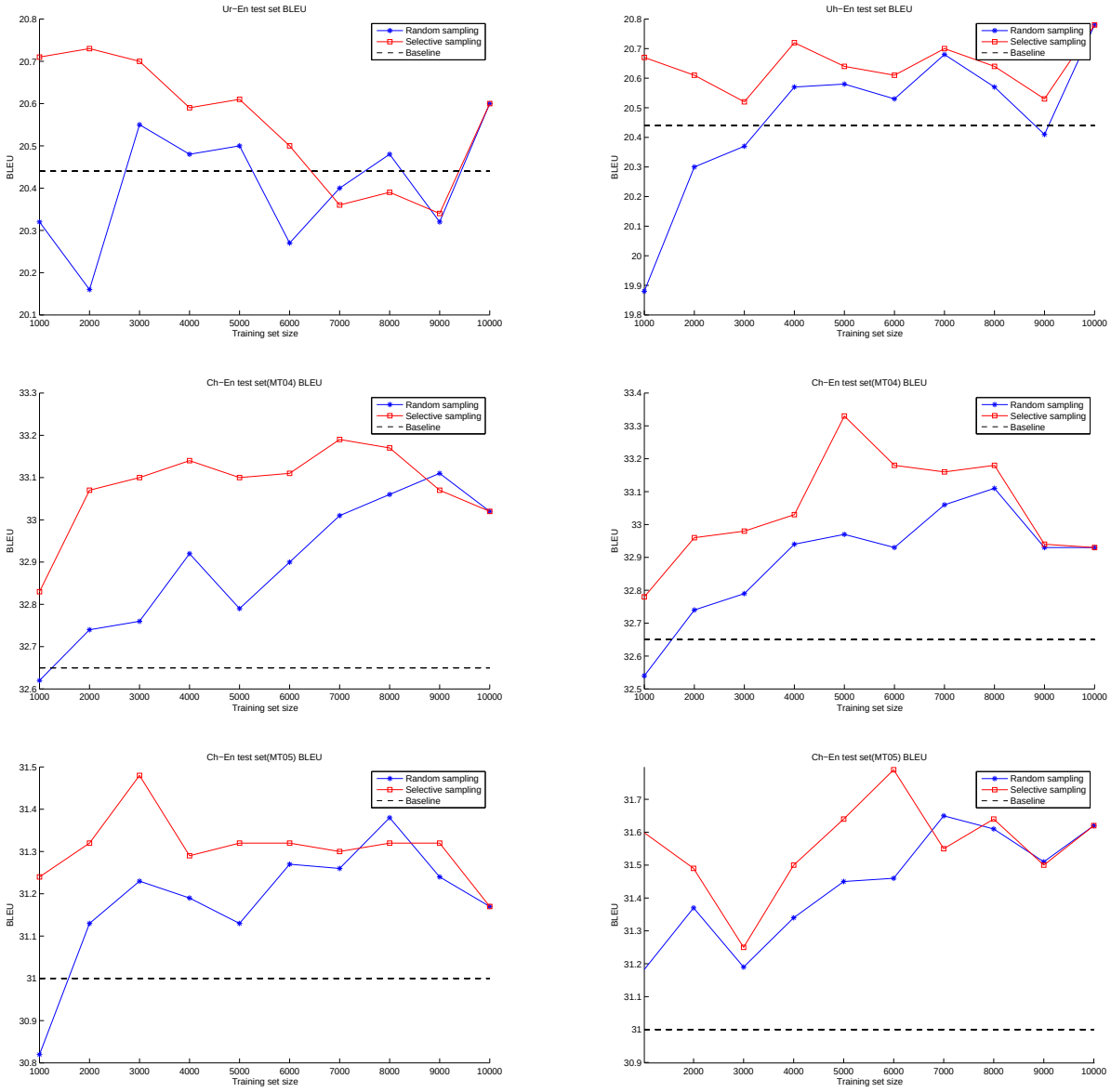

Figure 1: Comparison between BLEU scores on the test set given by the baseline system, system trained with random and selective sampling. Left column: semi-supervised discriminative LM training; Right column: supervised discriminative LM training. From top to bottom: Urdu-English test set, Chinese-English test set(MT04), Chinese-English test set(MT05).

2. Compared with random sampling, system performance is improved more significantly when less training samples are selected, since the sampling strategy will first add good samples to training and delay the entrance of bad samples.

3. The curves of random and selective sampling always merge at the end since at that point the training sets in both cases are the identical.

