[8th AMTA conference, Hawaii, 21-25 October 2008]

Large-scale Discriminative n-gram Language Models for Statistical Machine Translation

Zhifei Li and **Sanjeev Khudanpur**
Department of Computer Science and Center for Language and Speech Processing
Johns Hopkins University, Baltimore, MD 21218, USA
zhifei.work@gmail.com and khudanpur@jhu.edu

Abstract

We extend discriminative n-gram language modeling techniques originally proposed for automatic speech recognition to a statistical machine translation task. In this context, we propose a novel data selection method that leads to good models using a fraction of the training data. We carry out systematic experiments on several benchmark tests for Chinese to English translation using a hierarchical phrase-based machine translation system, and show that a discriminative language model significantly improves upon a state-of-the-art baseline. The experiments also highlight the benefits of our data selection method.

1 Introduction

Recent research in statistical machine translation (SMT) has made remarkable progress by evolving from word-based translation (Brown et al., 1993), through flat phrase-based translation (Koehn et al., 2003) and hierarchical phrase-based translation (Chiang, 2005; Chiang, 2007), to syntax-based translation (Galley et al., 2006). These systems usually contain three major components: a translation model, a word-reordering model, and a language model. In this paper, we mainly focus on improving the language model (LM).

A language model constitutes a crucial component in many other tasks such as automatic speech recognition, handwriting recognition, optical character recognition, etc. It assigns *a priori* probabilities to word sequences. In general, we expect a low probability for an ungrammatical or implausible word sequence. Normally, a language model is derived from a large corpus of text in the target language via maximum likelihood estimation (MLE), in conjunction with some smoothing (Chen and Goodman, 1998). In particular, the so called n-gram model is particularly effective and has become the dominant LM in most systems. Several attempts have been made, particularly in speech recognition, to improve LMs by appealing to more powerful estimation techniques, e.g., decision trees (Bahl et al., 1989), maximum entropy (Rosenfeld, 1996), neural networks (Bengio et al., 2001), and random forests (Xu and Jelinek, 2004). Attempts have also been made to extend beyond n-gram dependencies by exploiting (hidden) syntax structure (Chelba and Jelinek, 2000) and semantic or topical dependencies (Khudanpur and Wu, 2000). We limit ourselves to n-gram models in this paper, though the ideas presented extend easily to the latter kinds of models as well.

A regular LM obtained through the MLE technique is task-independent and tries to distinguish between likely and unlikely word sequences without considering the actual confusions that may exist in a particular task. This is sub-optimal, as different tasks have different types of confusion. For example, while the main confusion in a speech recognition task is between similar sounding words (e.g., "red" versus "read"), the confusion in a machine translation task is mainly due to multiple word senses or to word order. An LM derived using an explicitly discriminative criterion has the potential to resolve task-specific confusion: its parameters are tuned/adjusted by observing actual confusions in task-dependent outputs. Such discriminative n-gram language modeling has been investigated by Stolcke and Weintraub (1998), Chen et al. (2000), Kuo et al. (2002), and Roark et al. (2007) on various speech recognition tasks. It is scientifically interesting to see whether such techniques lead to improvement in an SMT task, given the substantial task differences. We do so in this paper.

We investigate application of the discriminative n-gram language modeling framework of Roark et al. (2007) to a large-scale SMT task. Our discriminative language model is trained using an averaged perceptron algorithm (Collins, 2002). In our task, there are millions of training examples available, and many of them may not be beneficial due to various reasons including noisy reference translations resulting from automatic sentence-alignment of a document-aligned bilingual text corpus. Moreover, our discriminative model contains millions of features, making standard perceptron training prohibitively expensive. To address these two issues, we propose a novel data selection method that strives to obtain a comparable/better model using only a fraction of the training data. We carry out systematic experiments on a state-of-the-art SMT system (Chiang, 2007) for the Chinese to English translation task. Our results show that a discriminative LM is able to improve over a very strong baseline SMT system. The results also demonstrate the benefits of our data selection method.

2 Discriminative Language Modeling

We begin with the description of a general framework for discriminative language modeling, recapitulating for the sake of completeness the detailed description of these ideas in (Collins, 2002; Roark et al., 2007).

