A Comparative Study of Discriminative Methods for Reranking LVCSR N-Best Hypotheses in Domain Adaptation and Generalization

Zhengyu Zhou, Jianfeng Gao, Frank K. Soong and Helen Meng**

Microsoft Research
{jfgao, frankkps}@microsoft.com
*The Chinese University of Hong Kong, Hong Kong
*{zyzhou, hmmeng}@se.cuhk.edu.hk

ABSTRACT

This paper is an empirical study on the performance of different discriminative approaches to reranking the N-best hypotheses output from a large vocabulary continuous speech recognizer (LVCSR). Four algorithms, namely perceptron, boosting, ranking support vector machine (SVM) and minimum sample risk (MSR), are compared in terms of domain adaptation, generalization and time efficiency. In our experiments on Mandarin dictation speech, we found that for domain adaptation, perceptron performs the best; for generalization, boosting performs the best. The best result on a domain-specific test set is achieved by the perceptron algorithm. A relative character error rate (CER) reduction of 11% over the baseline was obtained. The best result on a general test set is 3.4% CER reduction over the baseline, achieved by the boosting algorithm.

1. INTRODUCTION

The current state-of-the-art large vocabulary speech recognition system optimizes its parameters under the framework of maximum likelihood estimation. Recently, researchers adopt discriminative training approaches, whose objective is to minimize training error of the recognition system [1, 2, 3, 5, 12]. This work attempts to utilize various discriminative algorithms to improve LVCSR performance by reranking the N-best hypotheses.

There have been several previous efforts on discriminative reranking [1, 2, 5]. For example, [1] investigated the use of the perceptron algorithm under various training scenarios. Our paper follows the work in problem definition and feature selection, as well as extends the research by comparing different discriminative algorithms in terms of their performances in domain adaptation, generalization and time efficiency.

Domain adaptation refers to adapting a general LVCSR to a specific domain by using a domain-specific training corpus. In this paper, the domain-specific corpus refers to a novel-domain corpus, while the general corpus refers to a mix of different corpora, balanced among domains, styles and time. The baseline general LVCSR is trained on the general corpus. In domain adaptation experiments, we attempt to adapt the baseline LVCSR to the novel domain. Among the four algorithms we compared, i.e., perceptron, boosting, ranking SVM and MSR, the perceptron algorithm performs the best for domain adaptation, reducing the CER on a novel-domain test set from 20.0% to 17.8%.

Generalization in this work refers to the capability of generalizing the error reduction on domain-specific training data to a general test set. The objective to evaluate the discriminative algorithms in terms of generalization is to test the possibility of utilizing a domain-specific corpus to improve baseline LVCSR performance on general test set. This is especially meaningful because obtaining a large general speech corpus can be very difficult. In our generalization experiments, we attempt to improve the baseline system on general domain by training on the novel-domain corpus. The experimental results show that although the perceptron algorithm performs the best for domain adaptation, it performs the worst for generalization. Instead, boosting and ranking SVM (if ignore the training inefficiency of SVM) perform equally well, reducing the CER on a Mandarin general test set from 8.9% to 8.6%.

Our comparative study of discriminative algorithms on domain adaptation and generalization shows that it is beneficial to choose a suitable algorithm based on how well the training data matches the objective test data. We also prove that it is possible to use a domain-specific corpus to improve LVCSR performance by discriminative methods.

Three of the four algorithms in this study, namely perceptron, boosting and MSR, have previously been compare in [8] in the context of language model adaptation. This paper is the first work on comparing and analyzing the four algorithms in terms of both domain adaptation and generalization. Extending the work of [1], this is also the first attempt in applying the four discriminative algorithms for reranking LVCSR N-best hypotheses under a simple linear framework.

The rest of the paper is organized as follows: We first present the four discriminative algorithms, then describe the experiments. Finally, we compare and analyze in terms of relative performances of the algorithms.

2. DISCRIMINATIVE ALGORITHMS

This section follows [1] in defining the N-best reranking task as a linear discrimination problem. We then describe the four discriminative algorithms sequentially under this linear framework.

2.1 Problem definition

We set up the N-best reranking task based on the definitions and notations adapted from [7,8]:

- In the training data set, there are n speech utterances, and n_i

*The work was done while the first author was visiting Microsoft Research Asia.

sentence hypotheses for each utterance. Define $x_{i,j}$ as the j-th hypothesis of the i-th utterance. Define $x_{i,R}$ as the best utterance (the one with lowest CER) among $\{ x_{i,j} \}$.