An interesting observation is that the performance given by semi-supervised training is comparable with supervised training. That is to say, although the surface form of the sentences generated in semi-supervised cases may not be as fluent as human-labeled sentences, the mathematical representation of the sentences are valuable for training.

Figure 2 shows the BLEU gain of the 1-best translations in hallucinated training set after perceptron re-ranking(namely the difference between the 1-best BLEU after and before re-ranking). It can be seen that selective sampling increased the BLEU gain on the training set significantly, since we selected samples having large distances between positive and negative classes so that classification is made easier for perceptron. We observed similar trend in supervised cases, but omitted the plots due to limited space.

Figure 3 compares the $Q(\mathcal{D})$ score as the training set size increases between random and selective sampling in semi-supervised training. While the scores of random sampling remain on a low level(usually to the order of 10^{-4} which cannot be plotted precisely in the figure) for all sizes of training sets, scores of selective sampling are much higher(the average scores of random and selective sampling are 1.5×10^{-4} and 0.039 for Urdu-English, 1.4×10^{-4} and 0.035 for Chinese-English). The selective sampling procedure first adds samples with higher $Q(\mathcal{D})$ scores to the training set, and the score decreases monotonically as the training set size increases. Although we limited our search space to only $K = 200$ samples, the sampling procedure is still effective in preventing the $Q(\mathcal{D})$ score from dropping too fast. Again the plot for supervised case with similar trend is omitted here.

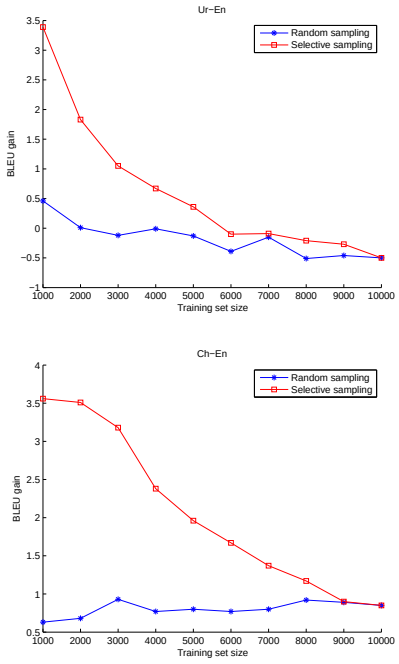

Figure 2: Comparison between BLEU score gains of the 1-best translations from the randomly and selectively sampled training sets.

6 Conclusions

In this paper we addressed the problem of how to make use of the training samples wisely in large-scale MT discriminative training. We presented a method of adding samples to the training set selectively to best fit the discriminative training scenario. The method defines a criterion that measures the suitability of a dataset for training, which can be computed incrementally for efficient sample selection. Experimental results showed consistently that via selective sampling, we are able to exceed the performance given by the system trained with the same amount of data sampled randomly, and the best score obtained is beyond the reach of random sampling.

References

Sankaranarayanan Ananthakrishnan, Rohit Prasad, David Stallard, Prem Natarajan. 2010. *Discriminative Sample Selection for Statistical Machine Translation*. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing(EMNLP), pp.626-635.

Colin Bannard, Chris Callison-Burch. 2005. *Paraphras-*

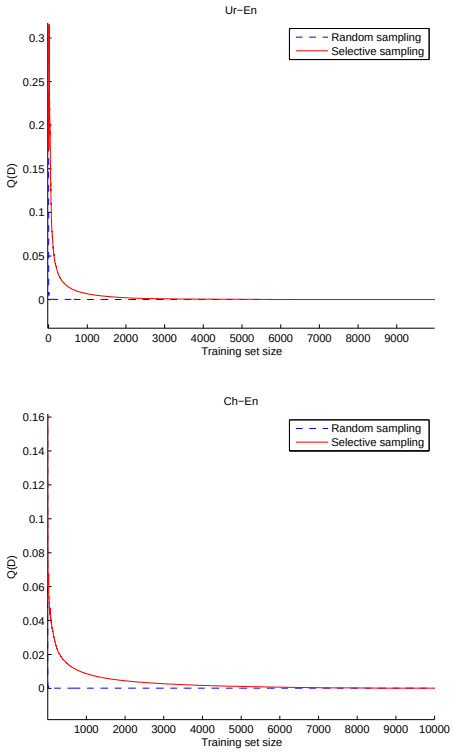

Figure 3: $Q(\mathcal{D})$ score comparison between the training sets sampled randomly and selectively(scores from the middle of the curves cannot be plotted precisely due to small dynamic ranges).

ing with bilingual parallel corpora. In Proceedings of the Association for Computational Linguistics (ACL).