2.1 Global Linear Models

A linear discriminant aims to learn a mapping from an input $x \in X$ to an output $y \in Y$, given

1. training examples (x^i, y^i), $i = 1 \cdots N$,
2. a representation $\Phi : X \times Y \rightarrow R^d$ mapping each possible (x, y) to a feature vector,
3. a function $\mathrm{GEN}(x) \subseteq Y$ that enumerates putative labels for each $x \in X$, and
4. a vector $\alpha \in R^d$ of free parameters.

In SMT, x is a sentence in the source language, $\mathrm{GEN}(x)$ enumerates possible translations of x into the target language, and y is the *desired* translation: either a *reference* translation produced by a bilingual human or the alternative in $\mathrm{GEN}(x)$ that is most similar to such a reference translation. In the latter case, y is often called the *oracle*-best (or simply *oracle*) translation of x.

The components $\mathrm{GEN}(\cdot)$, Φ, and α define a mapping from an input x to an output y^* through

$$y^* = \arg\max_{y \in \mathrm{GEN}(x)} \Phi(x, y) \cdot \alpha, \qquad (1)$$

where $\Phi(x, y) \cdot \alpha = \sum_j \alpha_j \Phi_j(x, y)$, with j indexing the feature dimensions, is the inner product.

Since y is a word-sequence, (1) is called *global* linear model to emphasize that the maximization is jointly over the entire sentence y, not locally over each word/phrase in y (as done in (Zens and Ney, 2006; Chan et al., 2007; Carpuat and Wu, 2007)).

The *learning* task is to obtain the "optimal" parameter vector α from training examples, while the *decoding* task is to search, given an x, for the maximizer y^* of (1). These tasks are discussed next in Section 2.2 and Section 2.3, respectively.

2.2 Parameter Estimation Methods

Given a set of training examples, the choice of α may be guided, for instance, by an explicit criterion such as maximizing, among distributions in an exponential family parameterized by Φ, the conditional log-likelihood of y^i given x^i. Algorithms that determine α in this manner typically operate in *batch* mode—they require processing all the training data (often repeatedly) before arriving at an answer—and require regularization techniques to prevent overfitting, but are amenable to parallelized computing and often exhibit good empirical performance.

On the other hand, *sequential* algorithms such as the linear perceptron (Collins, 2002) operate in *online* mode—processing the training data sequentially to incrementally adjust the parameters—and are not amenable to parallelization, but exhibit faster convergence to parameter values that yield comparable empirical performance, particularly when large amounts of training data are available. In this paper, we use the perceptron algorithm due to its simplicity and suitability to large data settings.

```
Perceptron($x$, GEN($x$), $y$)
1  $\alpha \leftarrow \vec{0}$          ▷ initialize as zero vector
2  for $t \leftarrow 1$ to $T$
3      for $i \leftarrow 1$ to $N$
4          $z^i \leftarrow \arg\max_{z \in \mathrm{GEN}(x^i)} \Phi(x^i, z) \cdot \alpha$
5          if ($z^i \neq y^i$)
6              $\alpha \leftarrow \alpha + \Phi(x^i, y^i) - \Phi(x^i, z^i)$
7  return $\alpha$
```

Figure 1: The Basic Perceptron Algorithm

2.2.1 Averaged Perceptron Algorithm

Figure 1 depicts the perceptron algorithm (Roark et al., 2007). Given a set of training examples, the algorithm *sequentially* iterates over the examples, and adjust the parameter vector α. After iterating over the training data a few times, an *averaged* model, defined as

$$\alpha_{\text{avg}} = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{N}\sum_{i=1}^{N}\alpha_t^i \tag{2}$$

is computed and is used for testing, where α_t^i represents the parameter vector after seeing the i-th example in the t-th iteration, N represents the size of the training set, and T is the number of iterations the perceptron algorithm runs.

2.3 The LM Reranking Framework

The model and training algorithms above are general and can be applied in many natural language and speech processing tasks. To apply them in a specific task, one needs to define an application-specific feature vector $\Phi(x, y)$, and a decoding algorithm that is needed both for perceptron training (see line-4 of Figure 1) and for testing on held-out data. As in (Roark et al., 2007), we use a reranking approach, employing a discriminative LM estimated using the averaged perceptron to rerank the N-best hypotheses produced by a baseline SMT system. Therefore, $\mathrm{GEN}(x)$ in our case is the N-best list for the input x generated by a baseline SMT system. The reranking approach has the advantage of being simple and capable of exploiting non-local features.

```
Rerank-Nbest(GEN($x$))
1  $y^* \leftarrow \mathrm{GEN}(x)[1]$        ▷ baseline 1-best
2  for $y$ in GEN($x$)
3      $S(x, y) \leftarrow \beta\Phi_0(x, y) + \sum_{j \in [1,F]} \alpha_j \Phi_j(x, y)$
4      if $S(x, y) > S(x, y^*)$
5          $y^* \leftarrow y$
6  return $y^*$
```

Figure 2: Discriminative LM reranking of the N-best list $\mathrm{GEN}(x)$ of a source sentence x.