- There is a separate test set of $y_{i,j}$ with similar definitions as the training set.
- Define D+1 features $f_d(h)$, d=0…D, h is a hypothesis. The features could be arbitrary functions mapping h to real values. Define the feature vector $\vec{f}(h) = \{f_0(h), f_1(h),..f_D(h)\}$.
- Define a discriminant function $g(h) = \sum_{i=0}^{D} w_i f_i(h)$. The decoding problem becomes searching for a $\vec{w}$ that satisfies the following conditions on the test set:
  $g(y_{i,R}) > g(y_{i,j}) \quad \forall i \forall j \neq R$

2.2 The perceptron algorithm

The perceptron algorithm views the N-best reranking task as a binary classification problem, attempting to find a vector $\vec{w}$ which minimizes the classification errors on training data. Instead of minimizing the training error directly, perceptron optimizes a minimum square error (MSE) loss function.

We adopt the average perceptron algorithm presented in [9] in our experiments. This approach first updates $\vec{w}$ by the standard iterative perceptron algorithm, as shown in Figure 1, then averages $\vec{w}$ using the formula below:

$$(w_d)_{avg} = (\sum_{i=1}^{t}\sum_{j=1}^{n} w_d^{i,j})/(t \cdot n) \tag{1}$$

where $w_d^{i,j}$ is the value for w_d after processing j training utterances in the ith iteration, t is the total number of iterations. Averaging the weights is to increase the model robustness [9].

```
1   Set $w_0$= 1 and $w_d$= 0, $d = 1...D$
2   For $j = 1...t$ ($t$ is the total number of iterations)
3       For each $n_i$, $i = 1...n$
4           Choose the $x_{i,j}$ with the largest $g(x_{i,j})$ value
5           For each $w_d$  ($\eta$ = size of learning step)
6               $w_d = w_d + \eta(f_d(x_{i,R}) - f_d(x_{i,j}))$
```

Figure 1. The perceptron algorithm

2.3 The boosting algorithm

Boosting has been applied to several speech recognition tasks [2, 4]. But previous work mainly focused on acoustic modeling. The boosting algorithm described in this work focuses on modeling linguistic features, and attempts to minimize the ranking error on training data. This algorithm is based on [7]. It uses the following loss function to approximate the minimum ranking error.

$$BLoss(\vec{w}) = \sum_{i=1}^{n}\sum_{j=2}^{n_i} \exp(-I[\vec{w}^t \vec{f}(x_{i,1}) - \vec{w}^t \vec{f}(x_{i,j})]) \tag{2}$$

where $I[\alpha]$=1 if $\alpha \geq 0$, and 0 otherwise. Without loss of generality, here we assume that $x_{i,R}$ always corresponds to $x_{i,1}$.

The boosting algorithm also uses an iterative approach [7, 8] to estimate $\vec{w}$, as shown in Figure 2. At each iteration, only one feature which contributes most to reducing the boosting loss function is chosen and its weight is updated. Details about feature selection and weight updating are described in [7]. Our boosting approach differs from the AdaBoost algorithm [4] in that it focuses on the most distinguishing features from the very beginning.

```
1   Set $w_0$= 1 and $w_d$= 0, $d = 1...D$
2   Select a feature $f_d$ which has largest estimated impact on
    reducing the value of boosting loss function.
3   Update $w_d$, and return to step 2.
```

Figure 2. The boosting algorithm

2.4 The ranking SVM algorithm

SVM has been used for both classification and ranking problems. We select the ranking SVM algorithm [10] because of the unbalance nature of our data sets, implementing the method by using the SVM-light toolkits [11].

The ranking SVM algorithm uses below loss function to approximate the minimum training error:

$$SLoss(\vec{w},\vec{\xi}) = \frac{1}{2}\vec{w} \cdot \vec{w} + C\sum \xi_{i,j} \tag{3}$$

This function is minimized subject to:

$$\vec{w} \cdot \vec{f}(x_{i,1}) - \vec{w} \cdot \vec{f}(x_{i,j}) \geq 1 - \xi_{i,j} \quad \forall \vec{x}_{i,1}, \vec{x}_{i,j} (j \neq 1) \tag{4}$$

$$\xi_{i,j} \geq 0 \quad \forall i \forall j \tag{5}$$

where C is a parameter trading off margin size against training error, and $\xi_{i,j}$ denote slack variables.