David Chiang 2007. *Hierarchical phrase-based translation*. Computational Linguistics, pp.201-228.

Michael Collins. 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.

Richard O. Duda, Peter E. Hart, David G. Stork. 2000. *Pattern Classification*. Wiley-Interscience.

Yuhong Guo, Dale Schuurmansdepartment. 2008. *Discriminative Batch Mode Active Learning*. In Advances in Neural Information Processing Systems(NIPS), No 20, pp.593-600. MIT Press, Cambridge, MA.

Gholamreza Haffari, Maxim Roy, Anoop Sarkar. 2009. *Active Learning for Statistical Phrase-based Machine Translation*. In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics(NAACL), pp.415-423.

Gholamreza Haffari, Anoop Sarkar. 2009. *Active Learning for Multilingual Statistical Machine Translation*. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp.181-189.

Steven C. H. Hoi, Rong Jin, Jianke Zhu, Michael R. Lyu. 2006. *Batch mode active learning and its application to medical image classification*. In Proceedings of the 23rd International Conference on Machine Learning(ICML), pp.417424, Morgan Kaufman.

Steven C.H. Hoi and Rong Jin and Jianke Zhu and Michael R. Lyu. 2008. *Semi-Supervised SVM Batch Mode Active Learning for Image Retrieval*. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.24-26.

Mark Hopkins, Jonathan May. 2011. *Tuning as Reranking*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP), pp.1352-1362.

Zhifei Li, Jason Eisner, ZiyuanWang, Sanjeev Khudanpur, Brian Roark. 2011. *Minimum Imputed Risk: Unsupervised Discriminative Training for Machine Translation*. In Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP), pp.920-929.

Zhifei Li, Sanjeev Khudanpur. 2008. *Large-scale Discriminative n-gram Language Models for Statistical Machine Translation*. In Proceedings of AMTA.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, Omar. Zaidan. 2009.

Joshua: An open source toolkit for parsing-based machine translation. In Proceedings of the fourth Workshop on Statistical Machine Translation, pp.26-30.

Zhifei Li, Ziyuan Wang, Sanjeev Khudanpur. 2010. *Unsupervised Discriminative Language Model Training for Machine Translation using Simulated Confusion Sets*. In Proceedings of COLING.

Percy Liang, Alexandre Bouchard-Cote, Dan Klein, Ben Taskar 2006. *An End-to-End Discriminative Approach to Machine Translation*. In Proceedings of the 44th Annual Meeting on Association for Computational Linguistics(ACL)

Nitin Madnani, Necip Fazil Ayan, Philip Resnik, Bonnie J. Dorr. 2007. *Using Paraphrases for Parameter Tuning in Statistical Machine Translation*. In Proceedings of the Second Workshop on Statistical Machine Translation, pp.120-127.

Claire Monteleoni, Matti Kaariainen 2007. *Practical Online Active Learning for Classification*. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1-8.

Franz Och. 2003. *Minimum error rate training in statistical machine translation*. In Proceedings of the Association for Computational Linguistics (ACL).

Omar F. Zaidan. 2009. *Z-MERT: A Fully Configurable Open Source Tool for Minimum Error Rate Training of Machine Translation Systems*. The Prague Bulletin of Mathematical Linguistics, No 91, pp.79-88.

Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu 2002. *BLEU:a Method for Automatic Evaluation of Machine Translation*. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics(ACL), pp.311-318.

Brian Roark, Murat Saraclar, Michael Collins, Mark Johnson. 2004. *Discriminative Language Modeling with Conditional Random elds and the Perceptron Algorithm*. In Proceedings of the annual meeting of the Association for Computational Linguistics (ACL).

Brian Roark, Murat Saraclar, Michael Collins. 2007. *Discriminative n-gram Language Modeling*. Computer Speech and Language, 21(2).

Libin Shen, Anoop Sarkar, Franz Joseph Och 2004. *Discriminative Reranking for Machine Translation*. In Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics(HLT-NAACL)

Burr Settles 2010. *Active Learning Literature Survey*. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. *A Study of Translation Edit Rate with Targeted Human Annotation*. In Proceedings of Association for Machine Translation in the Americas(AMTA).

Min Tang, Xiaoqiang Luo, Salim Roukos 2002. *Active Learning for Statistical Natural Language Parsing*. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics(ACL), pp.120-127.

Cynthia A. Thompson, Mary Elaine Califf, Raymond J. Mooney. 1999. *Active Learning for Natural Language Parsing and Information Extraction*. In Proceedings of the 16th International Conference on Machine Learning(ICML), pp.406-414, Morgan Kaufman.