2.3.1 Features used in the Discriminative LM

Each component $\Phi_j(x, y)$ of the feature vector can be any function of the input x and the output y. We define the feature vector specific to our language modeling task as follows.

Baseline Feature: We first define a baseline feature $\Phi_0(x, y)$, to be the score assigned to y by the baseline SMT system. This score itself is often a linear combination of several models, with the relative weights among these models obtained via some minimum error rate training procedure (Och, 2003).

Discriminative n-gram Features: The count of each n-gram in y constitutes a feature. E.g., the first n-gram feature may be,

$$\Phi_1(x, y) = \text{Count of the bigram “}\texttt{the of}\text{” in } y.$$

Normally, the baseline SMT system also employs an n-gram LM, and the baseline score $\Phi_0(x, y)$ is thus not independent of the n-gram features. However, we do not change the parameterization of the n-gram LM used by the SMT system; i.e. we keep $\Phi_0(x, y)$ fixed when carrying out our discriminative LM training.

2.3.2 Reranking as Decoding

Given the feature- and the parameter-vectors, the total score assigned to an output $y \in \mathrm{GEN}(x)$ for a given input x is

$$S(x, y) = \beta\Phi_0(x, y) + \sum_{j \in [1,F]} \alpha_j \Phi_j(x, y)\,, \tag{3}$$

where β is the weight for the baseline feature and F is the number of discriminative n-gram features. To find the optimal weight β for the baseline feature, one could simply treat Φ_0 as a feature in the discriminative language model (DLM) and set the value of β via the perceptron algorithm. This, however, may lead to *under-training* (Sutton et al., 2006) of the discriminative n-gram features in the DLM: the baseline feature is strongly indicative of the *overall* goodness of y for x, relative to any single discriminative n-gram feature which indicates the *local* goodness of y. Therefore, we use a fixed value for β during the training of the DLM, as suggested by Roark et al. (2007).

The decoding task therefore is to search for the y that maximizes $S(x, y)$ in (1). Figure 2 illustrates our language model reranking algorithm: it performs a linear scan of the N-best list and maintains the best variant y^* in terms of $S(x, \cdot)$.

3 Discriminative LM Training for SMT

The discriminative LM reranking framework of Section 2 is quite general and may be applied in many language modeling tasks. In this section, we discuss how this framework is applied in SMT.

The training for our discriminative LM involves five major steps:

1. Train the baseline SMT system(s);
2. Decode all x^i in the discriminative training data using the system(s) trained in Step 1;
3. Find the oracle translation y^i from the N-best list $\text{GEN}(x^i)$ corresponding to each x^i;
4. Select samples for discriminative training;
5. Train the discriminative LM per Figure 1.

Since parallel text is scarce, we use the same data in Steps 1 and 2 through 5 for SMT and discriminative training. One could train a single baseline SMT system on the entire parallel corpus, and decode all the source language sentences using this system. However, this may lead to a significant mismatch during actual test conditions. In particular, the N-best list generated on the SMT *training* data will have better translation quality than on unseen test data. To avoid this pitfall, we partition the parallel text into multiple disjoint sections, and train multiple baseline SMT systems in Step 1. To decode sentences from a particular section in Step 2, we use a baseline system that excluded that section. Unlike Roark et al. (2007), who carry out this *leave-one-out* training for the LM component but not the acoustic models, we do so for both the language and translation models. This is because both the TM and LM in an SMT system are non-parametric and equally susceptible to over-fitting.

In a large-scale task like statistical machine translation, Step 2 is the most time-consuming step, as millions of parallel sentences need to be decoded, each taking several CPU-seconds.

In Step 3, we find the oracle translation in each N-best list by using an automatic sentence-level metric of similarity to the reference translation, specifically BLEU (Papineni et al., 2002).[1] The oracle translation is used to provide supervision during discriminative training.