Inequality (4) shows that this optimization problem is a classification problem on $\vec{f}(x_{i,1}) - \vec{f}(x_{i,j})$, thus can be solved by decomposition algorithms for classification SVM.

2.5 The minimum sample risk algorithm

While all the above discriminative training methods use a loss function suitable for gradient search to simulate the minimum training error, MSR [12] attempts to minimize the training error directly. MSR first reduce the whole feature set to a small subset of highly distinguishing features, to reduce computational complexity and to ensure the generalization property. Then, in each iteration, the weights are updated one by one with a grid line search method.

Figure 4 shows the basic process of the MSR algorithm. A complete description of its implementation and the empirical justification are described in [12].

```
1   Set $w_0$= 1 and $w_d$= 0, $d = 1...D$
2   Rank all features by its expected impact on reducing training
    error, and select the top $N$ features
3   For $j = 1...t$ ($t$ is the total number of iterations)
4       For each $n = 1...N$
5           Update $w_n$ using linear search
```

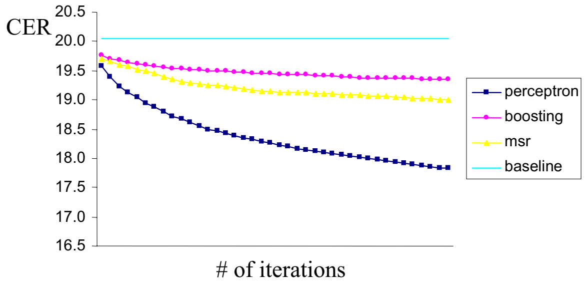
Figure 3. The MSR algorithm

3. EXPERIMENTS

3.1 Data sets

We compare the 4 discriminative training algorithms presented above on the task of Chinese dictation. We used two Mandarin dictation speech corpora in our experiments. One is a large novel-domain speech corpus with a balanced set of speakers in terms of gender and age. The other is a standard general-domain data test set (GTe-Set). We further divided the novel-domain corpus into the domain-specific training set (DTr-Set) and the domain-specific test set ((DTe-Set) in the following way: In every five utterances, we used the first four utterances for training and the fifth for testing. Table 1 shows the statistics of the three data sets.

Figure 4. Comparison on DTe_Set



Data Sets	Task	Utterance Count	Domain
DTr_Set	Training	84,498	Novel
DTe_Set	Testing	21,123	Novel
GTe_Set	Testing	500	General

Table 1. Data sets

The background (or baseline) LVCSR is a well-trained, general LVCSR. For acoustic modeling, the cross-word triphone models are trained on a seprate Mandarin dictation speech corpus of about 700 hours, collected by considering the distribution of gender and age throughout the recording. For language modeling, the trigram model is trained on about 28G (disk size) text corpora, balanced among different domains. We attempt to improve the baseline LVCSR by reranking its N-best output using discriminative methods.

3.2 Feature selection

For each of the N-best hypothesis h, we select its recognizer score (an interpolation of the acoustic and language model scores) as the base feature f_0. We define the remaining features, $f_i(h)$, $i = 1…D$, as unigram/bigram features in the following way:

1. Assign each word unigram/bigram a unique id i, $i = 1…D$
2. $f_i(h)$ is the count of the unigram/bigram with id i in h.

3.3 Evaluation metrics

We adopt two evaluation metrics for comparison purposes: (1) character error rate (CER), measured by the edit distance of character; (2) training time, which is estimated on a server with Intel Xeon CPU of 3.20 GHz.

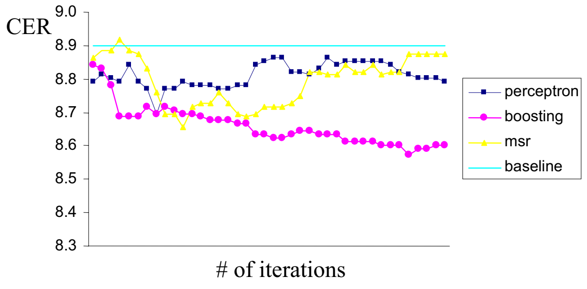
3.4 Algorithm comparison

In this subsection, we compare the performances of perceptron, boosting, ranking SVM and MSR on the two test sets. Due to the training inefficiency of the ranking SVM, only the top 20 hypotheses were adopted in training. We used 100-best hypotheses in testing. The results of the comparisons are shown in the table below.