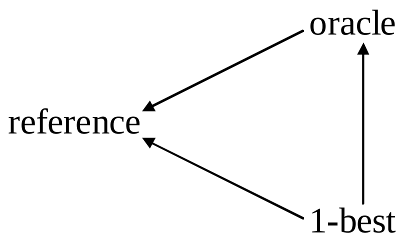
Step 4 is optional, and we discuss its benefits in Section 4. The details of Step 5 have already been discussed in Section 2.

4 Data Selection for Discriminative LMs

After decoding (Step 2) and oracle-finding (Step 3), we can simply present all the training examples (i.e., N-best lists together with the corresponding oracle hypotheses) to the discriminative training algorithm. Alternatively, we can carry out a *data selection* procedure that selects a subset of these examples. Data selection is useful for speeding up the training procedure as most training algorithms require computational time proportional to the size of training data. It also has potential to improve the model quality due to the following three reasons.

1. In general, it is critical to give *correct* supervision to the training algorithm in order for it to learn a proper model. However, the parallel training data is quite noisy. In particular, some pairs of sentences in the parallel data are not adequate translations of each other. This may be due to poor annotation, or because some of the parallel data are created by an automatic sentence-alignment tool, from a corpus that is *manually* aligned only at the *document* level.

[1]When there is no n-gram match (particularly 4-gram) between the hypothesis and reference, a smoothed version of BLEU is used.



$$G(\text{ref, oracle}) > T_1$$
$$G(\text{ref, oracle}) - G(\text{ref, 1-best}) > T_2$$
$$G(\text{oracle, 1-best}) > T_3$$

Figure 3: To be selected for discriminative training, a sample must satisfy the 3 stated conditions.

2. In a discriminative model with millions of features, it is important to make the model as compact as possible. Intuitively, the more training data, the more features will be activated. To help the model learn the most *profitable* features, we should present the training examples where the baseline 1-best can be improved the most, while ignoring or postponing examples with only a marginal gap between the 1-best and the oracle hypothesis.

3. Another reason for data selection is that due to the localness of n-gram features, and to its corrective role, our discriminative LM cannot correct drastic errors made by the baseline system. Therefore, it is helpful to present the training algorithm with only *correctable* training examples while ignoring cases where the translation model is so far off the mark that no reasonable weight given to a corrective LM will suffice.

In light of these observations, we propose the data selection method illustrated in Figure 3, where $G(x, y)$ represents the goodness (e.g., BLEU) of a hypothesis y with respect to a reference x, while T_1, T_2, and T_3 are three configurable thresholds. Clearly, each of the three conditions listed in the figure deals with one issue described above. Specifically, the first condition eliminates badly aligned parallel data; the second condition makes sure the training examples are highly profitable; while the third condition makes sure the examples are correctable.

5 Experimental Results

We report experimental results using the hierarchical machine translation system of Chiang (2007) for Chinese to English translation.

5.1 Training Data

We compile a parallel dataset consisting of various corpora distributed by the Linguistic Data Consortium (LDC) for NIST MT evaluation. The selected subset has about 1M sentence pairs, corresponding to about 28M words in each language. To train the English language model, we use a 130M word subset of the English Gigaword corpus (LDC2007T07) and the English side of the parallel corpora.

5.2 Baseline Systems Training

We partition the parallel text into 30 non-overlapping sections, and train 30 baseline SMT systems. We use the GIZA toolkit (Och and Ney, 2000), a suffix-array architecture (Lopez, 2007), the SRILM toolkit (Stolcke, 2002), and minimum error rate training (MERT) (Och, 2003) to obtain word-alignments, translation models, language models, and the optimal weights for combining these models, respectively. The baseline LM is a *trigram*[2] LM with modified Kneser-Ney smoothing (Chen and Goodman, 1998). The MERT is performed on the NIST MT'03 evaluation data set.

As mentioned in Section 3, we carry out the *leave-one-out* training for both LM and TM. When training an LM, we need to exclude one portion (among the 30 portions) of the English side of the parallel corpora. To do this efficiently, we first get the n-gram counts from the subset of Gigaword and the whole English side of the parallel corpora, then deduct the n-gram counts collected from the portion that we want to exclude, and finally train an LM on the resulting counts. This procedure is done for each of the 30 baseline systems we need to train.

5.3 Data for Discriminative Training

The decoding of the training data (containing about 1M sentences) is the most expensive part in the training. We have developed a fast decoder (Li and

[2]We use a trigram LM because the decoding of the whole training set (about 1M sentences) is too computationally expensive if we use a larger order LM.

Khudanpur, 2008).[3] To decode source language sentences in a section, we use the baseline system trained on the data excluding that section.

For each Chinese sentence, we generate **300** *unique* hypotheses. Note that we decode the SMT training data only *once*.

5.4 Goodness of the Oracle Translations

Figure 4 shows the average BLEU scores on the oracle and 1-best translations for the training data. The discriminative LM will use this gap to learn how to pick better translations from the N-best list. As shown in Figure 4, the longer the sentences, the smaller the BLEU scores. This is expected, as longer sentences are more difficult to translate.

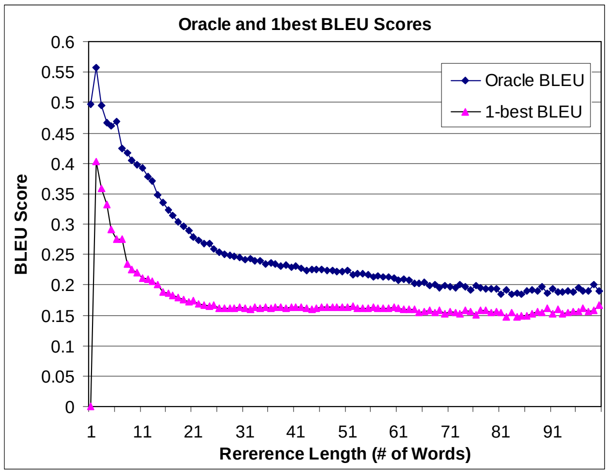


Figure 4: Average BLEU Scores (4-gram BLEU with one reference) for the Oracle and 1-best Hypotheses in the training data.

We also report, in Table 1, the BLEU scores of the oracle translations on three NIST evaluation data sets on which we will later report performance. This table sets an upper bound on improvements possible using the discriminative LM. Note that the NIST MT'03 is used for maximum-BLEU training, and thus the difference between the BLEU scores on the oracle and 1-best is smaller compared with other MT sets.

[3]The decoder is freely available to be downloaded at http://www.cs.jhu.edu/∼zfli/.

Task	1-best	oracle
MT'03	0.350	0.407
MT'04	0.357	0.440
MT'05	0.326	0.412
MT'06	0.283	0.351

Table 1: Oracle and 1-best BLEU Scores (4-gram BLEU with four references) on MT Sets

5.5 Impact of Data Selection

As mentioned in Section 4, there are three configurable thresholds (i.e., T_1, T_2, and T_3) in our data selection algorithm (see Figure 3). Different thresholds lead to selection of different training data for the perceptron training, therefore, we need to train a perceptron model for each different configuration of the thresholds.

Clearly, it is impossible to evaluate all the possible configurations. To carry out experiments efficiently, we first set the default parameters as $T_1 = 0.10$, $T_2 = 0.01$, and $T_3 = 0.20$, leading to the selection of about 630K training sentences from about 1M. Then, we vary one of the thresholds while keeping the other two at the default values. In total, we train 42 perceptron models, one for each configuration of the thresholds.

Figures 5, 6, and 7 report the BLEU scores on the MT'04 test set under the perceptron models trained by varying T_1, T_2, and T_3, respectively. In all the models, we use *discriminative* unigram and bigram features[4]. The number of iterations in the perceptron training, and the baseline weight β, are tuned by maximizing the BLEU score on MT'04. As seen in the figures, there is no telling pattern. This may be due to the instability of the perceptron training algorithm or due to the fact we are unable to capture the complex interaction between the three thresholds.

Notably, however, all the BLEU scores are consistently better than the MT'04 baseline of 0.357. This shows the benefit of data selection as we can train a comparable/better model with much less training data.

[4]As noted by Roark et al. (2007), adding trigram features does not lead to much improvment in the speech recognition task. We observe a similar trend in our machine translation task. Therefore, we only report results with unigram and bigram features here.

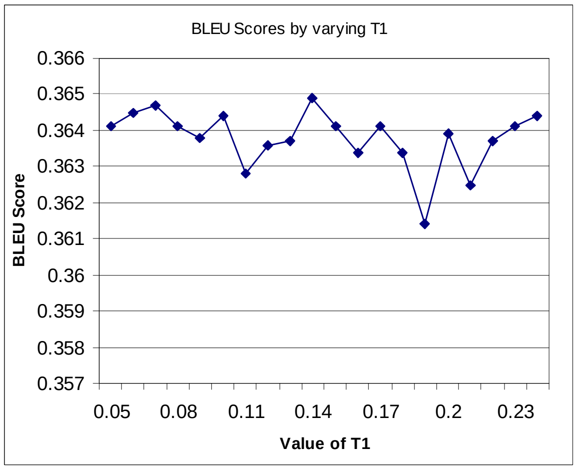


Figure 5: BLEU Scores on MT'04 when varying the value of $T_1 \in [0.05, 0.25]$ with a step size 0.01.