Algorithm	Training Time	**D**Te-Set CER %	**G**Te-Set CER %
Baseline	--	20.04	8.90
Perceptron	27 minutes	17.83	8.79
Boosting	16 minutes	19.35	8.60
MSR	16 minutes	19.00	8.87
Ranking SVM	54.8 hours	19.30	8.60
Oracle	--	11.29	4.16

Table 2. Comparison of discriminative algorithms for reranking LVCSR N-best hypotheses

Figure 5. Comparison on GTe_Set



In Table 2, "Baseline" refers to the CER of the recognizer outputs, i.e., the top-scoring hypotheses. "Oracle" refers to the CER of the best hypotheses (the ones with lowest CERs) among the 100-best hypotheses. Since the recognizer is a well-trained general domain LVCSR, the baseline performance on the general dictation test set (GTe_Set) achieved a very low error rate. Since DTe_Set is a domain-specific test set with a high perplexity when measured by the language model of the recognizer, the recognition performance of baseline is worse, as expected.

In the above table, the results reported for perceptron are the performances after 40 iterations, while the results reported for boosting and MSR are the performances after 2000 iterations. Notice that changing iteration numbers may lead to different algorithm performance rankings. Figure 4 and Figure 5 show the CER changes when the number of iterations increases from 1 to 40 for perceptron, and from 1 to 2000 for boosting and MSR. The rank of the iterative algorithms is consistent on DTe_Set, as shown in Figure 4. On the other hand, as shown in Figure 5, it is much harder to compare the algorithms on GTe_Set. For examples, in Figure 5, CER curves of Perceptron and MSR are crossed, and it is difficult to make a conclusion about which method is superior. However, although the behaviors of the three iterative algorithms on GTe_Set are chaotic, the boosting algorithm seems to outperform the others, unlike the case of DTe_Set. We will present a detailed discussion in the next session.

4. DISCUSSION

4.1 Domain adaptation vs. Generalization

The background LVCSR is on general-domain and the discriminative training set is on novel-domain. Thus, the performance of the discriminative algorithms on the novel-domain test set DTe_Set can be viewed as the effects of the algorithms on domain adaptation, adapting the general LVCSR to novel-domain using novel-domain data. Similarly, the algorithms' performance on the general test set GTe_Set can be viewed as their effects on generalization, generalizing the error reduction on domain-specific training data to general test data.

The experiment results in Section 3 suggest the following observations for domain-adaptation and generalization.

Observation 1: For domain adaptation, perceptron performs the best, shown in Figure 4 and Table 2.

Observation 2: For generalization, perceptron performs the worst and boosting performs the best in most of the iterations, as shown in Figure 5 and Table 2. While perceptron and MSR fluctuate dramatically in Figure 5, boosting shows a gentle declination in CER with increasing iterations.

Observation 3: Ranking SVM provides similar CER reduction as the boosting method for both domain adaptation and generalization, as shown in Table 2.

We attempt to explain the 1st observation for domain adaptation as follows: During feature selecting and tuning, perceptron treats all training samples equally while the other three algorithms only concentrate on the most distinguishing samples [6, 11, 12]. Thus, perceptron tunes a largest set of features, making the resulting model fit the domain best.

The second observation relating to generalization is mainly due to mismatches between training and test data. In the domain-specific training data, there are two knowledge sources. One is domain-specific knowledge, referring to the rules which are applicable only in the specific domain. The other is general knowledge, referring to the rules which are also applicable in general. When testing on the general test set, the discriminating power of the features tuned to capture general knowledge will be affected by those features tuned to capture the (interfering) domain-specific knowledge. So, the performance on general test set will be the competing result of the discriminating and interfering powers. For perceptron, the great fluctuation and the poor performance of the resulting model show that perceptron tends to assign heavier weights to domain-specific information than other algorithms. This is because of its undistinguishing way of selecting and tuning features. For boosting, the rough decline with much less fluctuation shows that boosting emphasizes more on general knowledge during feature selection and tuning, which implies that the features for general knowledge are more distinguishing than the features for domain knowledge. The reason is that in the novel domain, many linguistic phenomena only appear sparsely, while a general rule may occur everywhere. Thus, when considering the distinguishing power over the whole training data, the features tuned for general knowledge will be more favored by the boosting algorithm. For MSR, the results show that MSR's attitude towards the general knowledge and domain knowledge is more balanced than perceptron and boosting.