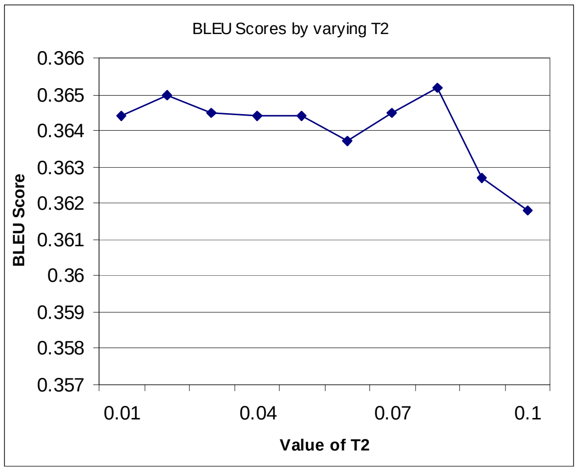


Figure 6: BLEU Scores on MT'04 when varying the value of $T_2 \in [0.01, 0.10]$ with a step size 0.01.

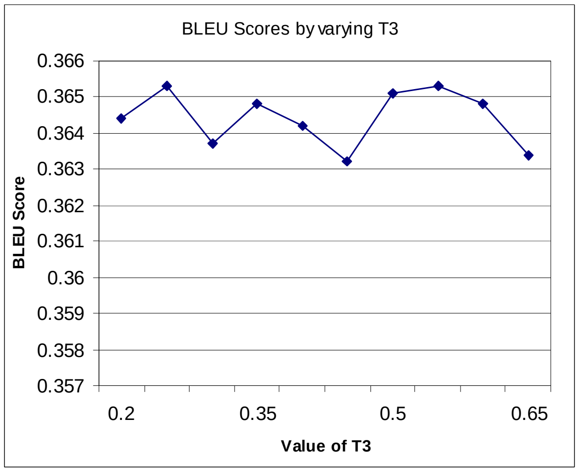


Figure 7: BLEU Scores on MT'04 when varying the value of $T_3 \in [0.20, 0.75]$ with a step size 0.05.

Task	Baseline		Reranking	
	Chiang (2007)	Ours	Full	Selected
MT'04	0.346	0.357	**0.365**	0.365
MT'05	0.318	0.326	0.332	**0.333**
MT'06	NA	0.283	0.292	**0.294**

Table 2: BLEU Scores on MT Test Sets for Discriminative Reranking on a unique N-best of size 300.

5.6 Results on NIST MT Benchmark Tests

In this subsection, we report the results on the test sets by applying a model that is trained under the best configuration learnt from the previous subsection. Specifically, the best configuration is $T_1 = 0.10$, $T_2 = 0.01$, and $T_3 = 0.25$, under which about 610K examples are selected from among about 1M examples in total. The number of n-gram features contained in the discriminative model are about 1.9M and 1.4M, for the case without data-selection and the one with data-selection, respectively. Moreover, the perceptron training obtains optimal performance after the first iteration, though a maximum of three iterations (i.e., $T = 3$ in Figure 1) have been run in our experiments.

The results in Table 2 show that our discriminative LM reranking framework improves the baseline system significantly. The results also show that the data selection procedure can lead to a comparable or better model with less training data. To put our results in perspective, in the second column of Table 2 we include the results reported by Chiang (2007) who uses a training set similar to ours. Clearly, our baseline itself is stronger than the system in Chiang (2007).[5]

To get a sense of what features (e.g., the ones affecting word order versus lexical choice) make the most contributions in improving the MT performance, we present in Table 3 the individual n-gram precisions reported by the BLEU script, and the relative improvement ratio (in terms of percentage). Roughly speaking, the features concerning lexical choice mainly affects unigram precision, while the features concerning word order mainly affects the

[5]The improvement may be due to the difference in training data or due to the fact that our decoder is scalable enough to allow larger search beams.

Task	**System**	**n-gram precision (%)**			
		1	**2**	**3**	**4**
MT'04	**baseline**	81.2	47.5	27.7	16.1
	rerank	81.2	47.9	28.1	16.3
Improvement (%)		0	0.8	1.4	1.2
MT'05	**baseline**	77.8	43.5	24.7	14.2
	rerank	77.7	43.6	25.0	14.6
Improvement (%)		-0.1	0.2	1.2	2.8
MT'06	**baseline**	79.7	42.6	23.3	12.8
	rerank	79.9	43.4	24.0	13.5
Improvement (%)		0.3	1.9	3.0	5.5

Table 3: n-gram Precisions and Relative Improvements on MT Test Sets

higher-order n-gram.[6] As clear from "relative improvement" rows in the table, our discriminative LM mainly improves on the word order, and not so much on lexical choice.

In all the experiments above, we use a tri-gram language model in the baseline system. We made this choice because the decoding of the whole training set (about 1M sentences) is too computationally expensive if we use a larger language model. To verify if the discriminative training is able to improve the MT performance when we have a larger language model, we have carried out several (though not all) of the preceding experiments with two-stage reranking, in which the N-best list is first reranked by a regular *higher-order* n-gram LM, and by the discriminative LM in a second reranking pass. We find that *the gains from the discriminative LM continue to hold up*, and we expect to report more comprehensive results in due time.

6 Related Work

The work most closely related to ours is the discriminative n-gram LM of Roark et al. (2007): they focus on a *speech recognition* task, while we present results for a statistical machine translation task. Additionally, we propose a novel data selection procedure, which not only speeds up the training but also slightly improves the model quality as shown in our experimental results.

In the context of an SMT task, discriminative training methods have been applied both in a *small-scale* setting (i.e, finding optimal weights among a small set of generative models) and in a *large-scale* setting (i.e., training optimal weights for millions of features).

In the *small-scale* setting, the minimum error rate training algorithm (Och, 2003) has become the de facto standard in SMT systems. Smith and Eisner (2006) propose an annealed minimum risk approach, while Zens et al. (2007) give a systematic experimental comparison for different training criteria. Shen et al. (2004) use perceptron-inspired algorithms to tune weights for tens of features, and the model is used to rerank a N-best as we have done here.

Large-scale discriminative training has been attempted by several groups. They differ in the features used, for example, reordering features (Tillmann and Zhang, 2006; Chang and Toutanova, 2007), translation model features (Blunsom et al., 2008), and both translation and language models features (Liang et al., 2006; Watanabe et al., 2007). They also differ in whether using a fixed n-best (e.g., Watanabe et al. (2007)) or running an end-to-end decoding (e.g., (Liang et al., 2006; Blunsom et al., 2008)), at each iteration of training. In the latter case, they resort to a relative weak baseline for computational efficiency. For example, Liang et al. (2006) use a *monotone* translation system, while Blunsom et al. (2008) do not use a language model and they train the baseline system only on sentences whose length is less than 15 words and whose reference is reachable by the decoder. In comparison, in this paper, we focus exclusively on the language model features using an n-best reranking method. Moreover, our improvement is over a full-grown state-of-the-art hierarchical machine translation system for a known difficult task (i.e., translation from Chinese to English).

While all the above discriminative models are *global* (meaning the optimization is over the whole output sequence), it is also possible to decompose the global prediction problem into many *independent local* prediction problems and then train a classifier for each local prediction problem, as done in

[6]One should note that this may not be strictly true. For example, the improvement in the lexical choice may also lead to the improvement in the word order. However, since our discriminative model reranks on a n-best, this effect may not be substantial.

(Zens and Ney, 2006; Chan et al., 2007; Carpuat and Wu, 2007). Compared with a global model, the local model has significantly less computational demand. On the other hand, independent locally optimal prediction may not yield a globally optimal prediction, and thus a global model may be preferred.

7 Conclusions

In this paper, we successfully apply a discriminative n-gram language model to improve a large-scale, competitive statistical machine translation system. We also propose a novel data selection method, through which a comparable/better model can be obtained with much less training data. We carry out systematic experiments on a hierarchical machine translation system (Chiang, 2007) for the Chinese to English translation task. Our results show that the discriminative language model is able to improve over a very strong baseline system.

Acknowledgments

We thank Damianos Karakos for very helpful discussions, and the anonymous reviewers for their constructive comments. This research was partially supported by the Defense Advanced Research Projects Agency's GALE program via Contract No̲ HR0011-06-2-0001.