The third observation is because although boosting and SVM adopt different loss functions and training approaches, they both attempt to maximize the minimum margin of any training sample and concentrate on the samples with smallest margin during the training [6, 11]. The two algorithms are similar in this sense.

The three observations can also be explained in terms of bias and variance. All these experimental results indicate that for perceptron, the bias is low and the variance is high. On the other hand, for boosting and SVM, the situation is reversed. MSR is somewhere in the middle. While low variance indicates a better generalization property, high variance may lead to better performance on domain adaptation. Based on the algorithms' characteristics, we can always select a suitable method for a given discriminative training /testing scenario.

4.2 Time efficiency

In Table 2, we can see that the ranking SVM algorithm needs much more training time than other algorithms. While getting the final perceptron/boosting/MSR model takes less than 30 minutes, training a SVM model takes about 55 hours. This is due to the heavy computational load of quadratic programming for SVM. However, since we do not use kernels for the ranking SVM algorithm in this work, the testing is almost equally efficient for all the four algorithms.

5. CONCLUSIONS

In this paper, we attempt to improve LVCSR performance by reranking its N-best hypotheses, using various discriminative approaches. Given a novel-domain training set, we found that discriminative reranking not only improves the recognition performance on the novel-domain test set, but also on the general test set. The best CER reduction on the general test set is from 8.9% to 8.6%; while the best CER reduction on the novel-domain test set is from 20.0% to 17.8%. This observation is especially meaningful when a large general training corpus is unavailable. We further compared the four discriminative algorithms in terms of domain adaptation, generalization and time efficiency. The results indicate that perceptron performs the best for domain adaptation while boosting and ranking SVM obtain the greatest error reduction for domain generalization. However, ranking SVM is inefficient for training, and this will be an obstacle.

6. ACKNOWLEDGMENTS

Many thanks to Ye Tian, Yu Shi, Jianlai Zhou, Frank Seide, Peng Yu from Microsoft Research Asia, Wei Yuan from the Shanghai Jiaotong University, Guihong Cao from the University of Montreal for their helps in this work. This project was partially supported by the HKSAR government under Central Allocation CUHK1/02C and also affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies.

7. REFERENCES

[1] B. Roark, M. Saraclar, and M. Collins, "Corrective language modeling for large vocabulary ASR with the perceptron algorithm", Proc. ICASSP, 2004.

[2] R. Zhang, and A.I. Rudnicky, "Apply N-best list re-ranking to acoustic model combinations of Boosting Training", Proc. ICSLP, 2004.

[3] A. Stolcke and M. Weintraub, "Discriminative language modeling", Proc. of the 9th Hub-5 Conversational Speech Recognition Workshop, 1998.

[4] C. Meyer, "Utterance-level boosting of HMM speech recognizers", Proc. ICASSP, 2002.

[5] V. Goel and W. Byrne, "Minimum bayes-risk automatic speech recognition", Computer Speech and Language, 14(2), pp. 115-135, 2000.

[6] R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods", The Annals of Statistics, 26(5), pp. 1651-1686, Oct. 1998.

[7] M. Collins, "Discriminative reranking for natural language parsing", Proc. ICML, 2000.

[8] H. Suzuki, J. Gao, "A comparative study on language model adaptation techniques using new evaluation metrics", Proc. HLT/EMNLP, 2005.

[9] M. Collins, "Discriminative training methods for hidden markov models: theory and experiments with perceptron Algorithms", Proc. EMNLP, 2002

[10] T. Joachims, "Optimizing search engines using clickthrough data", Proc. ACM, 2002.

[11] T. Joachims, *Making large-scale SVM learning practical. advances in kernel methods - support vector learning*, MIT-Press, 1999.

[12] J. Gao, H. Yu, W. Yuan and P. Xu, "Minimum sample risk methods for language modeling", Proc. HLT/EMNLP 2005.