References

Lalit R. Bahl, Peter F. Brown, Peter V. de Souza, and Robert L. Mercer. 1989. A tree-based statistical language model for natural language speech recognition. *IEEE Trans Acoustics, Speech, and Signal Processing*, 37(7):1001-1008.

Yoshua Bengio, Rejean Ducharme, and Pascal Vincent. 2001. A neural probabilistic language model. *In Advances in Neural Information Processing Systems (NIPS).*

Phil Blunsom, Trevor Cohn and Miles Osborne. 2008. A Discriminative Latent Variable Model for Statistical Machine Translation. *In Proceedings of ACL 2008.*

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263-311.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. *In Proceedings of EMNLP 2007.*

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007 Word sense disambiguation improves statistical machine translation. *In Proceedings of EMNLP 2007.*

Pi-Chuan Chang and Kristina Toutanova. 2007. A Discriminative Syntactic Word Order Model for Machine Translation. *In Proceedings of ACL 2007.*

Ciprian Chelba and Frederick Jelinek. 2000. Structured language modeling. *Computer Speech and Language*, 14(4):283-332.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology.

Zheng Chen, Kai-Fu Lee, and Ming-jing Li. 2000. Discriminative training on language model. *In Proceedings of International Conference on Spoken Language Processing(ICSLP) 2000.*

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. *In Proceedings of ACL 2005.*

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201-228.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. *In Proceedings of EMNLP 2002.*

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 Proceedings of COLING/ACL 2006.*

Sanjeev Khudanpur and Jun Wu. 2000. Maximum Entropy Techniques for Exploiting Syntactic, Semantic and Collocational Dependencies in Language Modeling. *Computer Speech and Language*, 14(4):355-372.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. *In Proceedings of NAACL 2003.*

Hong-Kwang Jeff Kuo, Eric Fosler-Lussier, Hui Jiang, and Chin-Hui Lee. 2002. Discriminative training of language models for speech recognition. *In Proceedings of ICASSP 2002.*

Zhifei Li and Sanjeev Khudanpur. 2008. A Scalable Decoder for Parsing-based Machine Translation with Equivalent Language Model State Maintenance. *In Proceedings SSST, ACL 2008 Workshop on Syntax and Structure in Statistical Translation.*

Percy Liang, Alexandre Bouchard-Cote, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. *In Proceedings of COLING/ACL 2006.*

Adam Lopez. 2007. Hierarchical Phrase-Based Translation with Suffix Arrays. *In Proceedings of EMNLP-CoNLL 2007.*

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. *In Proceedings of ACL 2003.*

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. *In Proceedings of ACL 2000*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. *In Proceedings of ACL 2002*.

Brian Roark, Murat Saraclar, and Michael Collins. 2007. Discriminative n-gram language modeling. *Computer Speech and Language*, 21(2):373-392.

Roni Rosenfeld. 1996. A Maximum Entropy Approach to Adaptive Statistical Language Modeling. *Computer Speech and Language*, 10(3):187-228.

Libin Shen, Anoop Sarkar and Franz Josef Och. 2004. Discriminative Reranking for Machine Translation. *In Proceedings of HLT/NAACL 2004*.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. *In Proceedings of ACL 2006*.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. *In Proceedings of the International Conference on Spoken Language Processing*, volume 2, pages 901-904.

Andreas Stolcke and Mitch Weintraub. 1998. Discriminitive language modeling. *In Proceedings of the 9th Hub-5 Conversational Speech Recognition Workshop.*

Charles Sutton, Michael Sindelar, and Andrew McCallum. 2006. Reducing Weight Undertraining in Structured Discriminative Learning. *In Proceedings of HLT-NAACL 2006*.

Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical MT. *In Proceedings of COLING/ACL 2006.*

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online Large-Margin Training for Statistical Machine Translation. *In Proceedings of EMNLP-CoNLL 2007.*

Peng Xu and Frederick Jelinek. 2004. Random Forests in Language Modeling. *In Proceedings of EMNLP 2004*.

Richard Zens and Hermann Ney. 2006. Discriminative reordering models for statistical machine translation. *In Proceedings of WSMT 2006*.

Richard Zens, Sasa Hasan, and Hermann Ney. 2007. A Systematic Comparison of Training Criteria for Statistical Machine Translation. *In Proceedings of EMNLP-CoNLL 2007.*